

Final Technical Report
on the
Surface Electrical Properties
Experiment

report prepared by
The University of Toronto

and submitted to MIT
in fulfillment of the
subcontract on NASA
contract NAS9-11540

Personnel

D.W. Strangway
A.P. Annan
J.D. Redman
J.R. Rossiter
J.A. Rylaarsdam
R.D. Watts

Part II of III Parts

Digital Processing
2) Science Data Processing

R.D. Watts

SCIENCE DATA PROCESSING

The format of science (VCO) data from the acquisition system has been given in Figure 7 of J.D. Redman's report on data digitization. These data are processed in four major stages: 1) frequency and quality determination, 2) merging, 3) demultiplexing , 4) calibration.

In the notation of Figure 7 of Redman's report, one VCO measurement is recorded by 12 digits as follows

$$N = (N_1 N_2 N_3)_{10} = \text{number of 5.2 KHz cycles counted}$$

$$V = (V_1 V_2 V_3 V_4 V_5)_{10} = \text{duration of M VCO cycles } (\mu\text{sec})$$

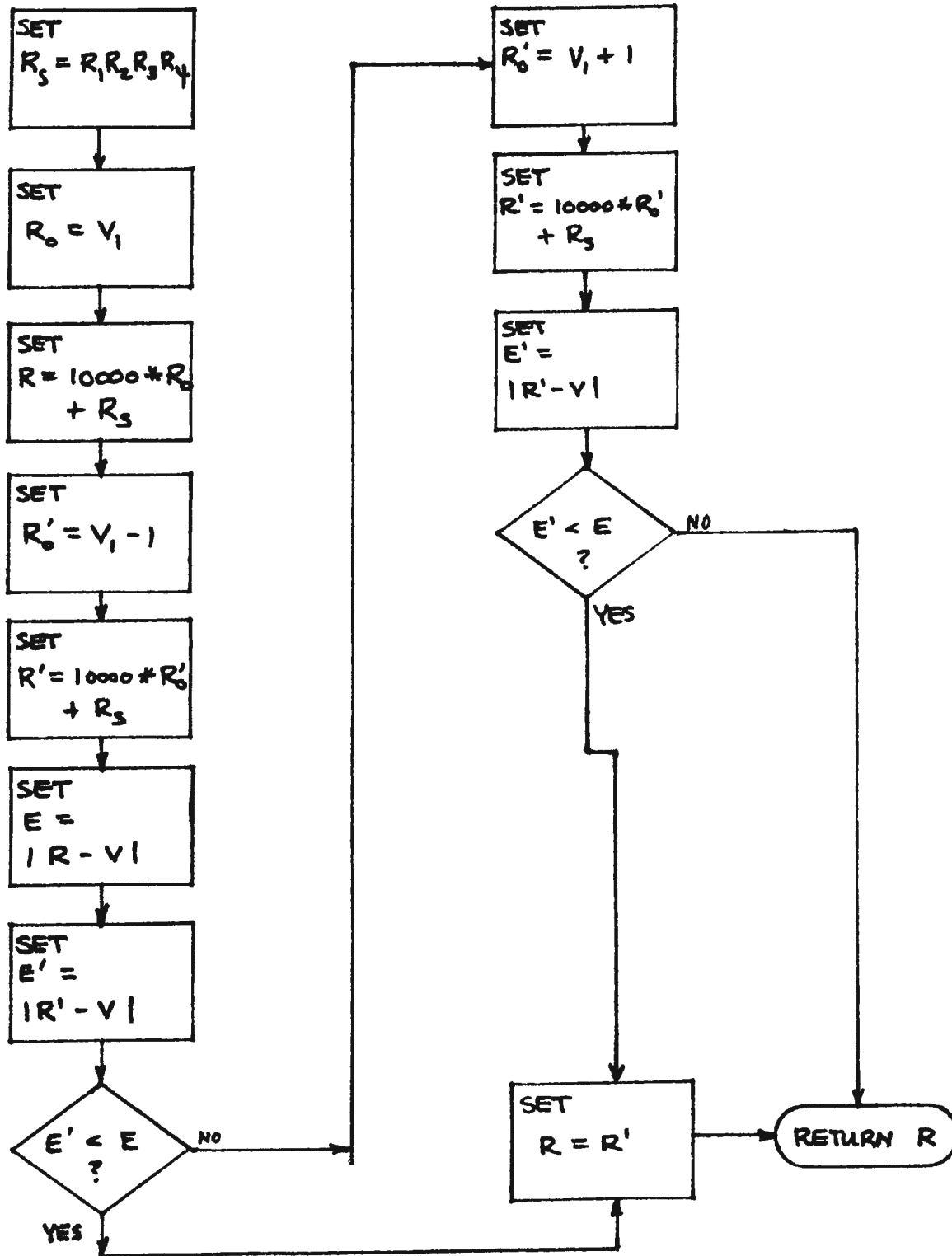
$$R = (\{R_0\} R_1 R_2 R_3 R_4)_{10} = \text{duration of N 5.2 KHz cycles}$$

Note that only the four low-order digits of R are recorded on tape. The high-order digit R_0 must be inferred from V and the operational characteristics of the DAS.

The determination of R_0 is based on the DAS design criterion that V and R should not differ by more than $1/500 \text{ sec} \approx 200 \mu\text{sec}$. Since R_0 represents the 10000's unit of μsec , R_0 can conceivably be only one less, one greater, or equal to V_1 . The algorithm diagrammed in Figure 1 bases the choice of one of these on the criterion of minimizing the difference between V and R. This function was performed in the routine FREQ.

The foregoing discussion assumed that the DAS output

FIGURE 1 DETERMINATION OF R_0



conformed to specifications. Because this was not always true, a syntax analysis was performed by the FREQ routine. Four types of errors were recognized, and an error indicator was set according to the seriousness of the errors which were found. The error indicator was set to the sum of the following individual error levels:

level 0 : no errors

1 : the measured VCO frequency was outside the range 300 - 3000 Hz., which spans the expected frequencies.

2 : the measured reference frequency was more than 100 Hz different from the mean reference frequency of 5213 Hz.

4 : the measured periods of VCO and reference cycles differed by more than 210 μ sec (1/5200 sec + 9.2%).

8 : one or more of the N, V, or R counters read zero.

Error level 8 was a terminal error, for which the frequency could not be computed and was set to zero. Error level 4 indicated a malfunction of the zero-crossing detector for the reference signal or a malfunction in the start/stop circuitry for reference period counting. However, the effects of a level 4 error could be quite small, especially

for low VCO frequencies. Error level 2 indicated either a large random fluctuation in the 5.2 KHz multivibrator in the DSEA, a tape-speed variation, or a counting problem. The tape-speed variation was probably the most frequent cause of this type of error, in which case the VCO frequency would still have been correctly determined. Error level 1 simply chopped off the allowable range of frequencies to a range spanning those observed in instrument calibration. VCO frequencies below 300 were set to 300, and those over 3000 were set to 3000.

For error levels below 8, the VCO frequency was computed by the following formulas:

$$T_{VCO} = \frac{V}{4} \times 10^{-6} \text{ sec.}$$

$$T_{REF} = \frac{R}{N} \times 10^{-6} \text{ sec.}$$

$$f_{VCO} = T_{VCO}^{-1} = \frac{4}{V} \times 10^6 \text{ Hz}$$

$$f_{REF} = \frac{N}{R} \times 10^6 \text{ Hz}$$

$$\begin{aligned} f_{VCO} - \text{CORRECTED} &= \frac{5213}{f_{REF}} f_{VCO} \\ &= \frac{5213 \times 4 \times R}{V \times N} \text{ Hz} \end{aligned}$$

The source of the correction frequency 5213 has been discussed in Redman's report on data digitization. It represents the mean actual 5.2 KHz reference frequency.

Each of the four final science tapes (SEP400-403) was processed with the program CHANGE, which used FREQ to change all VCO readings to frequency-status pairs. The output data were stored as four files on tape SEPDO4, as shown in Figure 2. Since there were two bad records on tape SEP403, the frequency values from these records were set to zero, and the error status values were set to 16.

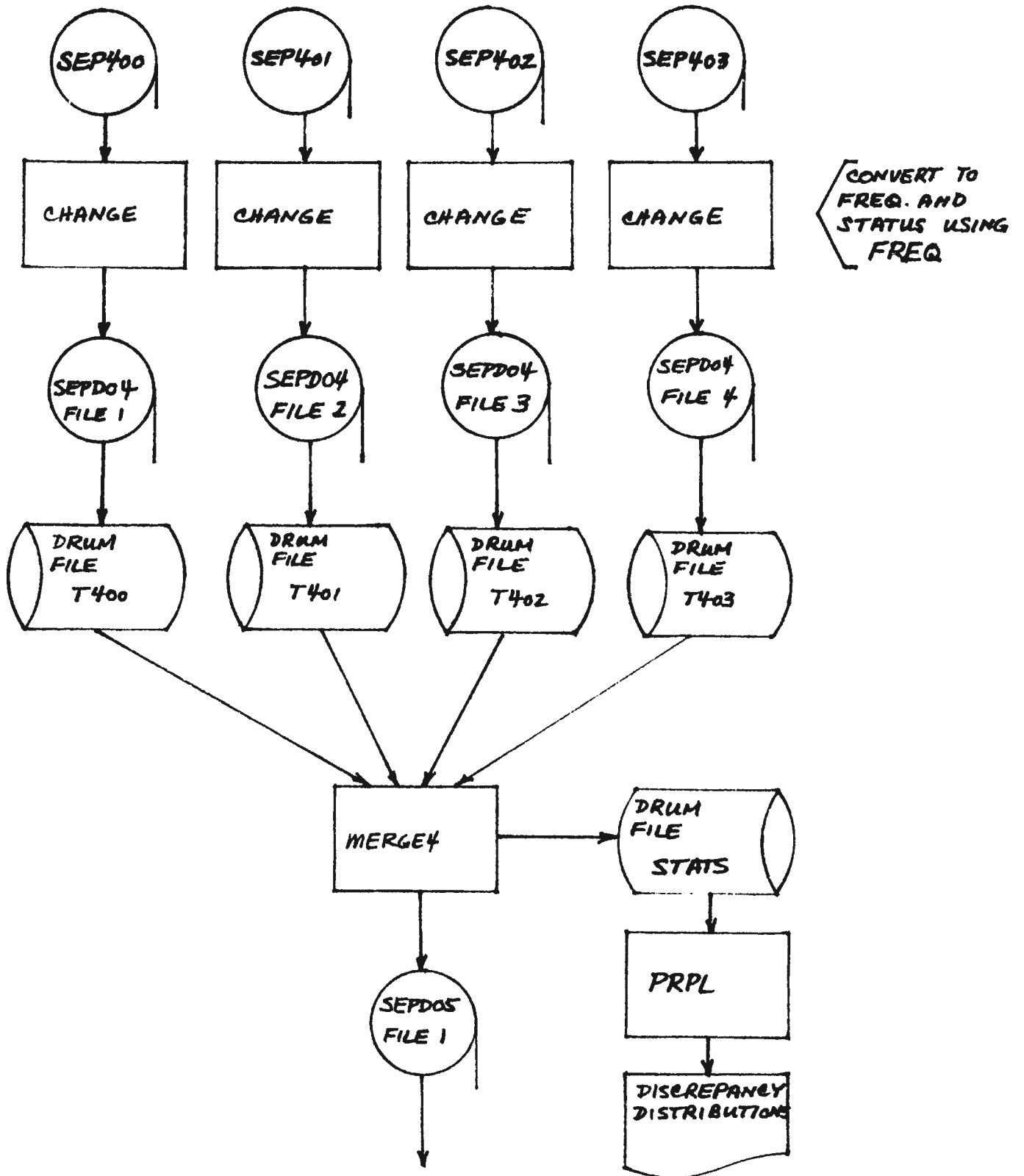
The drum-file copies of CHANGE output (T400-403) were merged by the program MERGE4. Output was stored in the first file of tape SEPDO5. Merging was performed according to the following rules.

1. Taking the four corresponding readings from files T400-403, reject those not having the lowest error status.
2. Search the values with lowest error status for the pair of values with the smallest discrepancy.
3. Average the values from the pair with smallest discrepancy. Use this value.

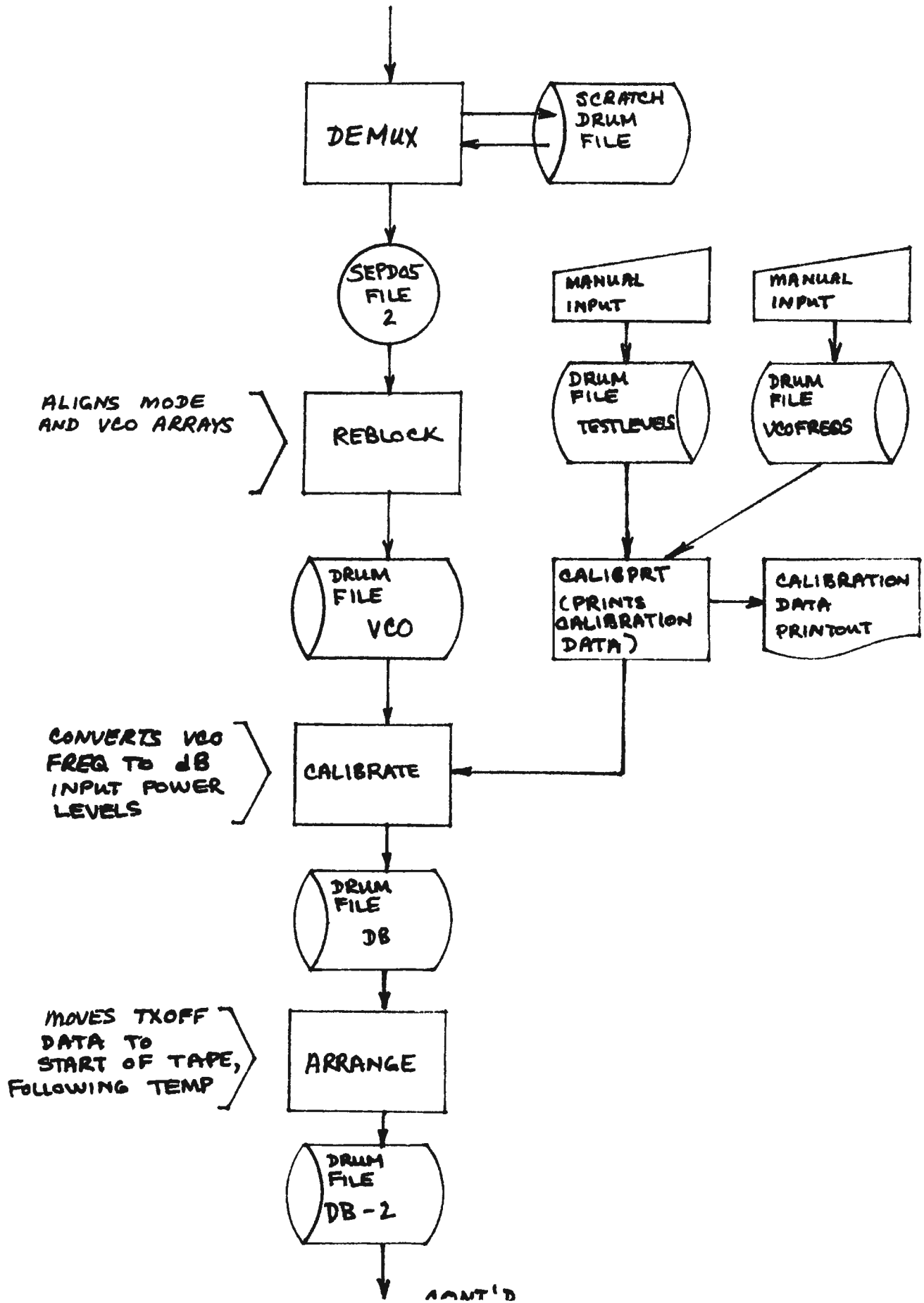
The reason for rule 1 is simply to use the most trustworthy values available, judged according to the seriousness of syntax errors. Rule 2 guarantees that the most repeatable measurements are used. Rule 3, assuming that the errors of measurement are Gaussian (which they are only approximately),

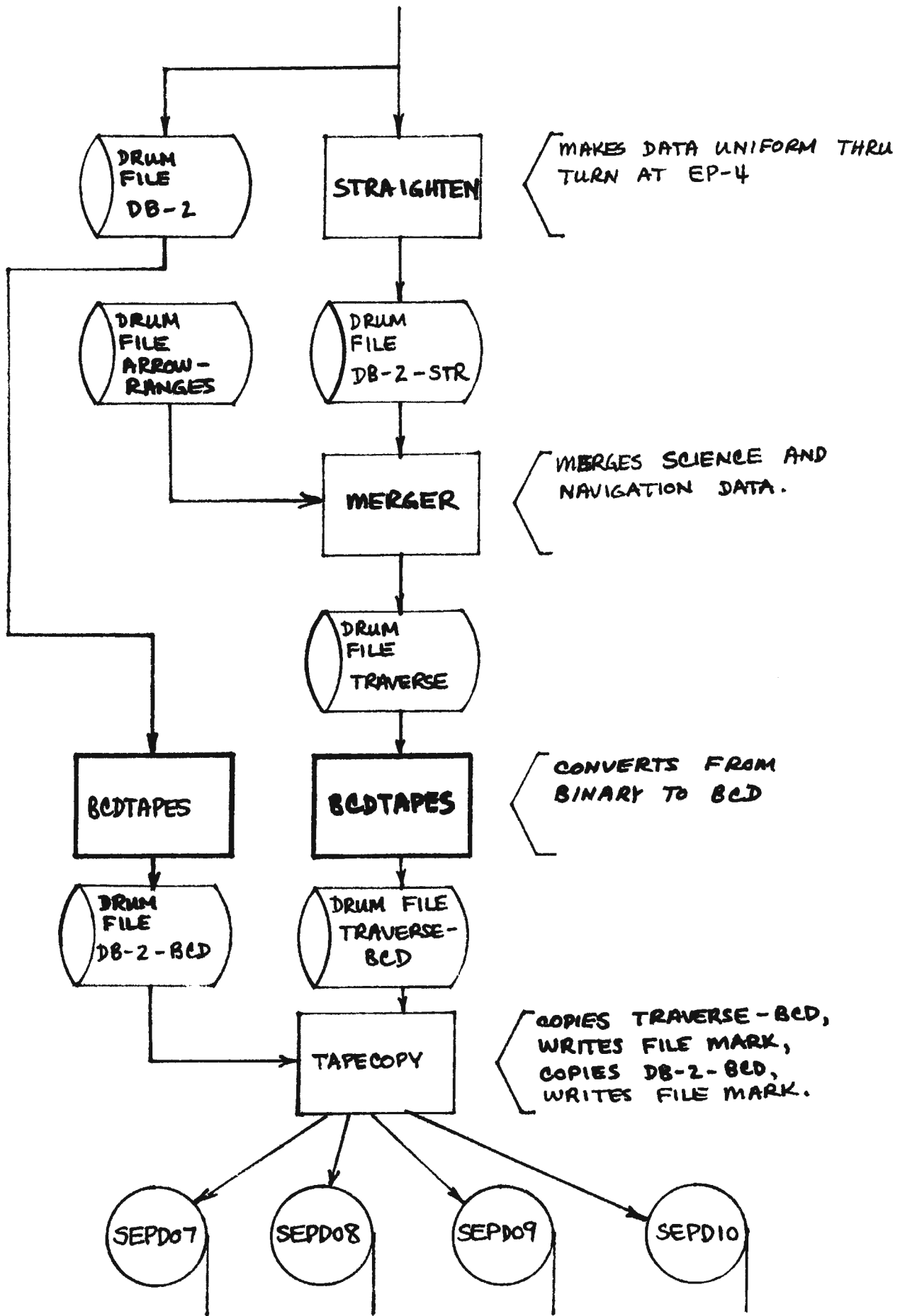
FIGURE 2

VCO DATA PROCESSING



CONT'D





reduces the measurement error by a factor $1/\sqrt{2}$.

MERGE4 provided the following ancillary functions:

1) printout of all non-zero status-value data which were used, 2) distributions of frequency discrepancies for the values from rule 3 above, broken down by 100 Hz intervals, with 5 Hz. granularity. These are shown in Figure 3. 3) printouts of all points where the discrepancy of accepted values was large (Figure 4).

The discrepancy distributions from MERGE4 were stored in the drum file STATS. Program PRPL graphed the distributions as histograms on the line printer (Figure 3). These graphs show the frequency of occurrence of various discrepancies, where the source of the first accepted value precedes the source of the second accepted value in the sequence SEP400, 401, 402, 403, and the discrepancy is the first accepted value minus the second.

Science data output from MERGE4 were still in multiplexed form (i.e., data of various types were mixed together in a known sequence). The DEMUX program demultiplexed the data, collecting all data of one type (e.g. 4 MHz NS X) into a single array. Because the memory capacity of the computer was insufficient to hold all the data, the demultiplexing was done in two stages. Input data were first demultiplexed in core,

Figure 3

Distributions of discrepancies between the two accepted values (i.e. the values with lowest error level and smallest absolute discrepancy). Data are grouped according to the mean frequency, in 100 Hz intervals, and plotted with 5 Hz granularity.

FREQUENCY INTERVAL (100HZ.) STARTS AT 300HZ.

-100I	I	.00
-95I	I	.00
-90I	I	.00
-85I	I	.00
-80I	I	.00
-75I	I	.00
-70I	I	.00
-65I	I	.00
-60I	I	.00
-55I	I	.00
-50I	I	.01
-45I	I	.00
-40I	I	.00
-35I	I	.00
-30I	I	.01
-25I	I	.03
-20I	I	.16
-15I	I	.18
-10I	I	.35
-5I*	I	1.06
0I*	I	95.87
5I*	I	.84
10I	I	.23
15I	I	.13
20I	I	.11
25I	I	.02
30I	I	.01
35I	I	.01
40I	I	.00
45I	I	.00
50I	I	.00
55I	I	.00
60I	I	.00
65I	I	.00
70I	I	.00
75I	I	.00
80I	I	.00
85I	I	.00
90I	I	.00
95I	I	.00
100I	I	.00

AVERAGE ABSOLUTE ERROR = .3HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 50CHZ.

-1001	I	.00
-951	I	.00
-901	I	.00
-851	I	.00
-801	I	.00
-751	I	.00
-701	I	.00
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.00
-451	I	.01
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.03
-51	I	.14
01	I	99.74
51	I	.06
101	I	.01
151	I	.00
201	I	.00
251	I	.00
301	I	.00
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .0HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 500HZ.

-100I	I	.06
-95I	I	.06
-90I	I	.06
-85I	I	.06
-80I	I	.06
-75I	I	.06
-70I	I	.06
-65I	I	.06
-60I	I	.06
-55I	I	.06
-50I	I	.06
-45I	I	.06
-40I	I	.06
-35I	I	.06
-30I	I	.06
-25I	I	.06
-20I	I	.06
-15I	I	.06
-10I	I	.06
-5I	I	.06
0I	I	.06
5I	I	.06
10I	I	.06
15I	I	.06
20I	I	.06
25I	I	.06
30I	I	.06
35I	I	.06
40I	I	.06
45I	I	.06
50I	I	.06
55I	I	.06
60I	I	.06
65I	I	.06
70I	I	.06
75I	I	.06
80I	I	.06
85I	I	.06
90I	I	.06
95I	I	.06
100I	I	.06

AVERAGE ABSOLUTE ERROR = .0HZ.

FREQUENCY INTERVAL (ICCHZ.) STARTS AT 60CHZ.

-100I	.00
-75I	.00
-50I	.00
-25I	.00
0I	.00
25I	.00
50I	.00
75I	.00
100I	.00
125I	.00
150I	.00
175I	.00
200I	.00
225I	.00
250I	.00
275I	.00
300I	.00
325I	.00
350I	.00
375I	.00
400I	.00
425I	.00
450I	.00
475I	.00
500I	.00
525I	.00
550I	.00
575I	.00
600I	.00
625I	.00
650I	.00
675I	.00
700I	.00
725I	.00
750I	.00
775I	.00
800I	.00
825I	.00
850I	.00
875I	.00
900I	.00
925I	.00
950I	.00
975I	.00
1000I	.00

AVERAGE ABSOLUTE ERROR = .CHZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 80CHZ.

-100	I	.00
-95	I	.00
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.00
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.00
-35	I	.00
-30	I	.00
-25	I	.00
-20	I	.00
-15	I	.00
-10	I	.00
-5	I	.00
0	I	.00
5	I	.00
10	I	.00
15	I	.00
20	I	.00
25	I	.00
30	I	.00
35	I	.00
40	I	.00
45	I	.00
50	I	.00
55	I	.00
60	I	.00
65	I	.00
70	I	.00
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.00

AVERAGE ABSOLUTE ERROR = .0CHZ.

RECQUENCY INTERVAL (10CHZ.) STARTS AT 900CHZ.

1001	I	.05
-951	I	.00
-901	I	.05
-851	I	.00
-801	I	.00
-751	I	.00
-701	I	.00
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.00
-451	I	.00
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.00
-51	I	.05
01	I	.99.36
51	I	.05
101	I	.05
151	I	.00
201	I	.00
251	I	.00
301	I	.00
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .0CHZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 100CHZ.

-1001	I	.00
-951	I	.00
-901	I	.00
-851	I	.00
-801	I	.00
-751	I	.00
-701	I	.00
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.00
-451	I	.00
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.00
-51	I	.00
01	I	.00
51	I	.00
101	I	.00
151	I	.00
201	I	.00
251	I	.00
301	I	.00
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .CHZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 1100HZ.

-1001	I	.03
-951	I	.05
-901	I	.03
-851	I	.00
-801	I	.03
-751	I	.00
-701	I	.03
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.05
-451	I	.00
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.00
-51	I	.12
01	I	.99.71
51	I	.06
101	I	.05
151	I	.00
201	I	.00
251	I	.03
301	I	.06
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .0HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 120CHZ.

-100	I	.00
-95	I	.00
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.00
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.00
-35	I	.00
-30	I	.00
-25	I	.00
-20	I	.00
-15	I	.00
-10	I	.00
-5	I	.00
0	I	.00
5	I	.00
10	I	.00
15	I	.00
20	I	.00
25	I	.00
30	I	.00
35	I	.00
40	I	.00
45	I	.00
50	I	.00
55	I	.00
60	I	.00
65	I	.00
70	I	.00
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.00
105	I	.00
110	I	.00
115	I	.00
120	I	.00
125	I	.00
130	I	.00
135	I	.00
140	I	.00
145	I	.00
150	I	.00
155	I	.00
160	I	.00
165	I	.00
170	I	.00
175	I	.00
180	I	.00
185	I	.00
190	I	.00
195	I	.00
200	I	.00
205	I	.00
210	I	.00
215	I	.00
220	I	.00
225	I	.00
230	I	.00
235	I	.00
240	I	.00
245	I	.00
250	I	.00
255	I	.00
260	I	.00
265	I	.00
270	I	.00
275	I	.00
280	I	.00
285	I	.00
290	I	.00
295	I	.00
300	I	.00
305	I	.00
310	I	.00
315	I	.00
320	I	.00
325	I	.00
330	I	.00
335	I	.00
340	I	.00
345	I	.00
350	I	.00
355	I	.00
360	I	.00
365	I	.00
370	I	.00
375	I	.00
380	I	.00
385	I	.00
390	I	.00
395	I	.00
400	I	.00
405	I	.00
410	I	.00
415	I	.00
420	I	.00
425	I	.00
430	I	.00
435	I	.00
440	I	.00
445	I	.00
450	I	.00
455	I	.00
460	I	.00
465	I	.00
470	I	.00
475	I	.00
480	I	.00
485	I	.00
490	I	.00
495	I	.00
500	I	.00
505	I	.00
510	I	.00
515	I	.00
520	I	.00
525	I	.00
530	I	.00
535	I	.00
540	I	.00
545	I	.00
550	I	.00
555	I	.00
560	I	.00
565	I	.00
570	I	.00
575	I	.00
580	I	.00
585	I	.00
590	I	.00
595	I	.00
600	I	.00
605	I	.00
610	I	.00
615	I	.00
620	I	.00
625	I	.00
630	I	.00
635	I	.00
640	I	.00
645	I	.00
650	I	.00
655	I	.00
660	I	.00
665	I	.00
670	I	.00
675	I	.00
680	I	.00
685	I	.00
690	I	.00
695	I	.00
700	I	.00
705	I	.00
710	I	.00
715	I	.00
720	I	.00
725	I	.00
730	I	.00
735	I	.00
740	I	.00
745	I	.00
750	I	.00
755	I	.00
760	I	.00
765	I	.00
770	I	.00
775	I	.00
780	I	.00
785	I	.00
790	I	.00
795	I	.00
800	I	.00
805	I	.00
810	I	.00
815	I	.00
820	I	.00
825	I	.00
830	I	.00
835	I	.00
840	I	.00
845	I	.00
850	I	.00
855	I	.00
860	I	.00
865	I	.00
870	I	.00
875	I	.00
880	I	.00
885	I	.00
890	I	.00
895	I	.00
900	I	.00
905	I	.00
910	I	.00
915	I	.00
920	I	.00
925	I	.00
930	I	.00
935	I	.00
940	I	.00
945	I	.00
950	I	.00
955	I	.00
960	I	.00
965	I	.00
970	I	.00
975	I	.00
980	I	.00
985	I	.00
990	I	.00
995	I	.00
1000	I	.00

AVERAGE ABSOLUTE ERROR = .1HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 140CHZ.

-100	I	.27
-95	I	.00
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.00
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.00
-35	I	.00
-30	I	.00
-25	I	.00
-20	I	.00
-15	I	.00
-10	I	.23
-5	I	.45
0	I	.98
5	I	.45
10	I	.00
15	I	.00
20	I	.00
25	I	.00
30	I	.00
35	I	.00
40	I	.00
45	I	.00
50	I	.00
55	I	.00
60	I	.00
65	I	.00
70	I	.00
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.27

AVERAGE ABSOLUTE ERROR = .1HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 1500HZ.

-1001	I	.00
-951	I	.00
-901	I	.00
-851	I	.00
-801	I	.00
-751	I	.00
-701	I	.00
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.00
-451	I	.00
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.00
-51	I	.99
01	I	.99
51	I	1.15
101	I	.00
151	I	.00
201	I	.00
251	I	.00
301	I	.00
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .1HZ.

FREQUENCY INTERVAL (.100HZ.) STARTS AT 1000HZ.

-100	.00
-95	.00
-90	.00
-85	.00
-80	.00
-75	.00
-70	.00
-65	.00
-60	.00
-55	.00
-50	.00
-45	.00
-40	.00
-35	.00
-30	.00
-25	.00
-20	.00
-15	.00
-10	.00
-5	.00
0	.00
5	.00
10	.00
15	.00
20	.00
25	.00
30	.00
35	.00
40	.00
45	.00
50	.00
55	.00
60	.00
65	.00
70	.00
75	.00
80	.00
85	.00
90	.00
95	.00
100	.00

AVERAGE ABSOLUTE ERROR = .1HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 1700HZ.

-100	0.00
-95	0.00
-90	0.00
-85	0.00
-80	0.00
-75	0.00
-70	0.00
-65	0.00
-60	0.00
-55	0.00
-50	0.00
-45	0.00
-40	0.00
-35	0.00
-30	0.00
-25	0.00
-20	0.00
-15	0.47
-10	0.00
-5	1.97
0	34.36
5	2.34
10	0.00
15	0.00
20	0.00
25	0.00
30	0.00
35	0.47
40	0.00
45	0.00
50	0.00
55	0.00
60	0.00
65	0.00
70	0.00
75	0.00
80	0.00
85	0.00
90	0.00
95	0.00
100	0.00

AVERAGE ABSOLUTE ERROR = .4HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 190CHZ.

-100	I	.54
-95	I	.53
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.00
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.00
-35	I	.00
-30	I	.00
-25	I	.00
-20	I	.00
-15	I	.34
-10	I	.00
-5	I	.33
0	I	.57
5	I	.33
10	I	.00
15	I	.00
20	I	.00
25	I	.00
30	I	.00
35	I	.00
40	I	.00
45	I	.00
50	I	.00
55	I	.00
60	I	.00
65	I	.00
70	I	.00
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.00

AVERAGE ABSOLUTE ERROR = .4HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 1900CHZ.

-1001	I	.00
-0951	I	.00
-0901	I	.00
-851	I	.00
-801	I	.00
-751	I	.00
-701	I	.00
-651	I	.00
-601	I	.00
-551	I	.00
-501	I	.00
-451	I	.00
-401	I	.00
-351	I	.00
-301	I	.00
-251	I	.00
-201	I	.00
-151	I	.00
-101	I	.00
-51	I	.53
01	I	5.71
51	I	84.93
101	I	7.76
151	I	.59
201	I	.23
251	I	.00
301	I	.00
351	I	.00
401	I	.00
451	I	.00
501	I	.00
551	I	.00
601	I	.00
651	I	.00
701	I	.00
751	I	.00
801	I	.00
851	I	.00
901	I	.00
951	I	.00
1001	I	.00

AVERAGE ABSOLUTE ERROR = .9HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 200CHZ.

-100	.00
-95	.00
-90	.00
-85	.00
-80	.00
-75	.00
-70	.00
-65	.00
-60	.00
-55	.00
-50	.00
-45	.00
-40	.00
-35	.00
-30	.00
-25	.00
-20	.00
-15	.00
-10	.42
-5	9.44
0	85.23
5	.42
10	.00
15	.00
20	.00
25	.00
30	.00
35	.00
40	.00
45	.00
50	.00
55	.00
60	.00
65	.00
70	.00
75	.00
80	.00
85	.00
90	.00
95	.00
100	.00

AVERAGE ABSOLUTE ERROR = .8HZ.

FREQUENCY INTERVAL (156HZ) STARTS AT 2100HZ.

```
-----  
-100I*****  
-95I .7.89  
-90I .00  
-85I .00  
-80I .00  
-75I .00  
-70I .00  
-65I .00  
-60I .00  
-55I .00  
-50I .00  
-45I .00  
-40I .00  
-35I .00  
-30I .00  
-25I .00  
-20I .00  
-15I* .61  
-10I .00  
-5I*****  
0I*****75.76  
5I*****  
10I .00  
15I .00  
20I .00  
25I .00  
30I .00  
35I .00  
40I .00  
45I .00  
50I .00  
55I .00  
60I .00  
65I .00  
70I .00  
75I .00  
80I .00  
85I .00  
90I .00  
95I .00  
100I* .61  
-----
```

AVERAGE ABSOLUTE ERROR = .9HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 220CHZ.

-100I**	
-95I	.99
-90I	.00
-85I	.00
-80I	.00
-75I	.00
-70I	.00
-65I	.00
-60I	.00
-55I	.00
-50I	.00
-45I	.00
-40I	.00
-35I	.00
-30I*	.49
-25I	.00
-20I	.00
-15I	.00
-10I****	.00
-5I*****	2.44
CI*****	117.66
SI*****	64.39
10I****	113.17
15I***	2.44
20I**	1.45
25I*	.98
30I	.00
35I	.00
40I	.00
45I	.00
50I	.00
55I	.00
60I	.00
65I	.00
70I	.00
75I	.00
80I	.00
85I	.00
90I	.00
95I	.00
100I	.00

AVERAGE ABSOLUTE ERROR = 2.4HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 2300HZ.

-100	I	.33
-95	I	.00
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.30
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.30
-35	I	1.19
-30	I	.30
-25	I	.00
-20	I	.39
-15	I	.30
-10	I	2.03
-5	I	11.87
0	I	55.83
5	I	110.39
10	I	1.43
15	I	1.49
20	I	.59
25	I	.59
30	I	1.19
35	I	.00
40	I	.00
45	I	.00
50	I	.30
55	I	.00
60	I	.00
65	I	.00
70	I	.30
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.00

AVERAGE ABSOLUTE ERROR = 3.9HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 2400HZ.

-100*	I	1.44
-95*	I	.00
-90*	I	.00
-85*	I	.00
-80*	I	.00
-75*	I	.00
-70*	I	.00
-65*	I	.00
-60*	I	.00
-55*	I	.00
-50*	I	.00
-45*	I	.00
-40*	I	.00
-35*	I	.00
-30*	I	.00
-25*	I	.00
-20*	I	.00
-15*	I	.00
-10*	I	.00
-5*	I	.00
0*	I	.00
5*	I	.00
10*	I	.00
15*	I	.00
20*	I	.00
25*	I	.00
30*	I	.00
35*	I	.00
40*	I	.00
45*	I	.00
50*	I	.00
55*	I	.00
60*	I	.00
65*	I	.00
70*	I	.00
75*	I	.00
80*	I	.00
85*	I	.00
90*	I	.00
95*	I	.00
100*	I	.00

AVERAGE ABSOLUTE ERROR = 3.3HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 250CHZ.

-100***	I 1.94
-95Y	I .00
-90I	I .00
-85I	I .00
-80I	I .00
-75Y	I .00
-70I	I .00
-65I	I .00
-60I	I .00
-55Y	I .00
-50I	I .00
-45I	I .00
-40I	I .00
-35I	I .00
-30I	I .00
-25Y	I .00
-20I	I .00
-15I	I .00
-10I***	I 1.94
-5I*****	I 19.35
0I*****	I 65.16
5I*****	I 9.53
10I*****	I 2.53
15I*	I .53
20I	I .00
25I	I .00
30I	I .00
35I	I .00
40I	I .00
45I	I .00
50I	I .00
55I	I .00
60I	I .00
65I	I .00
70I	I .00
75I	I .00
80I	I .00
85I	I .00
90I	I .00
95I	I .00
100I	I .00

AVERAGE ABSOLUTE ERROR = 2.0CHZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 2600HZ.

-100I	*****	I 4.90
-95I		I .00
-90I		I .00
-85I		I .00
-80I		I .00
-75I		I .00
-70I		I .00
-65I		I .00
-60I		I .00
-55I		I .00
-50I		I .00
-45I		I .00
-40I		I .00
-35I		I .00
-30I		I .00
-25I		I .00
-20I		I .00
-15I	***	I 1.60
-10I	*****	I 4.00
-5I	*****	I 4.00
0I	*****	I 4.00
5I	*****	I 4.00
10I	*****	I 4.00
15I		I 3.40
20I*		I 5.20
25I		I .40
30I		I .00
35I		I .00
40I		I .00
45I		I .00
50I		I .00
55I		I .00
60I		I .00
65I		I .00
70I		I .00
75I		I .00
80I		I .00
85I		I .00
90I		I .00
95I		I .00
100I*		I .40

AVERAGE ABSOLUTE ERROR = 2.5HZ.

FREQUENCY INTERVAL (100HZ.) STARTS AT 270CHZ.

-1001**	I	.34
-981	I	.00
-961*	I	.28
-941*	I	.00
-921*	I	.00
-901	I	.00
-881	I	.00
-861	I	.00
-841	I	.00
-821	I	.00
-801	I	.00
-781	I	.00
-761	I	.00
-741	I	.00
-721	I	.00
-701	I	.00
-681	I	.00
-661	I	.00
-641	I	.00
-621	I	.00
-601	I	.00
-581	I	.00
-561	I	.00
-541*	I	.29
-521*	I	.29
-501*	I	.28
-481*	I	.28
-461*	I	.29
-441*	I	.56
-421*	I	.84
-401*	I	.84
-381*	I	.84
-361*	I	.84
-341*	I	.84
-321*	I	.84
-301*	I	.84
-281*	I	.84
-261*	I	.84
-241*	I	.84
-221**	I	.84
-201**	I	.84
-181**	I	.84
-161**	I	.84
-141**	I	.84
-121**	I	.84
-101**	I	.84
-81**	I	.84
-61**	I	.84
-41**	I	.84
-21**	I	.84
101**	I	.84
121**	I	.84
141**	I	.84
161**	I	.84
181**	I	.84
201**	I	.84
221**	I	.84
241**	I	.84
261**	I	.84
281**	I	.84
301**	I	.84
321**	I	.84
341**	I	.84
361**	I	.84
381**	I	.84
401**	I	.84
421**	I	.84
441**	I	.84
461**	I	.84
481**	I	.84
501**	I	.84
521**	I	.84
541**	I	.84
561**	I	.84
581**	I	.84
601**	I	.84
621**	I	.84
641**	I	.84
661**	I	.84
681**	I	.84
701**	I	.84
721**	I	.84
741**	I	.84
761**	I	.84
781**	I	.84
801**	I	.84
821**	I	.84
841**	I	.84
861**	I	.84
881**	I	.84
901**	I	.84
921**	I	.84
941**	I	.84
961**	I	.84
981**	I	.84
1001**	I	.84

AVERAGE ABSOLUTE ERROR = 5.9HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 260CHZ.

-100*	I	.00
-95*	I	.00
-90*	I	.00
-85*	I	.00
-80*	I	.00
-75*	I	.00
-70*	I	.00
-65*	I	.30
-60*	I	.30
-55*	I	.00
-50*	I	.30
-45*	I	.00
-40*	I	.30
-35*	I	.30
-30*	I	.91
-25*	I	.91
-20*	I	1.52
-15*	I	2.44
-10*	I	7.62
-5*	I	15.95
0*	I	40.85
5*	I	11.59
10*	I	7.62
15*	I	1.93
20*	I	1.83
25*	I	1.22
30*	I	1.52
35*	I	.51
40*	I	.61
45*	I	.30
50*	I	.61
55*	I	.30
60*	I	.00
65*	I	.00
70*	I	.00
75*	I	.00
80*	I	.00
85*	I	.00
90*	I	.00
95*	I	.00
100*	I	.30

AVERAGE ABSOLUTE ERROR = 7.3HZ.

FREQUENCY INTERVAL (10CHZ.) STARTS AT 290CHZ.

-100	I	.00
-95	I	.00
-90	I	.00
-85	I	.00
-80	I	.00
-75	I	.00
-70	I	.00
-65	I	.00
-60	I	.00
-55	I	.00
-50	I	.00
-45	I	.00
-40	I	.00
-35	I	.00
-30	I	.00
-25	I	.00
-20	I	.00
-15	I	.00
-10	I	.00
-5	I	.00
0	I	.00
5	I	.00
10	I	.00
15	I	.00
20	I	.00
25	I	.00
30	I	.00
35	I	.00
40	I	.00
45	I	.00
50	I	.00
55	I	.00
60	I	.00
65	I	.00
70	I	.00
75	I	.00
80	I	.00
85	I	.00
90	I	.00
95	I	.00
100	I	.00

AVERAGE ABSOLUTE ERROR = 9.0HZ.

RECORD	1	WORD	54	STATUS	0	TAPE	0	FREQ	1983.	TAPE	1	FREQ	2439.
RECORD	1	WORD	121	STATUS	0	TAPE	0	FREQ	2365.	TAPE	3	FREQ	2979.
RECORD	1	WORD	174	STATUS	0	TAPE	0	FREQ	2356.	TAPE	1	FREQ	2952.
RECORD	2	WORD	7	STATUS	0	TAPE	2	FREQ	2998.	TAPE	3	FREQ	2955.
RECORD	2	WORD	53	STATUS	0	TAPE	0	FREQ	2400.	TAPE	1	FREQ	2961.
RECORD	2	WORD	162	STATUS	0	TAPE	0	FREQ	2428.	TAPE	1	FREQ	2359.
RECORD	2	WORD	174	STATUS	0	TAPE	0	FREQ	1972.	TAPE	1	FREQ	2392.
RECORD	3	WORD	15	STATUS	0	TAPE	0	FREQ	2344.	TAPE	3	FREQ	2990.
RECORD	3	WORD	19	STATUS	0	TAPE	0	FREQ	1990.	TAPE	1	FREQ	2346.
RECORD	3	WORD	27	STATUS	0	TAPE	0	FREQ	2906.	TAPE	1	FREQ	2377.
RECORD	3	WORD	51	STATUS	0	TAPE	1	FREQ	1489.	TAPE	2	FREQ	1967.
RECORD	3	WORD	59	STATUS	2	TAPE	0	FREQ	2968.	TAPE	1	FREQ	2966.
RECORD	3	WORD	73	STATUS	0	TAPE	0	FREQ	2415.	TAPE	2	FREQ	2961.
RECORD	3	WORD	75	STATUS	0	TAPE	0	FREQ	2407.	TAPE	2	FREQ	2997.
RECORD	3	WORD	102	STATUS	0	TAPE	0	FREQ	2443.	TAPE	1	FREQ	2942.
RECORD	3	WORD	103	STATUS	0	TAPE	0	FREQ	2387.	TAPE	3	FREQ	2920.
RECORD	3	WORD	147	STATUS	0	TAPE	0	FREQ	2426.	TAPE	3	FREQ	2977.
RECORD	3	WORD	149	STATUS	0	TAPE	0	FREQ	2961.	TAPE	1	FREQ	2999.
RECORD	4	WORD	123	STATUS	0	TAPE	0	FREQ	1965.	TAPE	1	FREQ	2904.
RECORD	4	WORD	174	STATUS	0	TAPE	0	FREQ	2377.	TAPE	1	FREQ	2467.
RECORD	5	WORD	18	STATUS	0	TAPE	0	FREQ	1999.	TAPE	1	FREQ	2414.
RECORD	5	WORD	54	STATUS	0	TAPE	0	FREQ	2322.	TAPE	1	FREQ	2437.
RECORD	5	WORD	97	STATUS	0	TAPE	0	FREQ	2394.	TAPE	3	FREQ	2954.
RECORD	5	WORD	151	STATUS	0	TAPE	0	FREQ	2384.	TAPE	1	FREQ	2922.
RECORD	5	WORD	195	STATUS	0	TAPE	0	FREQ	2765.	TAPE	1	FREQ	2700.
RECORD	6	WORD	1	STATUS	0	TAPE	0	FREQ	2401.	TAPE	1	FREQ	1983.
RECORD	6	WORD	5	STATUS	0	TAPE	0	FREQ	2397.	TAPE	2	FREQ	2970.
RECORD	6	WORD	25	STATUS	0	TAPE	0	FREQ	1950.	TAPE	1	FREQ	2357.
RECORD	6	WORD	30	STATUS	0	TAPE	0	FREQ	2364.	TAPE	2	FREQ	2965.
RECORD	6	WORD	39	STATUS	0	TAPE	0	FREQ	2924.	TAPE	2	FREQ	2927.
RECORD	6	WORD	45	STATUS	0	TAPE	1	FREQ	2719.	TAPE	2	FREQ	2911.
RECORD	6	WORD	49	STATUS	0	TAPE	0	FREQ	1957.	TAPE	1	FREQ	2374.
RECORD	6	WORD	54	STATUS	0	TAPE	0	FREQ	2016.	TAPE	1	FREQ	2999.
RECORD	6	WORD	97	STATUS	0	TAPE	0	FREQ	1958.	TAPE	1	FREQ	2364.
RECORD	6	WORD	100	STATUS	0	TAPE	0	FREQ	2239.	TAPE	1	FREQ	2923.
RECORD	6	WORD	121	STATUS	0	TAPE	0	FREQ	2450.	TAPE	1	FREQ	2352.
RECORD	6	WORD	122	STATUS	0	TAPE	1	FREQ	1919.	TAPE	2	FREQ	2920.
RECORD	6	WORD	126	STATUS	1	TAPE	0	FREQ	3000.	TAPE	1	FREQ	3000.
RECORD	6	WORD	139	STATUS	5	TAPE	1	FREQ	2374.	TAPE	2	FREQ	2954.
RECORD	6	WORD	140	STATUS	0	TAPE	0	FREQ	2265.	TAPE	2	FREQ	2914.
RECORD	6	WORD	144	STATUS	0	TAPE	0	FREQ	1397.	TAPE	1	FREQ	1598.
RECORD	6	WORD	146	STATUS	0	TAPE	0	FREQ	1956.	TAPE	1	FREQ	2332.
RECORD	6	WORD	149	STATUS	0	TAPE	0	FREQ	1947.	TAPE	1	FREQ	2375.
RECORD	6	WORD	150	STATUS	0	TAPE	0	FREQ	2363.	TAPE	1	FREQ	2422.
RECORD	6	WORD	170	STATUS	0	TAPE	1	FREQ	2913.	TAPE	2	FREQ	2959.
RECORD	6	WORD	173	STATUS	0	TAPE	0	FREQ	1957.	TAPE	2	FREQ	2950.
RECORD	7	WORD	2	STATUS	0	TAPE	0	FREQ	1983.	TAPE	1	FREQ	2353.
RECORD	7	WORD	5	STATUS	0	TAPE	0	FREQ	1981.	TAPE	1	FREQ	2326.
RECORD	7	WORD	19	STATUS	0	TAPE	1	FREQ	2960.	TAPE	2	FREQ	2923.
RECORD	7	WORD	49	STATUS	0	TAPE	0	FREQ	1961.	TAPE	1	FREQ	2413.
RECORD	7	WORD	126	STATUS	0	TAPE	0	FREQ	2430.	TAPE	1	FREQ	2370.
RECORD	8	WORD	76	STATUS	0	TAPE	0	FREQ	2414.	TAPE	1	FREQ	2997.
RECORD	8	WORD	122	STATUS	0	TAPE	2	FREQ	2966.	TAPE	3	FREQ	2994.
RECORD	9	WORD	14	STATUS	0	TAPE	0	FREQ	2266.	TAPE	1	FREQ	2904.
RECORD	9	WORD	62	STATUS	0	TAPE	2	FREQ	2786.	TAPE	3	FREQ	2948.
RECORD	16	WORD	51	STATUS	2	TAPE	0	FREQ	1776.	TAPE	2	FREQ	1776.
RECORD	26	WORD	42	STATUS	2	TAPE	1	FREQ	1395.	TAPE	3	FREQ	1395.
RECORD	32	WORD	132	STATUS	0	TAPE	0	FREQ	1775.	TAPE	3	FREQ	1187.
RECORD	46	WORD	194	STATUS	2	TAPE	2	FREQ	1210.	TAPE	3	FREQ	1210.
RECORD	47	WORD	97	STATUS	2	TAPE	0	FREQ	860.	TAPE	1	FREQ	860.

MERGE4 OUTPUT ACCEPTED VALUES WITH NON-ZERO ERROR LEVEL OR LARGE DISCREPANCY

FIGURE 4

RECORD	58	WORD	10	STATUS	2	TAPE	0	ERROR	950.	TAPE	1	ERROR	950.
RECORD	64	WORD	100	STATUS	2	TAPE	0	ERROR	740.	TAPE	2	ERROR	740.
RECORD	89	WORD	180	STATUS	2	TAPE	1	ERROR	1979.	TAPE	2	ERROR	1980.
RECORD	116	WORD	44	STATUS	2	TAPE	1	ERROR	955.	TAPE	2	ERROR	955.
RECORD	121	WORD	98	STATUS	2	TAPE	2	ERROR	374.	TAPE			
RECORD	122	WORD	110	STATUS	2	TAPE	0	ERROR	474.	TAPE	1	ERROR	474.
RECORD	123	WORD	24	STATUS	2	TAPE	2	ERROR	971.	TAPE	3	ERROR	971.
RECORD	154	WORD	143	STATUS	2	TAPE	2	ERROR	455.	TAPE	3	ERROR	455.
RECORD	271	WORD	113	STATUS	2	TAPE	0	ERROR	500.	TAPE	2	ERROR	500.
RECORD	343	WORD	73	STATUS	2	TAPE	0	ERROR	452.	TAPE	3	ERROR	452.
RECORD	352	WORD	67	STATUS	2	TAPE	1	ERROR	562.	TAPE	2	ERROR	562.
RECORD	358	WORD	132	STATUS	2	TAPE	1	ERROR	411.	TAPE	2	ERROR	411.
RECORD	358	WORD	170	STATUS	2	TAPE	2	ERROR	598.	TAPE	3	ERROR	598.
RECORD	383	WORD	79	STATUS	2	TAPE	0	ERROR	331.	TAPE	1	ERROR	327.

END OF FILE WRITTEN ON UNIT 4

387 RECORDS PROCESSED.

STATUS COUNTS

0	72736
1	1
2	19
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0

**NUMBER OF ACCEPTANCES
AT EACH ERROR LEVEL.**



NORMAL EXIT. EXECUTION TIME:

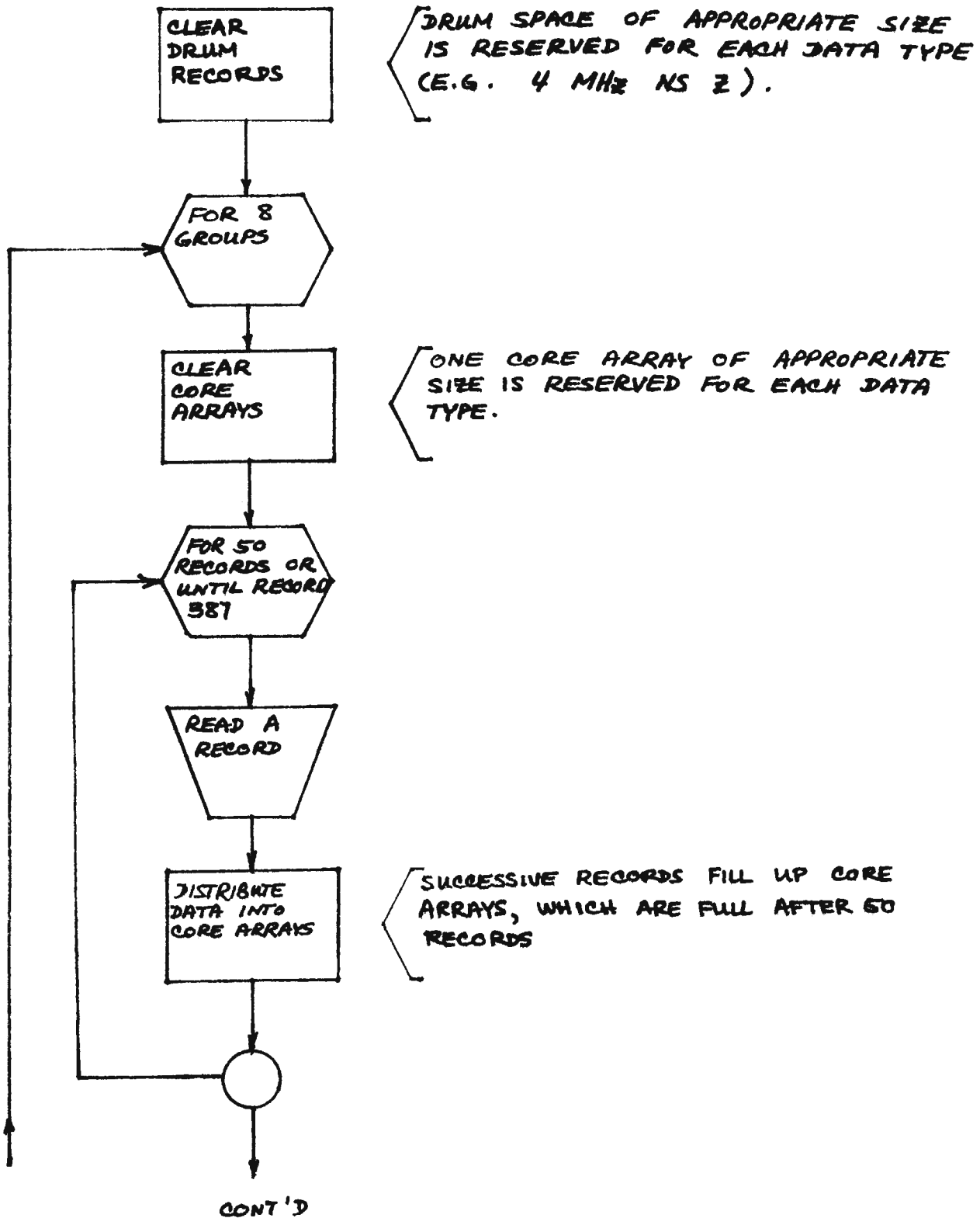
53044 MILLISECONDS.

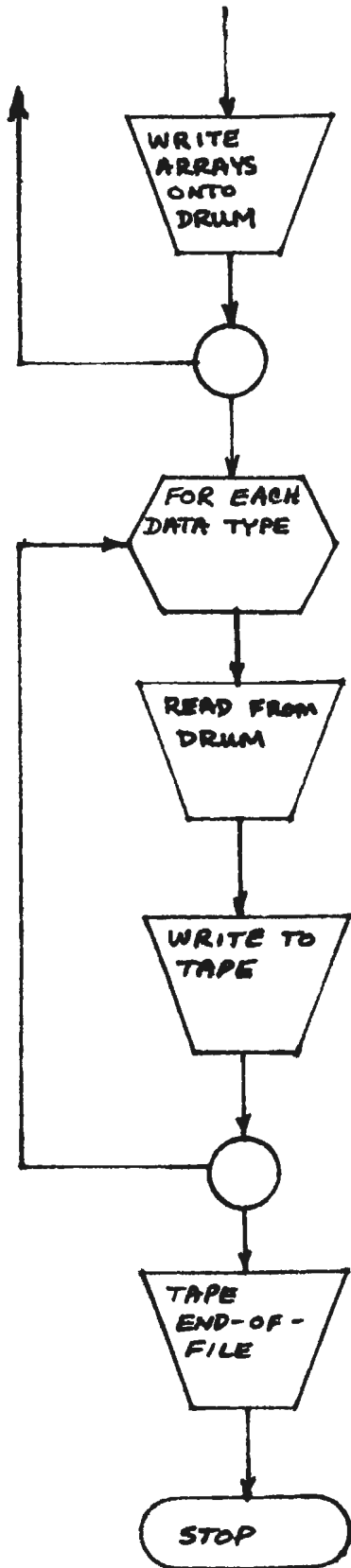
50 records at a time, then output to random-access drum. The next 50 records were then demultiplexed in core, and these data were concatenated to the previous data on the drum. The random access feature was used to skip over space reserved for as yet unprocessed data. Figure 5 shows the general flow of the DEMUX program as well as diagrams of core and drum space allocations.

The DEMUX program contained a coding error which deleted data from the arrays for 4, 8, 16, and 32 MHz. The error occurred in the final transfer of data from drum to tape, in subroutine TAPER (internal subroutine in DEMUX). Figure 6 diagrams the nature of the error. In the tape output routine, the entire $400*N$ data array (N is the number of samples per record of the data type being considered) should have been processed at once. In Figure 6, the data array in the drum diagram should have been written on tape as 4 contiguous parts of 387 values, truncated to 387. The effect was a compression of the data - 13 values missing after every 387 values on tape. There are $(N-1)$ such error regions. At the end of the tape arrays, there is garbage of length $13* (N-1)$.

The error is complicated somewhat by the next stage of processing. Because of the design of the DAS, the mode word appearing in record i predicted the receiver mode in

FIGURE 5 DEMUX - DEMULTIPLEXING FLOW





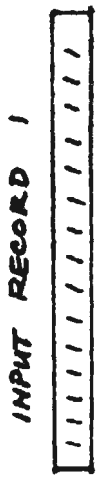
DATA IS APPENDED TO PREVIOUSLY WRITTEN DATA OF SAME TYPE ON DRUM.

ALL DATA OF ONE TYPE (387 x N SAMPLES PER RECORD)

N BLOCKS OF 387 WORDS

CORE DEMULTIPLEXING

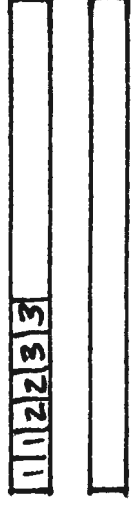
CORE



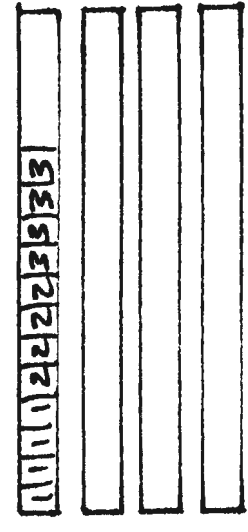
⋮



⋮



⋮



⋮

MODE

TEMP

1 MHz NS X

1 MHz NS Y

4 MHz NS X {

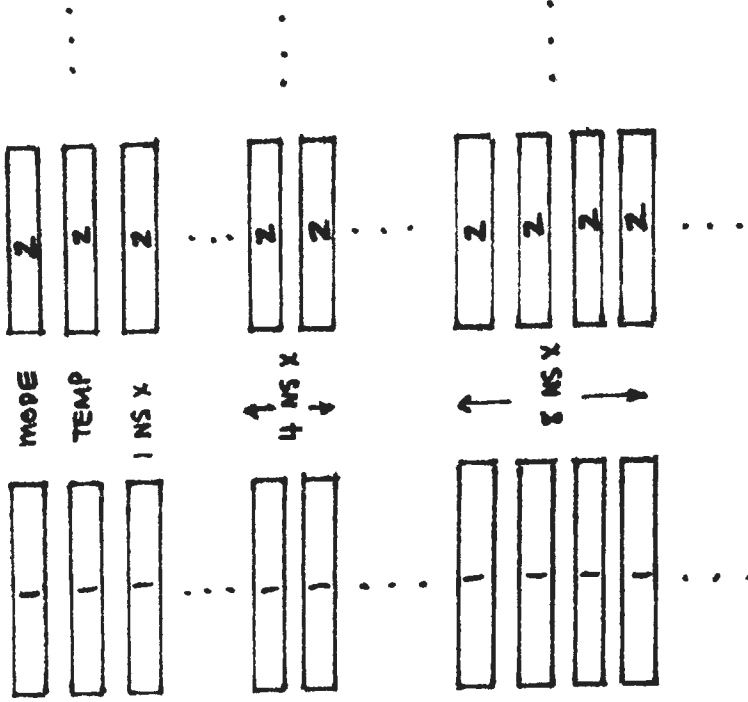
4 MHz NS Y {

8 MHz NS X

DRUM DEMULTIPLEXING

CORE ARRAYS

1ST 50 RECORDS 2ND 50 RECORDS



DRUM

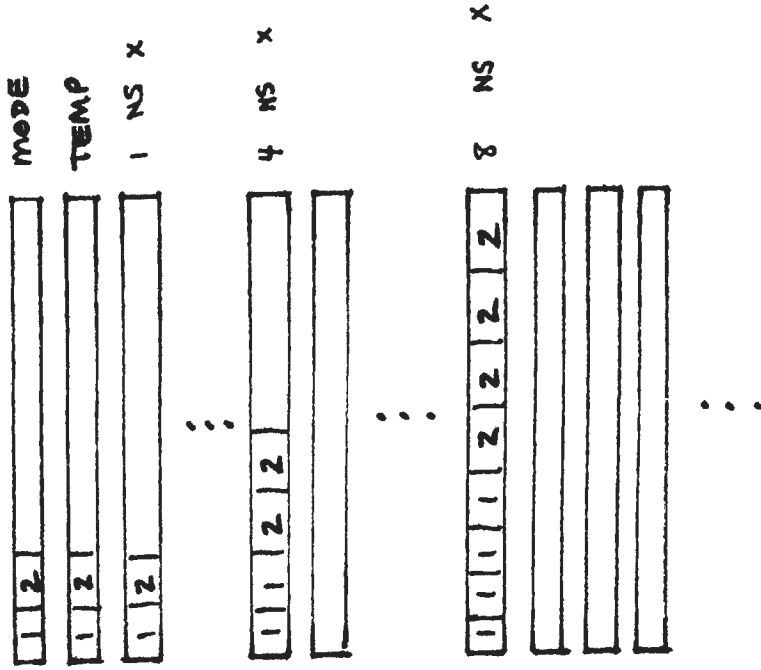
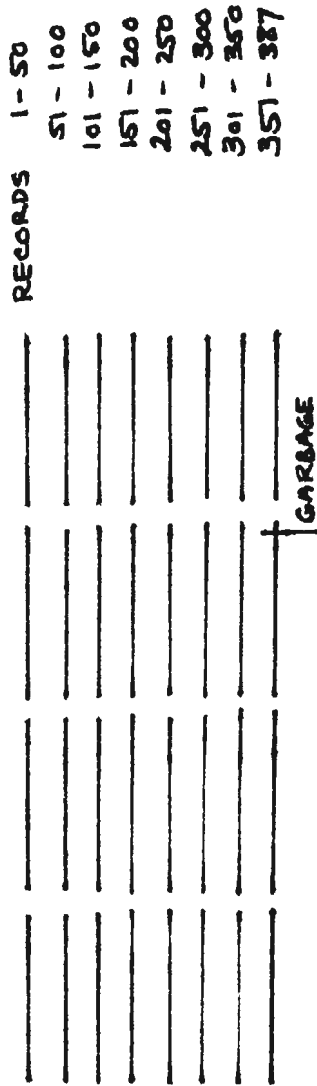
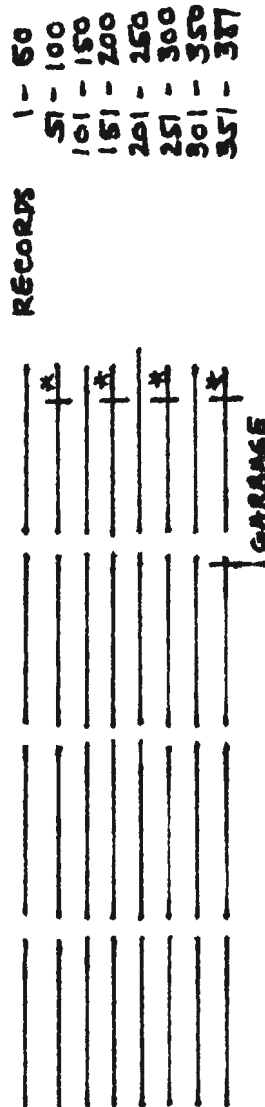


FIGURE 6 DEMULTIPLEXING ERROR
 EXAMPLE: 8 MHz DATA (4 SAMPLES / INPUT RECORD)

DRAM:



TAPE:



* DENOTES AND END-OF-RECORD AS WRITTEN ONTO TAPE,
 WITH THE NEXT 13 VALUES SKIPPED (THE NEXT RECORD STARTS
 13 VALUES LATER), FOR THIS EXAMPLE, THE FINAL 39
 VALUES ARE GARBAGE.

record $i + 1$. To make the mode array agree exactly with the data arrays, with no shift, N points were dropped from the start of each data array. This was accomplished by reading N blocks of 387 values (concatenated), deleting the first N values, and writing N blocks of 386 values. This was done by the REBLOCK program. The output was stored in drum file VCO.

Location of errors - data is missing after the following locations:

	SAMPLE #	APPARENT TIME FROM START
4 MHz	385	0:20:47
8 MHz	383	0:10:20
	770	0:20:47
	1157	0:31:14
16 MHz	379	0:05:07
	766	0:10:20
	1153	0:15:34
	1540	0:20:47
	1927	0:26:01
	2314	0:31:14
	2701	0:36:28
32 MHz	374	0:03:06
	761	0:06:19
	1148	0:09:32

SAMPLE #	APPARENT TIME FROM START
1535	0:12:45
1922	0:15:58
2309	0:19:11
2696	0:22:24
3083	0:25:37
3470	0:28:50
3857	0:32:03
4244	0:35:15

To correct the timing errors, 13 filler values should be inserted following each of the tabulated error locations. This should be done with care, since at 8, 16, and 32 MHz the insertion of the first 13 fillers moves the location for insertion of subsequent sets of 13 fillers. The last 13* (N-1) values should be discarded.

The next step of processing was conversion from VCO frequencies to dB values for scientific data, and from VCO frequency to temperature. The calibration data for the flight unit were typed in and stored in drum files TESTLEVELS and VCOFREQS. These data are printed out in Figure 7. The receiver was operating in the medium (65°) to hot (105°) and above temperature range during the entire traverse, so the cold calibration was ignored.

A two-dimensional linear interpolation was done by

CALIBRATE, first determining the temperature, then using the temperature to interpolate between the medium and high temperature dB vs. frequency curves. Since the calibration data were given as VCO frequency as a function of dB input power, the inverse function first had to be determined. This was done by linear interpolation for VCO frequencies in 1 Hz steps from 300 Hz to 3000 Hz. The dB levels for frequencies below/above the lowest/highest calibrated value were set to the lowest/highest value. Interpolation was not done between these 1 Hz values - they were used purely as a lookup table. The granularity of the data (near the center of the calibration curves) is about 1/27 dB, or .9% on a linear power scale.

Up to this time, there has been no mention of the disposition of the following arrays: MODE, TEMP, TXOFF, CAL. The contents of each will be described here.

The MODE array consists of 386 words of the form $(MAR)_{10}$, where M (the 100's digit) is 1 or 2 depending on whether the receiver was in sync'ed or sync-search mode during the i^{th} record, corresponding to the i^{th} word in the MODE array. The 10's digit, A, is the antenna indicator. Values 1, 2, and 3 correspond to the X, Y, and Z antennas during synch search and to the

FIGURE 7. FLIGHT UNIT CALIBRATION DATA

1 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-134.23	380	366	349	378	360	342	383	373	352
-129.24	418	400	376	412	390	366	424	408	383
-119.29	645	631	574	936	614	559	653	640	580
-109.37	947	940	899	944	936	893	950	943	901
-99.41	1202	1195	1168	1201	1196	1167	1203	1198	1170
-89.54	1420	1424	1418	1421	1426	1417	1422	1426	1420
-79.75	1684	1686	1679	1684	1686	1679	1684	1686	1680
-69.62	1967	1959	1952	1968	1959	1952	1968	1959	1953
-59.68	2259	2239	2225	2260	2239	2225	2259	2232	2225
-49.74	2517	2492	2471	2518	2492	2471	2517	2492	2472
-39.80	2769	2759	2744	2770	2758	2743	2769	2758	2744
-34.84	2897	2880	2873	2880	2876	2873	2879	2879	2874
-29.86	2946	2962	2970	2948	2960	2969	2946	2959	2969

2 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-134.56	387	376	356	394	378	356	387	379	355
-129.56	442	422	391	449	424	390	442	421	393
-119.61	691	680	620	698	682	620	694	683	625
-109.69	972	964	924	972	962	919	974	966	925
-99.72	1217	1213	1190	1216	1212	1186	1218	1215	1191
-89.78	1435	1441	1435	1434	1441	1433	1435	1442	1435
-79.88	1698	1701	1689	1698	1700	1698	1699	1703	1698
-69.93	1982	1973	1970	1982	1974	1970	1983	1975	1970
-60.01	2272	2253	2242	2272	2253	2242	2273	2254	2243
-50.04	2530	2507	2489	2529	2502	2489	2532	2508	2489
-40.10	2781	2774	2763	2781	2775	2763	2783	2776	2763
-35.14	2890	2895	2892	2890	2894	2892	2892	2895	2892
-30.16	2948	2967	2979	2947	2965	2978	2951	2966	2978

4 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-135.05	392	380	361	393	382	361	394	380	360
-130.06	447	430	399	448	429	401	449	428	400
-120.12	701	690	637	696	690	637	698	689	637
-110.20	980	974	938	980	975	938	979	974	937
-100.23	1446	1224	1204	1225	1225	1204	1224	1223	1203
-90.29	1626	1451	1447	1444	1454	1448	1444	1452	1447
-80.38	1709	1713	1712	1709	1714	1712	1707	1715	1712
-70.44	1993	1986	1984	1993	1987	1983	1993	1986	1983
-60.52	2280	2263	2255	2282	2265	2252	2281	2263	2254
-50.55	2540	2518	2501	2540	2519	2501	2539	2518	2501
-40.63	2789	2783	2774	2789	2784	2774	2789	2784	2774
-35.66	2895	2900	2900	2895	2901	2899	2895	2901	2899
-30.67	2947	2967	2980	2950	2966	2978	2952	2969	2978

8 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-134.86	437	427	401	433	430	401	435	430	403
-129.86	515	505	464	515	501	462	515	502	461
-119.98	789	795	747	789	795	747	792	796	745
-110.02	1059	1055	1018	1060	1055	1019	1060	1055	1019
-100.04	1292	1294	1283	1292	1295	1283	1293	1295	1284
-90.11	1524	1532	1527	1524	1532	1528	1527	1533	1528
-80.18	1797	1799	1798	1796	1801	1800	1798	1802	1798
-70.26	2083	2074	2068	2084	2074	2069	2084	2075	2068
-60.34	2361	2340	2327	2361	2341	2327	2361	2341	2327
-50.36	2618	2601	2583	2618	2600	2583	2618	2601	2583
-40.41	2855	2857	2851	2855	2857	2851	2856	2858	2851
-35.44	2937	2950	2955	2938	2949	2955	2938	2951	2955
-30.47	2949	2970	2984	2954	2969	2983	2953	2973	2983

16 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-134.58	430	430	408	433	437	412	433	431	410
-129.60	504	501	468	521	511	480	514	510	471
-119.80	777	795	756	780	801	765	781	797	758
-109.91	1051	1059	1033	1053	1061	1035	1053	1059	1032
-99.92	1285	1299	1297	1286	1300	1299	1287	1300	1298
-90.00	1517	1538	1543	1518	1538	1545	1519	1539	1544
-80.11	1799	1813	1817	1802	1810	1821	1799	1812	1822
-70.31	2076	2081	2082	2078	2080	2082	2077	2081	2083
-60.42	2354	2344	2338	2354	2345	2340	2354	2345	2339
-50.43	2612	2605	2596	2612	2605	2597	2612	2605	2596
-40.51	2853	2865	2868	2854	2865	2867	2854	2865	2866
-35.54	2940	2956	2967	2940	2955	2966	2940	2955	2965
-30.57	2955	2971	2987	2953	2969	2984	2952	2969	2983

222:

32 MHZ

DBM	X ANTENNA			Y ANTENNA			Z ANTENNA		
	COLD	MED	HOT	COLD	MED	HOT	COLD	MED	HOT
-133.45	415	399	378	430	411	385	423	407	383
-128.49	499	464	410	518	480	442	507	474	443
-119.06	762	745	697	770	760	698	769	755	700
-109.26	1044	1030	985	1046	1030	987	1044	1030	987
-99.34	1282	1276	1238	1282	1277	1257	1281	1276	1256
-89.42	1513	1513	1499	1513	1513	1501	1512	1513	1501
-79.62	1806	1800	1788	1782	1803	1789	1802	1794	1791
-70.21	2066	2048	2034	2064	2044	2031	2064	2042	2034
-60.41	2337	2310	2274	2338	2308	2290	2336	2308	2290
-50.49	2593	2566	2488	2593	2568	2540	2592	2565	2539
-40.57	2834	2826	2756	2835	2825	2810	2833	2825	2809
-35.62	2928	2931	2927	2925	2929	2968	2925	2930	2925
-30.65	2950	2969	2981	2949	2967	2980	2951	2969	2980

frequency pairs (32, 16), (8, 4), (2, 1) during calibration (CAL) and transmitter off (TXOFF) frames. The 1's digit, R, tells whether the receiver re-synced at the beginning of the record. The MODE array is the first array on the tape.

The TEMP array is self-explanatory. It gives the temperature in degrees Fahrenheit. It is the second array on the tape.

The TXOFF data fill 6 tape records (records 3-8). These were recorded during times when the transmitter was turned off, but the receiver was active. Figure 8 shows the contents of the TXOFF records.

Figure 8

TXOFF ARRAY - TRANSMITTER OFF DATA
6 RECORDS - 386 WORDS/RECORD

	MODE	DIGIT A	⇒	FREQ
X	}	1		32
Y		2		8
Z		3		2
X	}	1		16
Y		2		4
Z		3		1

The TXOFF arrays were fully calibrated (converted to dB)

in the same way as the science VCO data. This procedure required that the TXOFF data be present in core as the interpolation was being done for each frequency - antenna combination (the alternative was to generate the dB vs. VCO tables twice). The TXOFF array was therefore read into core preceding the science VCO data, calibrated along with the science VCO data back to its position following the TEMP array.

The CAL array consists of three kinds of data:

1) receiver front-end noise measurement, 2) noise-diode source amplified by 20 dB, and 3) noise diode source unamplified. It contains 6 records, as diagrammed in Figure 9. These are the 9th-14th records on tape. Because these data are intended for use in calibrating the VCO - vs. - input power characteristics of the receiver, CAL data were left as VCO frequencies.

Figure 9

CAL ARRAY - CALIBRATION DATA

6 RECORDS

386 WORDS/RECORD

		MODE DIGIT A	⇒	FREQ	
G	_____	}	{	32	
NA	_____			1	8
N	_____			2	2
G	_____	}	{	16	
NA	_____			1	4
N	_____			2	1

G = Input grounded (front-end noise)
 NA = Noise diode amplified 20 dB
 N = Noise diode unamplified

Drum file DB-2 represented the full set of final science data. It was copied verbatim to the second file of the distribution tapes SEPDO7 - D10.

For use in scientific interpretation, the turn at EP4, which was "removed" from the navigation data, had to be specially handled for scientific data as well. The treatment which was applied was: the turn was identified by the stops preceeding and following the turn. The last values which existed prior to the execution of the turn, during the stop, were repeated through the time when the turn was completed. This gives the appearance for plotting, that the turn was not made. This function was performed by the STRAIGHTEN routine. The output was stored in drum file DB-2-STR.

The output from the navigation data processing, drum file ARROW-RANGES, was merged with DB-2-STR and stored in drum file TRAVERSE by MERGER.

The routine BCDTAPE was used in two versions (which differed only in the number of records they processed) to convert the binary data in TRAVERSE and DB-2 into BCD mode for tape transmission. Title arrays were placed at the beginning of each output file, named TRAVERSE-BCD and DB-2-BCD, respectively. These files were copied to tapes SEPDO7-D10 for transmission.

The detailed format of transmission tapes SEPDO7-D10 is given in Figure 10.

Figure 10

Distribution Tape Format

Tapes SEPDO7, SEPDO8, SEPDO9, SEPDO10

7-Track

Even Parity

800 BPI

BCD

Unlabeled

Fixed Unblocked Records

Record Size = Block Size = 386 - 6 char words
= 2316 chars.

Two Files

First File: Straightened Science & Nav. Data.

Second File: Unstraightened Science Data only.

FILE 1

RECORD FORMAT CONTENTS

1 27(14A6),8A6 TITLE: 27 card images plus padding.

2 386I6 MODE: (MAR) 1.

M = 1 Data acquisition mode
M = 2 Sync acquisition mode
A = 1 X antenna, 32 or 16 MHz
A = 2 Y antenna, 8 or 4 MHz
A = 3 Z antenna, 2 or 1 MHz
R = 0 No receiver resync
R = 1 Receiver resync

3	386F6.1	TEMP	Temperature, degrees F.
4	386F6.1	TXOFF	X antenna, 32, 8, or 2 MHz
5	"	"	Y antenna, " " " MHz
6	"	"	Z antenna, " " " MHz
7	"	"	X antenna, 16, 4, or 1 MHz
8	"	"	Y antenna, " " " MHz
9	"	"	Z antenna, " " " MHz

Determined by "A" digit of mode word

10	"	CAL	Front-end noise, 32, 8 or 2 MHz
11	"	"	Diode + 20 DB, " " "
12	"	"	Diode, " " "
13	"	"	Front-end noise, 16, 4, or 1 MHz

14	386F6.1	CAL	Diode + 20 DB, 16, 4, or 1 MHz
15	"	"	Diode, " " "
16	"	RANGE ₁	Range array for 1 MHz data, meters
17	"	1MHz Endfire	X antenna power in dBm
18	"	"	Y " "
19	"	"	Z " "
20	"	Broadside	X " "
21	"	"	Y " "
22	"	"	Z " "
23	"	RANGE ₂	Range array for 2 MHz data
24	"	2 MHz Endfire	X
25	"	"	Y
26	"	"	Z
27	"	Broadside	X
28	"	"	Y
29	"	"	Z
30	2(386F6.1/)	RANGE ₄	
31			
32	"	4 MHz Endfire	X
33			

34	2 (386F6.1/)	4 MHz Endfire Y
35		
.		
.		
.		
44	4 (386F6.1/)	RANGE ₈
45		
46		
47		
48	"	8 MHz Endfire X
.		
51		
.		
.		
72	8 (386F6.1/)	RANGE ₁₆
.		
79		
80	"	16 MHz Endfire X
.		
87		
.		

128

13(386F6.1/)

RANGE 32

.

.

140

141

" 32 MHz Endfire X

.

.

153

.

.

206

" 32 MHz Broadside Z

.

218

End-of-file

FILE 2

Structured exactly like File 1 except range arrays are absent.

RECORD	CONTENTS	RECORD	CONTENTS
1	TITLE	112	32 MHZ
2	MODE	.	"
3	TEMP	189	
4	TXOFF		
.	"		
10	CAL		
.	"		
16	1 MHZ		
.	"		
22	2 MHZ		
.	"		
28	4 MHZ		
.	"		
40	8 MHZ		
.	"		
64	16 MHZ		
.	"		

APPENDIX

Programs for processing science (VCO) data and merging it with Nav Data.

FREQ

U346R0*Y*11R.FREQ

```

1  FUNCTION FREQ(X,IND)
2  INTEGER VCOPER,REFPER,SIG,STERR,NPER
3  FREQ)=ABS(Y-VCOPER)
4  REAL X(2)
5  IND=0
6  SIG=10000*FID(24,4,Y(2))
7  VCOPER=SIG+1000*FID(2,4,Y(1))+
8  .      100*FID(8,4,X(1))+10*FID(14,4,X(1))+FID(22,4,X(2))
9  REFPER=1000*FID(20,4,X(1))+100*FID(26,4,X(1))+
10 .      10*FID(32,4,X(1))+FID(20,4,X(2))
11  REFPER=REFPER+SIG
12  STERR=SQRT(REFPER)
13  IF(STERR=ERR(REFPER+10000))11,1,2
14  1 IF(STERR=ERR(REFPER-10000))13,3,4
15  2 REFPER=REFPER+10000

16  GO TO 3
17  4 REFPER=REFPER-10000
18  3 NPER=100*FID(2,4,X(2))+10*FID(8,4,X(2))+FID(14,4,X(2))
19  IF(NPER),10,
20  IF(REFPER),10,
21  IF(VCOPER),10,
22  FREQ=FLOAT(NPER)/REFPER
23  FREQ=5213.*4./VCOPER/FREQ
24  IF(ABS(FREQ-1450.)<.1).OR.(1350.)GO TO 5
25  IND=1
26  FREQ=MAX(.30,MIN(.30,1.,FREQ))
27  5 IF(ABS(FREQ-5213.*4.)<.01).OR.(100.*F-6) IND=IND+2
28  IF(ERR(FREQ)<.1*.210) IND=IND+4
29  RETURN
30  10 IND=0
31  FREQ=0.
32  RETURN
33
34
35  END

```

MODEF

U346RD*Y*118.MODEF

```
1      FUNCTION MODEF(X)
2      INTEGER X,XANT,YANT,RESYNC,MODE,ANT
3
4      C      GET BITS FROM END OF WORD
5      MODE=2-FLD(35,1,Y)
6      XANT=1-FLD(34,1,X)
7      YANT=1-FLD(33,1,Y)
8      RESYNC=FLD(29,1,Y)
9
10     C      CHECK ERROR CONDITIONS
11     IF(XANT.NE.0.AND.YANT.NE.0)GO TO 1
12
13     C      CORRECT THE MODE INDICATOR IF RESYNC WAS RECEIVED
14     IF(RESYNC.EQ.1)MODE=1
15     C      GENERATE ANTENNA DIGIT
16     ANT=3-2*XANT-YANT
17
18     C      STACK DIGITS INTO MODEF
19     MODEF=10*MODE+10*ANT+RESYNC
20
21     RETURN
22
23     C      ERRORS DETECTED IN DATA
24     I WRITE(6,60)MODE,XANT,YANT,RESYNC
25     600 FORMAT(' MODE DIGIT ERROR. RAW DATA = ',6I1)
26     MODEF=0
27     RETURN
28     END
```


CHANGE

U3A6RDY*11B.CHANGE

```

1      C      CHANGE
2      C      CONVERT CARD TO FREQUENCY AND STATUS
3      REAL RECORD(413),DATA(189)
4      INTEGER STATUS(189)
5      EQUIVALENCE (DATA(189),MODE)
6
7      C      SKIP BAD RECORDS
8      1 CALL RDAL1(4,RECORD,413,42,47,42)
9      GO TO 1
10
11
12      C      PROCESS 387 RECORDS
13      2 DO 100 IREC=1,387
14
15      C      READ A RECORD
16      CALL RDAL1(4,RECORD,413,511,519,513)
17      GO TO 12
18
19      C      END OF FILE = UNEXPECTED
20      11 WRITE(6,600)IREC
21      600 FORMAT(' AT RECORD',I4)
22      STOP
23
24      C      ERROR RECORD = SET FREQUENCIES TO ZERO, STATUS TO 14
25      10 WRITE(6,600)IREC
26      DO 13 IWORD=1,189
27      DATA(IWORD)=0.
28      13 STATUS(IWORD)=14
29      C      GO WRITE RECORD
30      GO TO 120
31
32      C      CONVERT GOOD RECORD
33      12 DO 14 IWORD=1,188
34      14 DATA(IWORD)=FREQ(RECORD(2*IWORD+1),STATUS(IWORD))
35      MODE=MODE+1(RECORD(302))
36      STATUS(189)=0
37
38      C      WRITE DATA
39      100 WRITE(7)DATA,STATUS
40
41      C      PROCESSING COMPLETED
42
43      ENDFILE 7
44      WRITE(6,601)
45      601 FORMAT(' 387 RECORDS PROCESSED')
46      * END OF FILE WRITTEN ON UNIT 7*
47      STOP
48      END

```

MERGE4

1/3

1366RDWY*118,MERGE4

```

1      C      MERGE4
2      C      MERGES FINAL FOUR SCIENCE TAPES
3      REAL DATA(4,189),OUTPUT(188)
4      INTEGER STATUS(4,189),COUNTS(17)/17*07,MODE(4),
5      .      DIST(41,27)/1107*07,RECORD,TAPE,UNIT
6      EQUIVALENCE (DATA(1,189),MODE(1))
7
8      C      FOR EACH INPUT RECORD
9      DO 1 RECORD=1,307
10
11     C      READ EACH TAPE
12     DO 2 TAPE=1,4
13     UNIT=6+TAPE
14     2 READ(UNIT)(DATA(TAPE,WORD),WORD=1,109),(STATUS(TAPE,WORD),
15     .      WORD=1,189)
16
17     C      CHECK MODES FOR AGREEMENT
18     DO 3 TAPE=2,4
19
20     IF(MODE(TAPE).EQ.MODE(1))GO TO 3
21     IF(STATUS(TAPE,1).EQ.16)GO TO 2
22     C      MODE ERROR
23     WRITE(6,600)RECORD,MODE(1),TAPE,MODE(TAPE)
24     600 FORMAT(' MODES DON'T AGREE AT RECORD ',I4/
25     .      ' * MODE 1 = ',I4,' * MODE ',I2,' = ',I4)
26     STOP
27     3 CONTINUE
28
29     C      FOR EACH WORD IN A RECORD, PROCESS THE DATA
30     DO 4 WORD=1,188
31     4 CALL PROCES(DATA(1,WORD),STATUS(1,WORD),OUTPUT(WORD))
32
33     C      OUTPUT THE RESULTS
34     1 WRITE(4)OUTPUT,MODE(1)
35
36     C      FINISH PROCESSING
37     ENDDO 4
38     WRITE(6,601)
39     601 FORMAT(' END OF FILE WRITTEN ON UNIT 12/
40     .      ' * 287 RECORDS PROCESSED. ')
41
42     C      PRINT STATUS STATISTICS
43     WRITE(6,602)(I,COUNTS(I+1),I=0,16)
44     602 FORMAT('STATUS COUNTS'/15,I9)
45
46     C      SAVE ERROR DISTRIBUTIONS FOR PLOTTING
47     WRITE(3)DIST
48     ENDDO 3
49
50     C-----

```

```

51
52
53     SUBROUTINE PROCES(DATA,STATUS,OUTPUT)
54     C     SELECTS AND PROCESSES DATA
55     REAL DATA(4),STACK(4)
56     INTEGER COUNT,PSTAT,PIVOT,SECONE,COUNT1,PIVOT1,
57     .     STATUS(4),UNIT(4)
58
59     C     COUNT VALUES AT MINIMUM ERROR LEVEL AND PLACE THEM
60     C     IN STACK.
61     COUNT=1
62     C     MOVE FIRST VALUE INTO STACK, SET PSTAT
63     STACK(1)=DATA(1)
64     PSTAT=STATUS(1)
65     UNIT(1)=0
66
67     C     FOR OTHER VALUES
68     DO 1 IVAL=2,4
69
70     C     COMPARE NEW STATUS TO PSTAT
71     IF(STATUS(IVAL)-PSTAT)2,3,1
72
73     C     NEW ERROR LEVEL IS LOWER - CLEAR THE LIST
74     2 COUNT=0
75
76     C     NEW ERROR LEVEL IS SAME
77     C     INCREMENT COUNTER AND ADD TO LIST
78     3 COUNT=COUNT+1
79     STACK(COUNT)=DATA(IVAL)
80     PSTAT=STATUS(IVAL)
81
82     UNIT(COUNT)=IVAL-1
83
84     C     PROCEED FOR REMAINING VALUES
85     1 CONTINUE
86
87     C     KEEP STATUS STATISTICS
88     COUNTS(PSTAT+1)=COUNTS(PSTAT+1)+1
89
90     C     WAS SINGLE VALUE BEST
91     IF(COUNT.NE.1)GO TO 4
92
93     C     YES - USE SINGLE VALUE
94     OUTPUT=STACK(1)
95
96     C     IF ZERO STATUS, RETURN
97     IF(PSTAT.EQ.0)RETURN
98
99     C     NON-ZERO STATUS MESSAGE
100    WRITE (6,600)RECORD,WORD,PSTAT,UNIT(1),OUTPUT
101    600 FORMAT(' RECORD',I4,' WORD',I4,' STATUS',I3,
102    .     ' 2(' I4E',I2,' FREQ',F4.0))
103    RETURN

```

```

103
104 C      MULTIPLE VALUES OF SAME STATUS
105 C      SET MINIMUM DIFFERENCE TO MAX, THEN SEEK ACTUAL MIN
106 C      4 DIFMIN=2700.
107
108 C      FOR EACH PIVOT VALUE
109 C      COUNT=COUNT-1
110 C      DO 5 PIVOT=1,COUNT
111 C      PIVOT1=PIVOT+1
112 C      FOR EACH SECOND VALUE ABOVE PIVOT
113 C      DO 6 SECOND=PIVOT1,COUNT
114
115 C      IS DISCREPANCY LARGED FOR THIS PAIR OF VALUES
116 C      DIF=STACK(PIVOT)-STACK(SECOND)
117 C      IF(ABS(DIF).GE.DIFMIN)GO TO 6
118
119 C      SMALLER - MAKE SUBSTITUTIONS, USE AVERAGE
120 C      DIFMIN=ABS(DIF)
121 C      DIF=0IF
122 C      FREQA=STACK(PIVOT)
123 C      ITAPEA=UNIT(PIVOT)
124 C      FREQB=STACK(SECOND)
125 C      ITAPEB=UNIT(SECOND)
126 C      OUTPUT=(FREQA+FREQB)/2.
127
128 C      CONTINUE FOR MORE PIVOTS AND SECONDS
129 C      5 CONTINUE
130 C      6 CONTINUE
131
132 C      KEEP DIFFERENCE STATISTICS
133 C      IREFQ=(OUTPUT-300.)/100.+1.
134 C      IREFQ=MIN(27,MAX(1,IREFQ))
135 C      IDIFE=0IF/5.+21.5
136 C      IDIFE=MIN(41,MAX(1,IDIFE))
137 C      DIST(IDIFE,IREFQ)=DIST(IDIFE,IREFQ)+1
138
139 C      PRINT MESSAGE FOR LARGE DIFFERENCES OR NON-ZERO STATUS
140 C      IF(DIFMIN.LT.54. .AND.PSTAT.EQ.0)RETURN
141 C      WRITE(6,600)RECORD,WORD,PSTAT,ITAPEA,FREQA,ITAPEB,FREQB
142
143
144
145 C      RETURN
146
147 C      END

```

PRPL

```

1*      C      PRPL
2*      C      PLOTS ACCURACY STATISTICS ON PRINTER-PLOTTER
3*      C      PARAMETER LWIDE=101
4*      C      INTEGER DMAX,TOT,BLANK/' '/,BAR/'I'/,STAR/'*'/,
5*      C      .      DIST(41,27),LINE(LWIDE),FORM1(10),FORM2(10)
6*
7*      C      SET UP FORMATS
8*      C      LWIDER=LWIDE
9*      C      ENCODE(400,FORM1)LWIDER
10*     400 FORMAT('(5X,'I3,'(1H-))')
11*     C      ENCODE(401,FORM2)LWIDER
12*     401 FORMAT('(I5,'I3,'A1,F5.2)')
13*
14*     C      READ ERROR DISTRIBUTION
15*     READ(4)DIST
16*
17*     C      LOOP THROUGH PLOTS
18*     DO 1 J=1,27
19*     JFREQ=300+100*(J-1)
20*
21*     C      SET UP LINE BUFFER
22*     LINE(1)=BAR
23*     DO 2 I=2,LWIDE
24*     2 LINE(I)=BLANK
25*     LINE(LWIDE)=BAR
26*
27*     C      LOCATE MAXIMUM VALUE AND TOTAL WEIGHT
28*     DMAX=DIST(1,J)
29*     TOT=DMAX
30*     DO 3 I=2,41
31*     TOT=TOT+DIST(I,J)
32*     3 DMAX=MAX(DMAX,DIST(I,J))
33*     DDMAX=DMAX
34*     IF(TOT)1,1,
35*     ATOT=TOT/100.

```

```

35*      C
37*      C      PRINT TOP LINE
38*      WRITE(6,600)JFREQ
39*      600 FORMAT(*FREQUENCY INTERVAL (ICOHZ.) STARTS AT',I5,'HZ.')
```

40* C PRINT HEADING

41* WRITE(6,FORM1)

42*

43* C PRINT LINES

44* DO 4 I=1,41

45* IFREQ=5*(I-21)

46* IPOC=1.5+DIST(I,J)*(LWIDE-1)/DDMAX

47* IPOC=MIN(LWIDE,MAX(1,IPOC))

48* IF(IPOS.EQ.1)GO TO 9

49* DO 8 II=2,IPOS

50* 8 LINE(II)=STAR

51* 9 X=DIST(I,J)/ATOT

52* WRITE(6,FORM2)IFREQ,LINE,X

53* DO 10 II=2,IPOS

54* 10 LINE(II)=BLANK

55* LINE(LWIDE)=BAR

56* 4 CONTINUE

57*

58* C WRITE BOTTOM LINE

59* WRITE(6,FORM1)

60* C

61* C COMPUTE MEAN ABSOLUTE ERROR

62* IERR=0

63* TOT=0

64* DO 20 I=2,41

65* K=DIST(I,J)

66* TOT=TOT+K

67* 20 IERR=IERR+K*ABS(I-21)

68* ERR=(5.*IERR)/TOT

69* WRITE(6,601)ERR

70* 601 FORMAT(* AVERAGE ABSOLUTE ERROR =',F5.1,'HZ.')

71*

72* 1 CONTINUE

73* END

66RD*1*1R.DEMUX

DEMUX

```

1      C      DEMUX
2      C      DEMUX INDEXES MODE, VCO DATA FROM MIRC64 OUTPUT
3      INTEGER ST/15/,SC/14/,S1/31/,S2/28/,S4/21/,23/,
4      .      S8/4/3,11,19,27/,S16/6/1,5,9,13,17,21,25,29/,
5      .      S32/13/2,1,6,9,10,12,14,18,20,22,24,26,30/,
6      .      JUMP
7      REAL RECORD(189),REC(6,1),MODE(397),TEMP(387),
8      .      T(50,6),C(50,6),D1(50,6),D2(50,6),D4(2,50,2),
9      .      D8(4,50,6),D16(8,50,6),D32(13,10,2)
10     EQUIVALENCE (RECORD,REC)
11
12     DEFINE FILE 3(1500,50,0,1,POINT)
13
14     C      CLEAR CORE AND DRUM ARRAY COUNTERS
15     I CORE=0
16     I DRUM=0
17
18     C      FOR EACH INPUT RECORD
19     DO I ITRC=1,387
20
21     C      READ THE RECORD
22     READ(4)RECORD
23
24     C      COUNT THE CORE ARRAY
25     I CORE=ICORE+1
26
27
28     C      MOVE THE DATA INTO CORE ARRAYS
29     MODE(ITRC)=RECORD(189)
30     TEMP(ITRC)=RECORD(188)
31     CALL MOVER(T,ST,1)
32     CALL MOVER(C,SC,1)
33     CALL MOVER(D1,S1,1)
34     CALL MOVER(D2,S2,1)
35     CALL MOVER(D4,S4,2)
36     CALL MOVER(D8,S8,4)
37     CALL MOVER(D16,S16,8)
38     CALL MOVER(D32,S32,13)
39
40     C      IF CORE IS NOT FULL, PROCESS NEXT RECORD
41     IF (ICORE.NE.50)GO TO 1
42
43     C      CORE IS FULL -- DUMP TO DRUM
44     COUNT DUMP RECORD, RESET CORE COUNTER
45     I DRUM=IDRUM+1
46     I CORE=0
47
48     CALL DRUMMR
49
50     C      FINISH ALL DATA
51     CONTINUE
52
53     C      EMPTY FINAL ARRAYS
54     I DRUM=IDRUM+1
55     CALL DRUMMR

```

```

56      C      DUMP TO TAPE
57      WRITE(7,MODE)
58      WRITE(7,TEMP)
59      CALL TAPER
60      ENDFILE 7
61      WRITE(6,6000)
62      ADD FORMAT(* END OF FILE WRITTEN ON UNIT 7*)
63      .      * END OF DEMULTIPLEXING.*
64      STOP
65
66
67
68      SUBROUTINE MOVEP(DATA,SOURCE,SIZE)
69      INTEGER SIZE,SOURCE(SIZE)
70      REAL DATA(SIZE,8,6)
71      DO 1 I=1,SIZE
72      I=SOURCE(I)
73      DO 1 ICOMP=1,6
74      1 DATA(I,ICOMP,ICOMP)=REC(ICOMP,I)
75      RETURN
76
77
78      SUBROUTINE DRUMWR
79      WRITE(6,600)IDRUM
80      ADD FORMAT(* MOVING RECORDS TO DRUM, IDRUM=*,I2)
81      JUMP=0
82      CALL DRUM(T,1)
83      CALL DRUM(C,1)
84      CALL DRUM(D1,1)
85      CALL DRUM(D2,1)
86      CALL DRUM(D4,2)
87      CALL DRUM(D8,4)
88      CALL DRUM(D16,8)
89
89      CALL DRUM(D32,16)
90      RETURN
91
92

```



```

93
94     SUBROUTINE DRUM(DATA,SIZE)
95     INTEGER SIZE,DEST
96     REAL DATA(50,SIZE,6)
97     DEST=JUMP+(IDRUM-1)*SIZE
98     DO 1 ICOMP=1,6
99     DO 2 J=1,SIZE
100    2 CALL TRANS(DATA(I,J,ICOMP),DEST+1)
101    1 DEST=DEST+8*SIZE
102    JUMP=JUMP+48*SIZE
103
104
105
106
107     SUBROUTINE TRANS(DATA,RECORD)
108     INTEGER RECORD,DATA(50)
109     WRITE(3*RECORD)DATA
110     FINT(3*POINT)
111     RETURN
112
113
114
115     SUBROUTINE TAPE9
116     REAL DATA(50,8),OUTPUT(387)
117     EQUIVALENCE (DATA,OUTPUT)
118     IREC=1
119     WRITE(6,600)
120     600 FORMAT(' WRITING DATA ON TAPE. ')
121     DO 1 I=1,31
122     DO 1 J=1,6
123     DO 2 K=1,8
124     CALL INPUT(DATA(I,K),IREC)
125     2 IREC=IREC+1
126     1 WRITE(7)OUTPUT
127     RETURN
128
129
130
131     SUBROUTINE INPUT(DATA,IREC)
132     REAL DATA(50)
133     READ(3*IREC)DATA
134     FINT(3*POINT)
135     RETURN
136
137
138     END

```

CALIBPRT

U366RONX*SCRIPTS,CALIBPRT

```

1      REAL DB(13,4)
2      INTEGER VCO(13,3,3,2)
3
4      C      INPUT DELEVEL DATA
5      DO 1 I=FEQ=1,6
6      READ(3,300)(DB(I,IFEQ),I=13,11)
7      300 FORMAT(7F7.2)
8      C      CHANGE SIGNS
9      DO 1 I=1,13
10     1 DB(I,IFEQ)=-DB(I,IFEQ)
11
12     C      INPUT VCO DATA
13     DO 2 ITEMP=3,1
14     DO 2 IFEQ=1,6
15     DO 2 IANT=1,3
16     2 READ(4,400)(VCO(I,IAnt,ITEMP,IFEQ),I=13,11)
17     400 FORMAT(7I5)
18
19     C      OUTPUT BY FREQ
20     DO 3 IFEQ=1,6
21     JREQ=2*(IFEQ-1)
22     WRITE(6,600)JREQ
23     600 FORMAT(777'I',12,' MHz')
24     .      TIC,'X ANTENNA',I32,'Y ANTENNA',I49,'Z ANTENNA'
25     .      ' DBM19,3(3Y,'COLO MED HOT')77)
26
27     C      OUTPUT THE DATA
28     3 WRITE(6,601)(DB(I,IFEQ),
29     .      ((VCO(I,IAnt,ITEMP,IFEQ),ITEMP=1,3),
30     .      IANT=1,3),I=1,13)
31     601 FORMAT(7F8.2,3(17,2I5))
32
33     STOP
34     END

```

REBLOCK

0366RD#Y*11B,REBLOCK

```
1      C      REBLOCK
2      C      COPIES DATA DELETING FIRST RECORDS AND SHIFTING MODE
3      C      INTEGER MODE(387),SIZE(6)/1,1,2,4,8,16/
4      C      REAL DATA(387,13)
5
6      C      COPY MODE
7      C      READ(7)MODE
8      C      CALL WRITER(MODE)
9
10     C      COPY TEMP
11     C      CALL COPY(1)
12
13     C      COPY TXOFF, CAL, 1, AND 2 MHZ DATA
14     DO 1 I=1,24
15     1 CALL COPY(1)
16
17     C      COPY HIGHER FREQUENCY DATA
18     DO 2 ICFREQ=3,6
19     1 SIZE=SIZE(ICFREQ)
20     DO 2 ICOMP=1,6
21     2 CALL COPY(1SIZE)
22
23     C      ENDFILE 8
24     C      WRITE(6,600)
25     C      ADD FORMATER END OF FILE WRITTEN ON UNIT 8*1
26     C      STOP
27
28
29     C      SUBROUTINE COPY(N)
30     DO 1 I=1,N
31     1 CALL READR(DATA(I,1))
32     CALL OUT(DATA(N+1,1),N)
33     RETURN
34
35     C      SUBROUTINE OUT(DATA,N)
36     REAL DATA(386,N)
37     DO 1 I=1,N
38     1 CALL WRITER(DATA(I,N))
39     RETURN
40
41     C      SUBROUTINE READER()
42     REAL D(387)
43     READ(7)D
44     RETURN
45
46     C      SUBROUTINE WRITER(D)
47     REAL D(386)
48     WRITE(8)D
49     RETURN
50
51     C      END
```

CALIBRATE

1/4

346R0W1*1R,CALIBRATE

```

1      C      CALIBRATE
2      C      USES DB-VS-VCO DATA TO CALIBRATE POWER LEVELS
3      C      INTEGER MODE(386),SIZE(6)/1,1,2,4,8,12/,
4      C      TXOTAP(6)/3,3,2,2,1,1/,TXOFF(7)/1,2,1,2,1,2/,
5      C      REAL TEMP(386),TXOFF(386,3,2),WORK(386),
6      C      TABLE(2,2701,31,DB(13,6),VCO(12,2,3,6)

7
8      C      READ CALIBRATION DATA
9      C      CALL SETUP
10
11     C      READ MODE ARRAY
12     C      READ(7)MODE
13     C      COPY MODE ARRAY
14     C      WRITE(8)MODE
15
16     C      READ TEMP ARRAY
17     C      READ(7)TEMP
18     C      CONVERT TO TEMPERATURE
19     C      DO 1 I=1,386
20     C      1 TEMP(I)=.973*TEMP(I)-47
21     C      WRITE TEMP ARRAY
22     C      WRITE(8)TEMP
23
24     C      READ TXOFF ARRAYS
25     C      DO 2 ITRANS=1,2
26     C      DO 2 ICOMP=1,2
27     C      2 CALL READR(TXOFF(I,ICOMP,ITRANS))
28
29     C      COPY CAL ARRAYS
30     C      DO 3 ICOMP=1,6
31     C      READ(7)WORK
32     C      3 WRITE(8)WORK
33
34     C      FOR EACH FREQUENCY
35     C      DO 10 IREFQ=1,6
36     C      FREQ=2***(IREFQ-1)+.1*MOD(IREFQ-1,2)
37     C      WRITE(6,6)IREFQ
38     C      600 FORMAT(* START PROCESSING FOR *,F4.1,*,MHZ DATA.*/)
39     C      ISIZE=SIZE(IREFQ)
40
41     C      COMPUTE INTERPOLATION TABLE
42     C      DO 11 ICOMP=1,3
43     C      11 CALL INTERP(DB(1,IREFQ),VCO(1,ICOMP,2,IREFQ),
44     C      .      VCO(1,ICOMP,3,IREFQ),TABLE(1,1,ICOMP))
45
46     C      INTERPOLATE THE SCIENCE DATA
47     C      DO 12 ITRANS=1,2
48     C      DO 12 ICOMP=1,3
49     C      CALL REEPTABLE(I,1,ICOMP)
50     C      DO 12 IREFQ=1,ISIZE
51     C      READ(7)WORK
52     C      DO 13 I=1,386
53     C      13 WORK(I)=POW(WORK(I),TEMP(I))
54     C      12 WRITE(8)WORK

```

```

25
56 C INTERPOLATE THE TXOFF DATA
57 DO 14 I=1,316
58 M=MODE(I)
59 MA=MODE(I,1000)/10
60 C MA IS THE MODE ANTENNA DIGIT
61 C IF ANTENNA DIGIT INDICATES OTHER FREQUENCIES, SKIP THIS PT
62 C IF(MA.NE.TXOTAB(I,REF))GO TO 14
63
64 C FREQUENCY ACREFC = GET CORRESPONDING XY FRAME DIGIT
65 MTX=TXOTAC(I,REF)
66 C INTERPOLATE
67 DO 15 ICOMP=1,3
68 CALL REFER(TABLC(I,1,ICOMP))

69 15 TXOFF(I,ICOMP,MTX)=DOW(TXOFF(I,ICOMP,MTX),TEMP(I))
70 14 CONTINUE
71
72 C FINISH FOR ALL FREQUENCIES
73
74 10 CONTINUE
75
76
77 C WRITE TXOFF DATA
78 DO 16 ITRANS=1,2
79 DO 16 ICOMP=1,3
80 16 CALL WRITER(TXOFF(I,ICOMP,ITRANS))
81
82 WRITE(6,591)
83 A01 FORMAT('END OF FILE WRITTEN ON UNIT 6//
84 * * * END OF CALIBRATION PROCESSING * *
85 ENDFILE 8
86 STOP
87
88
89
90 SUBROUTINE SETUP
91 C READS CALIBRATION DATA ARRAYS
92 READ(3,200)((DB(I,IFREQ),I=13,1),IFREQ=1,6)
93 200 FORMAT(7F7.2)
94 C CHANGE SIGNS
95 DO 1 IFREQ=1,6
96 DO 1 I=1,13
97 I DB(I,IFREQ)=-DB(I,IFREQ)
98
99 C READ VCO DATA
100 DO 2 ITEMP=1,1
101 DO 2 IFREQ=1,6
102 DO 2 IANT=1,3
103 2 READ(4,400)(VCO(I,IAnt,ITEMP,IFREQ),I=13,1)
104 400 FORMAT(7F5.0)
105
106 RETURN
107

```

```

153
154
155     SUBROUTINE INTERPOL(VCO,POWER)
156     C     INTERPOLATES FOR SINGLE-TEMPERATURE ARRAY
157     REAL POWER(2,2701),DR(13),VCO(13),X
158     INTEGER I,J,NEXT
159     DEFINE P(I)=POWER(I,I)
160
161     C     FILL THE ARRAY UP TO THE FIRST VCO VALUE
162     J=1
163     IF(VCO(1).LT.300.)GO TO 1
164     J=VCO(1)-299.
165     DO 2 I=1,J
166     2 P(I)=DR(I)
167     J=J+1
168
169     C     INTERPOLATE UP TO THE LAST VCO VALUE
170     1 DO 3 NEXT=2,13
171     DRL=DR(NEXT-1)
172     VCOL=VCO(NEXT-1)
173     VCON=VCO(NEXT)
174     X=(DR(NEXT)-DRL)/(VCON-VCOL)
175     DO 4 I=1,2701
176     IF(I+299..GT.VCON)GO TO 3
177     4 P(I)=(I+299-VCOL)*X+DRL
178     RETURN
179     3 J=I
180
181     C     FILL THE ARRAY BEYOND THE LAST VCO VALUE
182     DRL=DR(13)
183     DO 5 I=1,2701
184     5 P(I)=DR(13)
185     RETURN
186
187
188
189     END

```

```

108
109
110      SUBROUTINE READC(X)
111      REAL X(284)
112      READ(7)X
113      RETURN
114
115
116
117      SUBROUTINE WRITC(X)
118      REAL X(284)
119      WRITE(8)X
120      RETURN
121
122
123
124      C      FUNCTION FOR TEMPERATURE INTERPOLATION
125      FUNCTION POW(TEMP,TEMPD)
126      REAL POWER(2,2701)
127      INTEGER I
128      IF(TEMP.EQ.0.)GO TO 1
129      T=TEMP-298.5
130      T=MIN(2701,MAX(1,T))
131
132      POW=POWER(1,1)+(TEMP-65.)/40.*(POWER(2,1)-POWER(1,1))
133      RETURN
134      C      POWER.
135      RETURN
136
137
138      ENTRY INTER(OR,VCOMED,VCONOT,POWER)
139      C      INTERPOLATES FOR ONE FREQUENCY-ANTENNA COMBINATION
140
141      C      INTERPOLATE FOR 75 AND 105- DEGREE ARRAYS
142      CALL INTER(OR,VCOMED,POWER)
143      CALL INTER(OR,VCONOT,POWER(2,1))
144      RETURN
145
146
147
148
149      C      ENTRY IS REFER TO PREVIOUS INTERPOLATIONS
150      ENTRY REFER(POWER)
151      RETURN
152
153

```

ARRANGE

```

1      C      ARRANGE
2      C      MOVE TXOFF DATA BACK INTO POSITION AT START OF PAGE
3      REAL BOOK(396)
4      INTEGER SIZE(6)/1,1,2,4,8,13,1,1 MHZ*,
5      .      *2 MHZ*,*4 MHZ*,*8 MHZ*,*16 MHZ*,*32 MHZ*/
6
7      C      COPY NONE, TEMP
8      CALL COPY(1,*MODE*)
9      CALL COPY(1,*TEMP*)
10
11     C      SKIP CAL
12     CALL SKIP(6,*CAL*)
13
14     C      SKIP SCIENCE DATA
15     DO 1 IERFQ=1,6
16         1 CALL SKIP(6*SIZE(IEFQ),TITLE(IEFQ))
17
18     C      COPY TXOFF
19     CALL COPY(6,*TXOFF*)
20
21     C      START OVER
22     REWIND 7
23     WRITE(6,600)
24     600 FORMAT(' REWIND UNIT 7')
25
26     C      SKIP NONE AND TEMP
27     CALL SKIP(1,*MODE*)
28     CALL SKIP(1,*TEMP*)
29
30     C      COPY CAL
31     CALL COPY(6,*CAL*)
32
33     C      COPY SCIENCE DATA
34     DO 2 IERFQ=1,6
35         2 CALL COPY(6*SIZE(IEFQ),TITLE(IEFQ))
36
37     ENDDO IERFQ
38     WRITE(6,601)
39     601 FORMAT(' END OF FILE WRITTEN ON UNIT 04')
40     .      * END PROCESSING*
41
42
43     SUBROUTINE SKIP(N,TITLE)
44     WRITE(6,600)N,TITLE
45     600 FORMAT(' SKIPPING ',I3,' RECORDS OF ',I2,' DATA.')
46     DO 1 J=1,N
47         1 READ(7)WORK
48     RETURN
49
50
51     SUBROUTINE COPY(N,TITLE)
52     WRITE(6,600)N,TITLE
53     600 FORMAT(' COPYING ',I3,' RECORDS OF ',I2,' DATA.')
54     DO 1 J=1,N
55         READ(7)WORK
56         1 WRITE(8)WORK
57     RETURN
58
59     END

```


STRAIGHTEN

1/3

036680 *X*SERIES, STRAIGHTEN

```

1      C      STRAIGHTEN
2      C      PROPAGATES SCIENCE DATA THROUGH THE LOG-TIME
3      REAL DATA(100,13), DESCRIP(19)
4      INTEGER N, NREC, IRECI /396/RODPR(67)/1,1,2,9,9,127
5
6
7      C      READ AND ECHO IN
8      CALL READATT(3, DESCRIP, 19, 91, 91, 91)
9      GO TO 2
10
11     1 STOP
12     2 RECODE(1000, DESCRIP)
13     1000 FORMAT(10RX, I6)
14     *WRITE(A, 600) (DESCRIP(I), I=1, 18), N
15     AND FORMAT(11R0X, A77, 511, A77
16     , ' * DIRECTION', A77, FORWARD OR REVERSE = ', A77
17     , ' * IY, 19A77, IRECI =', I6)
18
19     C      COPY TO, CHANGE IRECI
20     ENCODE(1001, DATA) (OCORI(11), I=1, 13), I=101
21     1001 FORMAT(18A6, I6)
22     *WRITE(A, 611) (DATA(I, 1), I=1, 17)
23     AND FORMAT(1( ' *', A77), ' * NCR TO =',
24     , ' *', A77), IY, 19A6/A77/3( ' *', A77)
25     CALL *WRITE(2, DATA, 19, 91, 91)
26
27
28     C      COPY MORE
29     CALL COPY(1)
30
31     C      SKIP MAX DATA
32     CALL SKIP(3)
33
34     C      COPY TEMP
35     CALL COPY(1)
36
37     C      SKIP TIME
38     CALL SKIP(1)
39
40     C      COPY TYPE AND CAL
41     CALL COPY(12)
42
43     C      FOR EACH FREQUENCY

```

STRAIGHTEN

```

41
42      DO 3 IF EQ=1,6
43      NREC=RECREC(IEREC)
44
45      C      SKIP NAV
46      CALL SKIP(NREC)
47
48      C      FOR EACH COMPONENT
49      DO 3 ICOMP=1,6
50      C      CONSTRUCT, CORRECT, OUTPUT THE DATA
51      CALL COP(DATA)
52
53      3 CONTINUE
54
55      C      WRITE END OF FILE
56      CALL FILEND(2)
57
58      STOP
59
60
61
62      SUBROUTINE SKIP(NREC)
63      WRITE(6,60)NREC
64      60 FORMAT(' SKIPPING ',I3,' RECORDS')
65      DO 2 I=1,NREC
66      2 CALL RWAIT(3,DATA,I,$1,$1,$1)
67      RETURN
68      1 WRITE(6,60)I
69      60 FORMAT(' SKIPPING AT RECORD ',I5)
70      STOP
71
72
73      SUBROUTINE COPY(NREC)
74      WRITE(6,60)NREC
75      60 FORMAT(' COPYING ',I2,' RECORDS')
76      DO 2 I=1,NREC
77      CALL RWAIT(3,DATA,I,$1,$1,$1)
78      2 CALL RWAIT(2,DATA,I,REFC,$1,$1)
79      RETURN
80      1 WRITE(6,60)I
81      60 FORMAT(' COPYING AT RECORD ',I5)
82      STOP
83

```

```

84
85     SUBROUTINE COR(DATA)
86     C     CORRECTS ONE COMPONENT
87     REAL DATA(N,NREC)
88     WRITE(6,60)NREC
89     60) FORMAT(' CORRECTING',I3,' RECORDS')
90
91     C     INPUT THE ARRAY
92     DO 1 I=1,NREC
93     1) CALL RD'AIT(3,DATA(I,1),I,92,99,99)
94
95     C     CORRECT IT
96     CALL CORREC(DATA)
97
98     C     OUTPUT IT
99     CALL OUTPUT(DATA)
100
101     RETURN
102
103     C     ERROR
104     9) WRITE(6,600)I
105     600) FORMAT(' AT RECORD',I5)
106     STOP
107
108
109     SUBROUTINE OUTPUT(DATA)
110
111     REAL DATA(I,REC),NREC)
112     DO 1 I=1,NREC
113     1) CALL WR'AIT(2,DATA(I,1),I,REC,99,99)
114     RETURN
115     9) WRITE(6,600)I
116     600) FORMAT(' AT RECORD',I5)
117     STOP
118
119
120     SUBROUTINE CORREC(DATA)
121     REAL DATA(NREC,3)
122     DO 1 I=5,65
123     DO 1 J=1,NREC
124     1) DATA(I,1)=DATA(I,53)
125     RETURN
126
127
128     END

```

MERGER

```

1      C      MERGER
2      C      MERGE RANGE ARRAYS FROM RANGER WITH DATA FROM STRAIGHTEN
3      C      INPUT FROM ARROW-RANGES, SERD01
4      C      OUTPUT TO SERD03
5      C      REAL RANGES(386,30),DATA(386),LABEL(10)
6      C      INTEGER DIMS(6)/1,1,2,4,8,13/ROIMS(6)/1,2,3,5,9,17/
7
8      C      READ(2) RANGES
9      C      CALL WINDER(3)
10     C      CALL WINDER(4)
11
12     C      SKIP OLD LABEL
13     C      CALL RWRITE(3,LABEL,1,59,59,59)
14     C      GO TO 8
15     C      9 WRITE(6,609)
16     C      609 FORMAT(' AT LABEL ')
17     C      8 CONTINUE
18     C      SETUP NEW LABEL
19     C      ENCODE(1000,LABEL)
20     C      1000 FORMAT('1.1 AD10178EST FORKRD',
21     C      ' *SEP SITE TO 4.3 KM, 2ND DIGZ**N, EDH TURN REMOVED',
22     C      ' *, IRV NAV REST RANGES*,T109,* *386*')
23
24     C      OUTPUT NEW LABEL
25     C      CALL RWRITE(4,LABEL,19,519,519)
26
27     C      COPY MODE, TEMP, IXOFF, CAL
28     C      DO 1 I=1,14
29     C      CALL RWRITE(3,DATA,386,510,510,510)
30     C      1 CALL RWRITE(4,DATA,386,510,510)
31
32     C      FOR EACH FREQUENCY
33     C      DO 2 IREQ=1,6
34     C      IDIM=DIMS(IREQ)
35     C      JDIM=ROIMS(IREQ)
36
37     C      COPY RANGE
38     C      CALL OUT(RANGES(1,JDIM),IDIM)
39
40     C      FOR EACH COMPONENT
41     C      DO 3 ICOMP=1,6
42     C      COPY DATA
43     C      DO 3 I=1,IDIM
44     C      CALL RWRITE(3,DATA,386,511,511,511)
45     C      3 CALL RWRITE(4,DATA,386,511,511)
46     C      2 CONTINUE
47
48     C      CALL F1END(4)
49
50     C      STOP

```

MERGER 2/2

```
51      10 WRITE(6,600)I
52      600 FORMAT(' AT RECORD',I4)
53      STOP 10
54      12 STOP 12

55      11 WRITE(6,610)IFREQ,ICOMP,I
56      610 FORMAT(' AT FREQ',I4,' AT COMPONENT',I4/
57      .      ' AT RECORD',I4)
58      STOP 11
59
60
61      SUBROUTINE OUT(I,I)
62      REAL R(2*6,I)
63      DO J=1,I
64      1 CALL WR'ALL(I,R(I,J),386,59,59)
65      RETURN
66      2 STOP
67
68
69      END
```

BCDTAPES

COBOL
04

```
C      BCDTAPES
C      CONVERTS FILES WITH 386-WORD RECORDS TO BCD
      DIMENSION INPUT(386,2),OUTPUT(386,2),TITLE(386)

C
C      SET OUTPUT TAPE PARITY
      CALL PMPAR(3,0)

C
C      READ AND COPY TITLE
      READ(2,200)TITLE
200  FORMAT(13A6,A2)
      CALL WRWAIT(8,TITLE,386,$9,$9)

C
C      READ AND COPY MODE
      CALL RDWAIT(7,INPUT,386,$9,$9,$9)
      ENCODE(100,OUTPUT)(INPUT(I,1),I=1,386)
100  FORMAT(22I6)
      CALL WRWAIT(8,OUTPUT,386,$9,$9)

C
C      READ, CONVERT, AND COPY EVERYTHING ELSE
      IBUF=1
      CALL PREAD(7,INPUT,386,$9,$9,$9)
      DO 1 IREC=1,216
      IBUF=3-IBUF
      CALL PREAD(7,INPUT(1,IBUF),386,$10,$10,$9)
      CALL CONVRT(INPUT(1,3-IBUF),OUTPUT(1,3-IBUF))
      CALL PWRITE(8,OUTPUT(1,3-IBUF),386,$9,$9)
1    CONTINUE
      CALL PMPWAIT(8,$9,$9)
      CALL FILEND(8)
      STOP
10   WRITE(6,600)
600  FORMAT(' UNEXPECTED EOF  REACHED ON UNIT 7')
9    STOP

C
C      SUBROUTINE FOR REAL CONVERSIONS
      SUBROUTINE CONVRT(INPUT,OUTPUT)
      REAL INPUT(386),OUTPUT(386)
      ENCODE(100,OUTPUT)INPUT
100  FORMAT(22F6.1)
      RETURN
```

END

COBOL:44

COBOL:44

COBOL:44

TAPECOPY

```
C      TAPECOPY
C      COPIES FINAL DATA TAPES
      REAL DATA(386,2)
      CALL PMPAR(9,0)
      CALL TRANS(7,218)
      CALL TRANS(8,189)
      STOP

C
C
C      SUBROUTINE TRANS(IUNIT,NREC)
      CALL PREAD(IUNIT,DATA,386,$9,$9,$9)
      IBUF=1
      DO 1 IREC=1,NREC
      CALL PREAD(IUNIT,DATA(1,3-IBUF),386,$9,$9,$9)
      CALL MRWAIT(9,DATA(1,IBUF),386,$9,$9)
1  IBUF=3-IBUF
      CALL FILEND(9)
      CALL PRWAIT(IUNIT,$2,$2,$2)
2  RETURN
9  STOP
      END
SPAN:23
      23
```

A 4(i)

Apollo 17 SEP
Data Processing

John C. Rylaarsdam
July 1974

Contents

Introduction	2
The Stack	10
Documentation Routines	
LUNALIST, LUNALST2, LUNALST3	15
Editing Routines	
LUNACOPY, LUNACPY2, LUNACPY3	17
LUNACPY4	17
LUNACPY5	18
Plotting Routines	
LUNAPLOT, LUNAPLT2	20
LUNAPLT3	20
LUNAPLT4	21
LUNAPLT5	21
ANTENNAO	22
Statistical Routines	
CALSTAT	24
TXOSTAT	25
VLBIRT	26
Auxillary Routines	
LUNIN, LUNIN2, LUNIN3	28
SEPLOT	29
INTPOL	30
FILTER	30
PLINIT	31
File Formats	
SCI1, SCI2, SCI3	32
SCI1A, SCI2A, SCI3A	36
STAT1	37
EP4	38
NAV1	39

Program Listings	40
ANTENNA0	41
ANTPAT	43
BUBBLE	46
CALSTAT	47
FILTER	49
GAPLOT	50
INTPOL	52
LUNACOPY	53
LUNACPY2	57
LUNACPY3	60
LUNACPY4	64
LUNACPY5	68
LUNALIST	72
LUNALST2	74
LUNALST3	77
LUNAPLOT	80
LUNAPLT2	83
LUNAPLT3	87
LUNAPLT4	93
LUNAPLT5	97
LUNIN	104
LUNIN2	106
LUNIN3	109
ODCINT	111
PLINIT	113
RTPLOT	113
SEPLOT	115
STOPT	120
TXOSTAT	121
TXPLOT	125
VLBI RT	126
SCI2B Plots	130

Figures

1	Processing Flow	5
2	Examples of Stack Use	11
3	Algorithm for Assembly of Arrays of Range and dB Information	14

Tables

1	Data Set Summary	3
2	Incorrectly Labelled Blocks	16
3	Receiver Timing Sequence	19
4	Calibration Data Obtained From Earth-based Tests	25
5	Format of SCI1 Label Record	32
6	Interpretation of Mode Data	33
7	Record Contents on Files SCI1, SCI2, and SCI3	34
8	dB Data on Files SCI1A, SCI2A, and SCI3A	36
9(a)	Contents of File STAT1	37
9(b)	Function of the Index K for Transmitter-off and Calibration Arrays on File STAT1	38
10	Record Format for File EP4	39

Introduction

This report is a summary of intermediate stage processing operations performed on the data from the Apollo 17 surface electrical properties experiment. The starting points for these operations are the files designated SCI1, SCI2, and SCI3; the contents of these and all generated files, plots, and listings are summarized in table 1. File SCI1 is a preliminary release of the data; files SCI2 and SCI3 are the final data sets, respectively with and without data for the EP-4 turn, prepared by R. Watts, tape number SEPD09. (N.B. - subsequent references to removal of turn data refer to work described in this report, rather than to Watts' initial processing.)

The diagrams in figure 1 give an indication of the sequences of processing operations. More detailed information is provided by the descriptions of the programs. Annotated listings are also included, as well as precise tabulations of the formats of the various files. The program listings include descriptions of all required card input data.

In addition, two complete sets of plots (SCI2B) are included as a record of the data; in one set dB values are plotted versus the range in metres, and in the second set versus the range in wavelengths.

L M T T C R D
 A O E X A A B
 B D M - L N
 E E P O G
 L F E
 F

Notes

CAL1 Listing	* * *		
EP4		*	dB data for 490 m. <= range <= 535 m.
EP4 Listing		*	
EP4 Plot		*	
EP4A Listing		*	dB data for 490 m. <= range <= 535 m.
EP4A Plot		*	and LRV in motion
NAV1			times and odometer counts relative to beginning of traverse
RT1 Listing		*	includes VLBI data, converted
RT1 Plot		*	to ranges
SCI1	* * * * * *		
SCI1 Listing	* * * * * *		
SCI1A	*	*	dB data sampled at intervals of 0.1
SCI1A Listing	*	*	wavelength
SCI1A Plot		*	

Table 1 - Data set summary.

L M T T C R D
 A O E X A A B
 B D M - L N
 E E P O G
 L F E
 F

Notes

SCI2	* * * * *		no data for EP-4
SCI2 Listing	* * * * *		
SCI2 Plot		* *	
SCI2A	*	*	no data for EP-4;
SCI2A Listing	*	*	dB data sampled at
SCI2A Plot		*	intervals of 0.1
SCI2B Plot		* * *	wavelength
SCI3	* * * * *		
SCI3 Listing	* * * * *		range data from SCI2
SCI3A	*	*	dB data sampled at
SCI3A Listing	*	*	intervals of 0.1
SCI3A Plot		*	wavelength
STAT1	* * * *		also contains speed data
TX01 Listing		* *	all data except 490 m.
TX01 Plot		* *	<= range <= 535 m.
			and LRV in motion

Table 1 - Data set summary (continued).

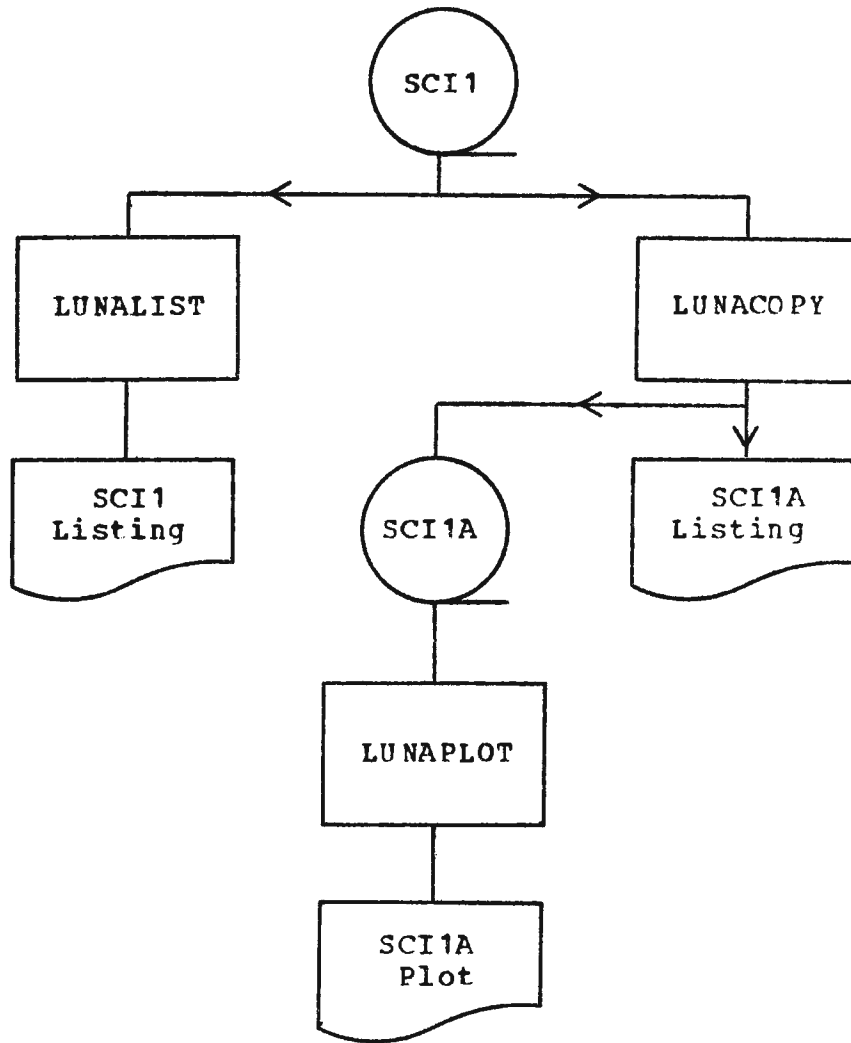


Figure 1(a) - Processing flow.

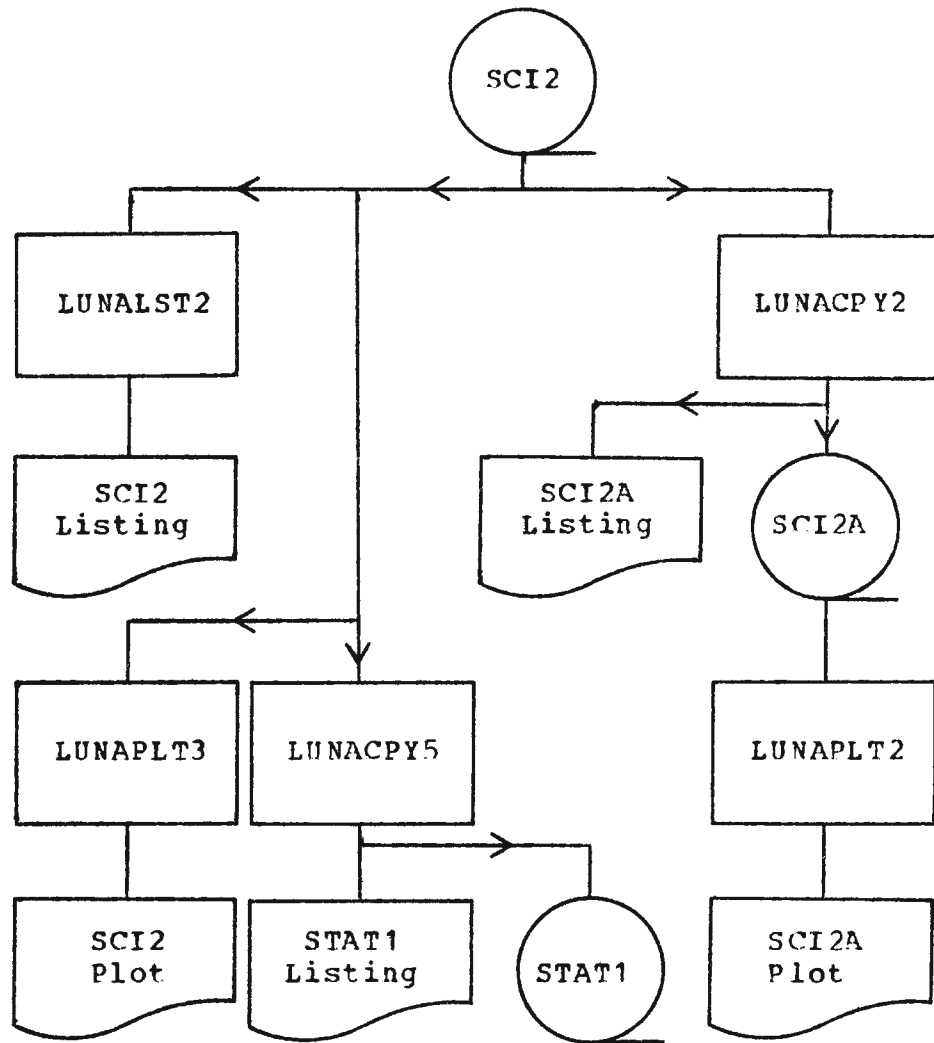


Figure 1(b) - Processing flow.

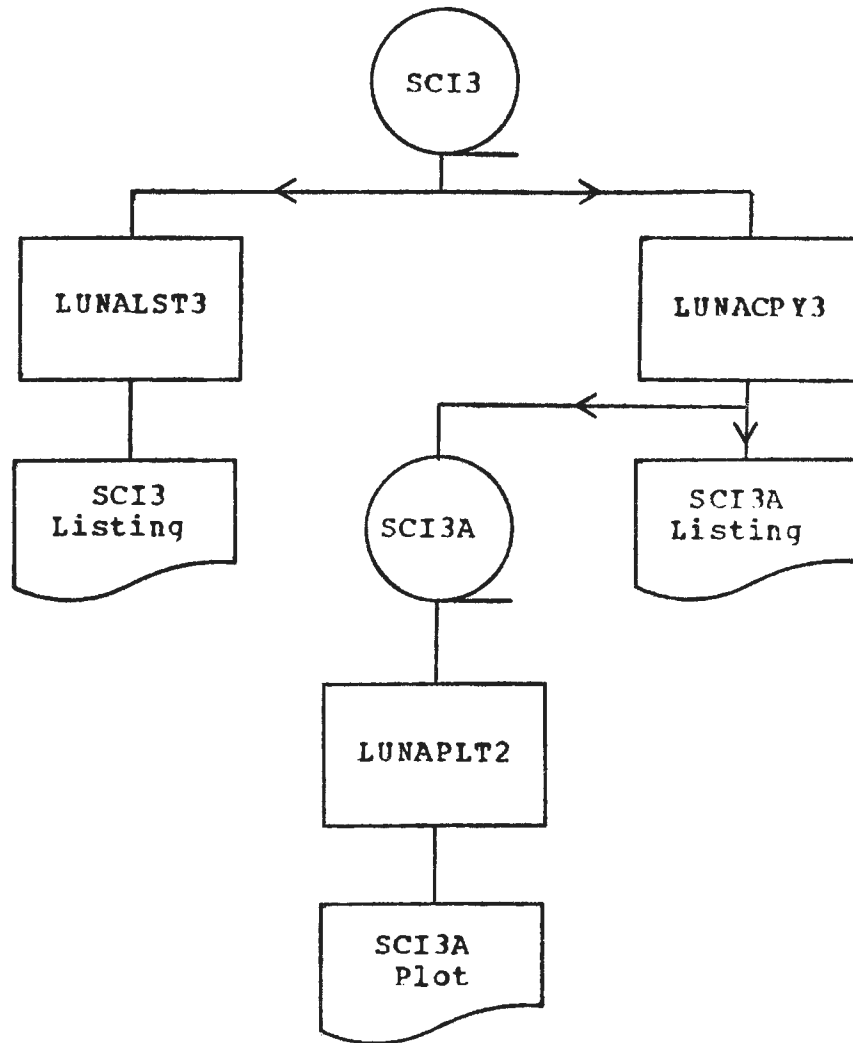


Figure 1(c) - Processing flow.

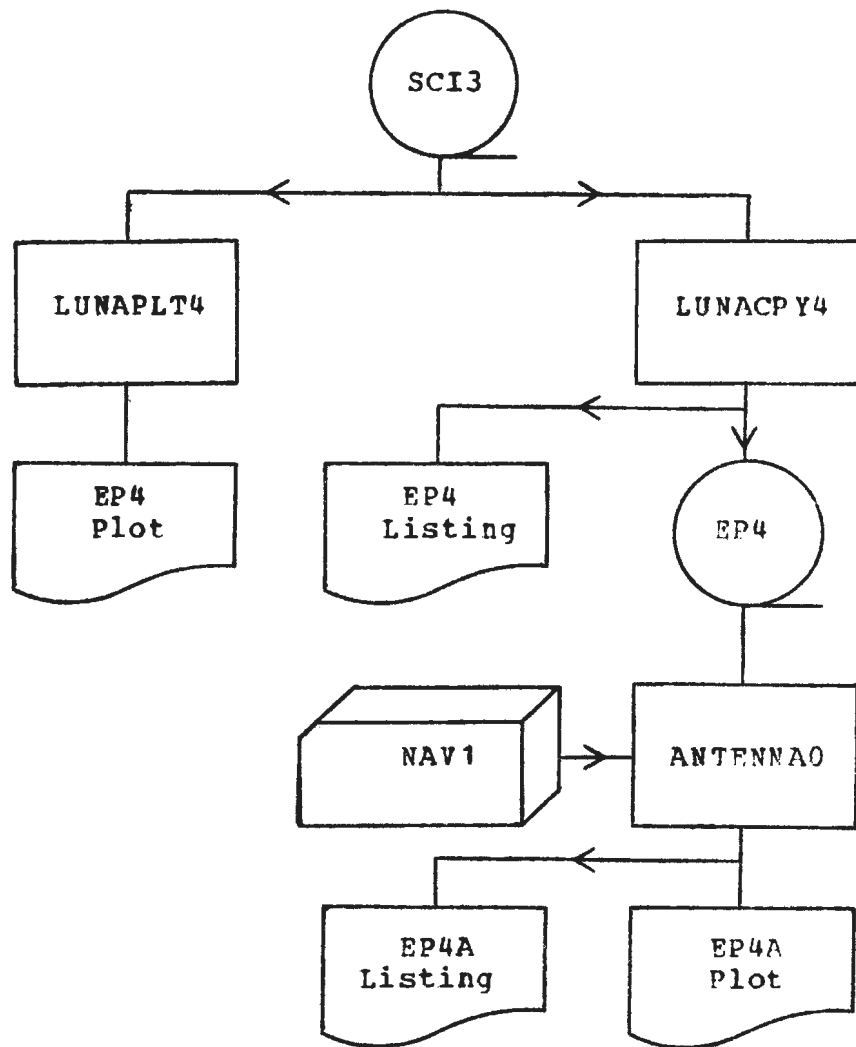


Figure 1(d) - Processing flow.

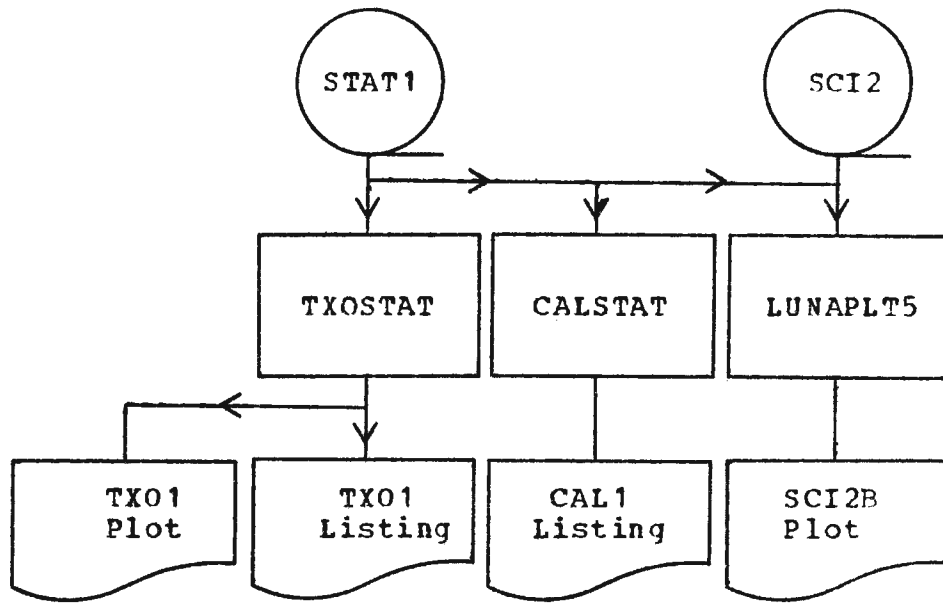


Figure 1(e) - Processing flow.

The Stack

A number of routines use a large array (named DATA, hereafter referred to by name, or as "the stack") and a set of indices to the array, as the basis of a system for manipulating range and dB data.

In most cases a particular set of range or dB information is contained in several blocks, which it is generally convenient to combine into one large block before processing. To accomplish this, three indices are associated with the stack as diagrammed in figure 2: IXX and IXY indicate the first words of range and dB data respectively; IORG gives the location of the first word into which data should be read to make an extension to the block currently being assembled. Other parameters relating to the stack may be defined where necessary.

The basic procedure to be used to process a complete file is outlined in figure 3. In this description "read a block" is taken to mean that the contents of one block from the input file are placed in locations IORG through $IORG + N - 1$ of the stack; the meaning of "last" is that given to it by the LUNIN routines.

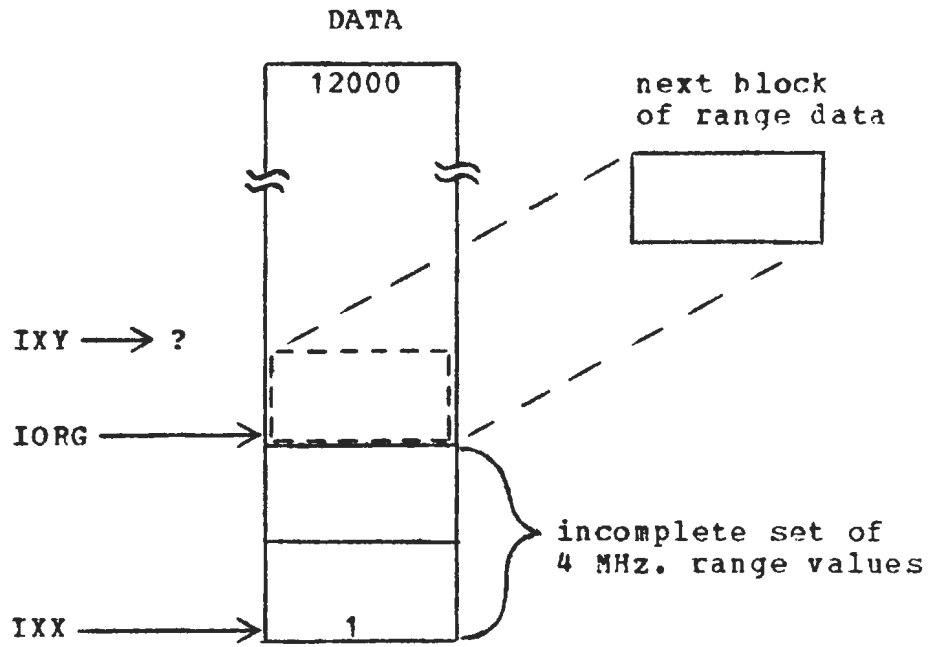


Figure 2(a) - Example of stack use:
assembling a range array.

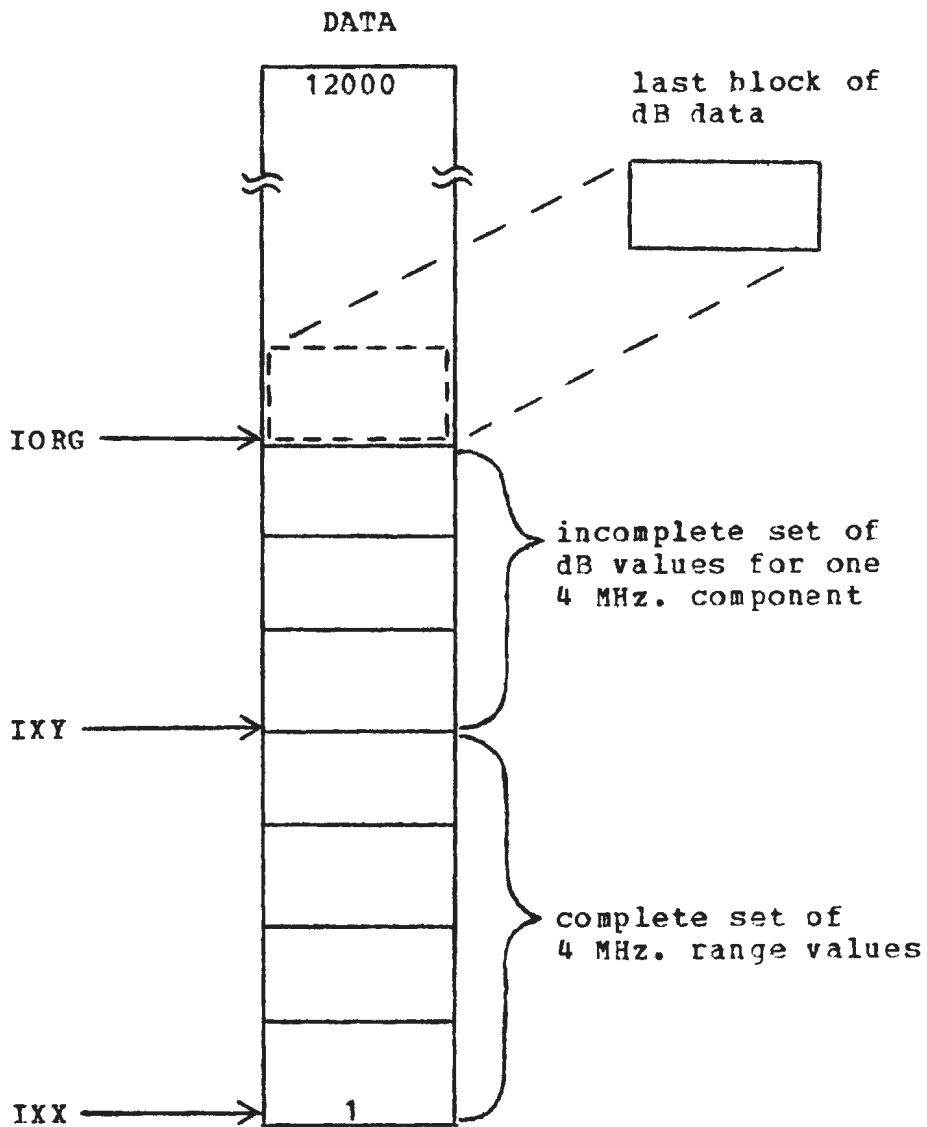


Figure 2(b) - Example of stack use: completing a dB array.

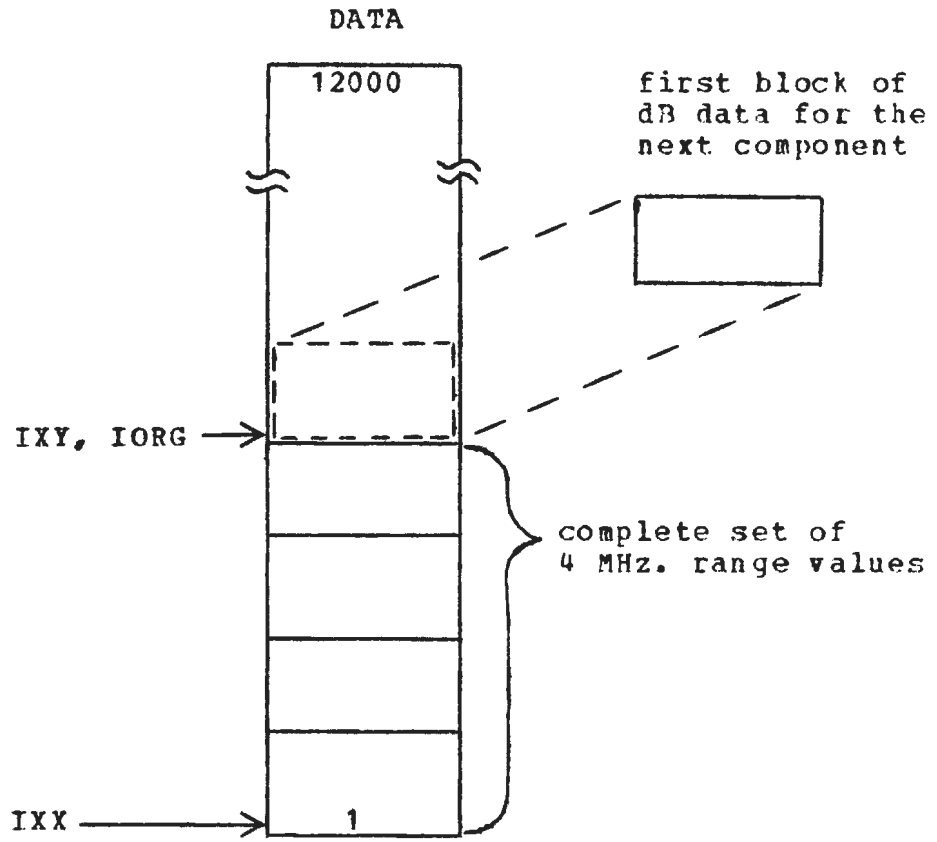


Figure 2(c) - Example of stack use: beginning the dB array for a new component.

```

do for frequency = 1, 2.1, 4, 8.1, 16, 32.1
  iorg = 1;  ixx = 1;  m = 0;

  ASSEMBLE THE RANGE ARRAY:
  repeat
    read a block;
    iorg = iorg + n;
    m = m + 1;
  until last;

  ixy = iorg;
  repeat
    l = 0;  iorg = ixy;

    ASSEMBLE THE DB ARRAY FOR ONE COMPONENT:
    repeat
      read a block;
      iorg = iorg + n;
      l = l + 1;
    until l = m;

    perform required processing;
  until last;
end;

```

Figure 3 - Algorithm for assembly of arrays of range and dB information.

Documentation Routines

LUNALIST, LUNALST2, LUNALST3

These routines produce complete listings of the data on files SCI1, SCI2, and SCI3 respectively.

The data are read from SCI1 and SCI2 by LUNIN and LUNIN2 respectively, and printed in blocks corresponding to the physical records on the files, fifteen values per line. Each block is preceded by a heading, containing the character information returned by the input routine, identifying the contents of the block.

The procedure for listing SCI3 is more complex, since the file contains no range data. LUNALST3 invokes LUNIN2 to read a record from SCI2, and inspects the returned value of ITYPE(2). If this value is five, indicating a block of range data, the block is listed. Otherwise, LUNIN3 is called to read a record from SCI3, which replaces the data from SCI2, and the new block is listed.

N.B. A bug, in all three routines, results in the identification of transmitter-off, calibration, and one megahertz range and dB data being printed incorrectly, as indicated in table 2. (The numbers in parentheses indicate how many records are affected.)

<u>contents of block</u>	<u>label printed</u>
temperature(1)	temperature
transmitter-off(1)	NONE
transmitter-off(5)	transmitter-off
calibration(1)	NONE
calibration(5)	calibration
1 MHz. range(1)	NONE
1 MHz. dB(6)	NONE

Table 2 - Incorrectly labelled blocks.

Editing Routines

LUNACOPY, LUNACPY2, LUNACPY3

These routines read files SCI1, SCI2, and SCI3 respectively, and produce the binary files SCI1A, SCI2A, and SCI3A, containing the label records from the input files, and six blocks of NPTS dB values each, for each frequency, after interpolation at intervals of 0.1 wavelength. NPTS varies with frequency, and is either the maximum number of interpolated values which could be generated, or 1000, whichever is smaller.

The stack mechanism is used to set up the arrays of range and dB values to be given to the interpolation routine, and the array RANGE is initialized with the appropriate equivalent in metres of 0, 0.1, 0.2, ... 99.9 wavelengths. Subroutine INTPOL is called to do the interpolation, and returns its results in array VCO.

After interpolation the values of NSTART and NPLOT (set by INTPOL) indicate the first and last elements of VCO which contain interpolated results. Before writing the array on the output file, the programs set VCO(1) through VCO(NSTART-1) and VCO(NPLOT+1) through VCO(1000) to zero.

LUNACPY4

This program produces a binary file (EP4) of range and dB values for the turn at EP-4, using file SCI3 as input. The program is a simple modification of LUNAPLT4, in which the call to subroutine GAPLOT is replaced by a write statement which generates a record on file EP4, and a second statement which writes the contents of the record on the printer.

LUNACPY5

This program is used to generate file STAT1, containing temperature, calibration, and transmitter-off data, and arrays of the ranges at which the data were obtained. Also, a set of crude LRV speed values, corresponding in time to the transmitter-off data, is computed and written on the output file.

The array of temperature values is simply copied from the input file (SCI2). The associated range data is the set of values for one megahertz, which is also copied directly onto the output file. (The one megahertz data were used since their occurrences are the closest in time to the temperature data - c.f. table 3.)

Transmitter-off data are read from the input file in two groups of three blocks each. For each group, the first, second, and third blocks contain data from the x, y, and z receiving antennae respectively. The frequency for each sample is dependant on the group, and on the tens digit of the corresponding element of the mode array: the first group contains data for frequencies of 32.1, 8.1, and 2.1 megahertz, and the second group for 16.0, 4.0, and 1.0 megahertz, corresponding respectively to tens digits of 1, 2, and 3. The contents of the blocks of calibration data are arranged similarly, with blocks one and four containing values for front-end noise, two and five containing values for the noise diode, and values for the noise diode plus 20 dB amplification in blocks three and six.

Using the number of the block on which it is working, and the appropriate contents of the mode array, the program generates arrays of values which may be indexed by frequency, and either antenna in the case of the transmitter-off data, or noise source in the case of the calibration data.

Approximate range values corresponding to the above data are obtained by selecting the values closest to them in the timing sequence from the 32.1 megahertz range array, and again using the contents of the mode array. Corresponding to each range value, an approximation of the LRV speed is computed by taking the difference of the immediately succeeding and preceding elements of the 32.1 megahertz range array.

<u>Record</u>	<u>Contents</u>	<u>Record</u>	<u>Contents</u>
1	16 MHz. v.c.o.	17	16 MHz. v.c.o.
2	32 MHz. v.c.o.	18	32 MHz. v.c.o.
3	8 MHz. v.c.o.	19	8 MHz. v.c.o.
4	32 MHz. v.c.o.	20	32 MHz. v.c.o.
5	16 MHz. v.c.o.	21	16 MHz. v.c.o.
6	32 MHz. v.c.o.	22	32 MHz. v.c.o.
7	4 MHz. v.c.o.	23	4 MHz. v.c.o.
8	32 MHz. v.c.o.	24	32 MHz. v.c.o.
9	16 MHz. v.c.o.	25	16 MHz. v.c.o.
10	32 MHz. v.c.o.	26	32 MHz. v.c.o.
11	8 MHz. v.c.o.	27	8 MHz. v.c.o.
12	32 MHz. v.c.o.	28	2 MHz. v.c.o.
13	16 MHz. v.c.o.	29	16 MHz. v.c.o.
14	32 MHz. v.c.o.	30	32 MHz. v.c.o.
15	transmitter-off	31	1 MHz. v.c.o.
16	calibration	32	synchronization/reset (also contains temperature data)

Table 3 - Receiver timing sequence.
Each record is 202.5 ms. duration.

Plotting Routines

LUNAPLOT, LUNAPLT2

These routines produce (CALCOMP) plots of dB versus distance, using interpolated data. LUNAPLOT is used for plotting the data in SCI1A; LUNAPLT2 may be used to plot data from either SCI2A or SCI3A.

One namelist (PREQ) control record is read for each frequency. The parameters which may be specified allow choice of components, maximum and minimum wavelengths, and maximum dB value to be plotted. Filtering of the data before plotting may be requested, a dB level may be specified for plotting a reference mark, and optional plot annotation may be supplied.

Subroutine PLINIT is called to initialize the plotting software. The actual plots are produced using the DATPLT entry point of subroutine SEPLOT.

LUNAPLT3

This program is used to generate (CALCOMP) plots of the data on SCI2, similar to the plots produced by LUNAPLOT and LUNAPLT2. The program differs from the others in that the data are not interpolated, and cannot be filtered before plotting; also, the portion of the data corresponding to the turn at EP-4 is deleted before plotting.

The program uses the stack mechanism to prepare data for the plotting routine. Removal of the data for the turn is accomplished as follows. After an entire array of range values has been assembled, the program locates the values to be deleted; then succeeding values are copied downward to maintain a contiguous set, and IORG is reset to indicate what is

then the first free location. The indices defining the gap are modified to apply to the dB segment of the stack, and each time a complete set of dB data is assembled, an equivalent compacting operation is performed.

Parameters to control the plotting operation are supplied on a set of namelist (CNTL) control records. The plotting is done using the DATPLT entry point of subroutine SEPLOTT.

LUNAPLT4

This program produces (CALCOMP or GOULD) plots of dB values for the EP-4 turn versus an implicit time scale, using data from file SCI3. The data are plotted as discrete points (marked by symbols) at equal horizontal intervals. Each component is plotted on a separate graph.

Each set of values to be plotted is assembled in the main program and passed to subroutine GAPLOT, which produces the plot. The method used to define the desired values is basically the complement of the method used in the two preceding programs to remove the same data. However, in this case the required segments of the arrays are merely located; no compacting operation is performed.

LUNAPLT5

This routine is a modification of LUNAPLT3 which allows the distances to be expressed in either metres or wavelengths. In addition, transmitter-off values from file STAT1 are plotted (as points, rather than continuous curves) as a baseline for each dB curve, using entry point BASEL of subroutine SEPLOTT.

ANTENNAO

This program is used to generate plots of the patterns of the three receiving antennae, based on the data for the turn at EP-4 contained in file EP4. Two methods are available for computing the angle between the LRV and the SEP transmitter: the plotted points may be at equal angular increments throughout the whole range, or the navigation data may be used to compute an approximate angular displacement for each point.

One namelist (CNTL) control record is required for each frequency. Parameters in each record allow choice of components to be plotted, initial angle and the difference between final and initial angles, and indices defining the data points obtained while the LRV was in motion; a Boolean value (NAVDAT) may be included to indicate whether or not navigation data are to be used in computing angles, and if this value is "true", a time must be supplied corresponding to the first data point in the set for which the LRV was moving.

The main program organizes the control information, and then enters a loop, in each cycle of which it reads one record from file EP4, and if the data in that record are to be plotted, passes the data and required control information to subroutine ANTPAT.

Subroutine ANTPAT begins by drawing a set of x and y axes and plotting a label indicating frequency and component. The total angular range is divided into equal intervals, based on the number of points to be plotted. If navigation data are to be used in computing angular displacements, the number of odometer counts at the beginning and end of the range are obtained by invocations of function ODCINT, and their difference (ODCRAN) is computed. The first dB value and the initial angle are converted to rectangular coordinates and the point is plotted. A

loop is then entered, which continues until the last data point has been plotted: the angle is decremented (to give clockwise rotation) either by the constant amount, or by using the result of an invocation of ODCINT to determine a fraction of the total angle; rectangular coordinates are computed, and the point is plotted; the index of the next value is determined, using the supplied parameters.

Function ODCINT is invoked with one argument, a time (T) in seconds. On the first entry a set of times and corresponding left- and right-wheel odometer counts are read from the card reader. For each triple the time and the average of the counts are saved. The value of the function is an odometer count obtained by linear interpolation, using T and the arrays of times and corresponding average counts.

Statistical Routines

CALSTAT

This program is used to compute various statistics for each set of calibration data on file STAT1: the means and standard deviations of the front-end noise; differences between the experimental noise diode values (with and without amplification) and values for the same data obtained from earth-based testing; the means and standard deviations of these differences.

The computations for front-end noise data are straightforward, and should require no explanation. Most of the variable names begin with "EG". The results of the calculations are written on FORTRAN logical eight, which is used as an auxiliary printer.

The calculations for the noise-diode data (with and without amplification - "with" is indicated by "PA" as part of the name of each variable involved) involve computation of an earth-based value, using linear interpolation according to temperature, of the v.c.o. frequencies given in table 4. The difference between the experimental and interpolated values is computed; the remainder of the calculation consists of the accumulation and scaling of the appropriate sums.

transmitter frequency	noise diode		noise diode + amplifier	
	66°F	112°F	66°F	112°F
1.	525.6	508.6	1167.7	1157.7
2.1	564.2	546.2	1208.	1208.
4.	593.3	566.3	1230.	1230.
8.1	694.8	682.8	1298.6	1304.6
16.	756.1	770.1	1293.7	1306.7
32.1	833.9	888.9	1199.6	1198.6

Table 4 - Calibration data obtained from earth-based tests.

TXOSTAT

This routine and its associated subroutines compute mean values and standard deviations for each set of transmitter-off data on file STAT1. Separate statistics are calculated for periods when the rover was stopped and in motion; in the latter case, values for the EP-4 turn are excluded from the computations. Each set of values is displayed twice: once in order of increasing distance from the transmitter, and once in order of increasing LRV speed. The dB values are also plotted versus LRV speed (if plots are not required, subroutine TXPLOT is simply replaced by a dummy routine).

A set of bounds, the same as the 32.1 megahertz bounds input, but adjusted relative to the beginning of the 32.1 megahertz range data rather than the beginning of the turn data, is required as input. These values are used by function STOPT in deciding whether the rover was moving or stopped for each point within the EP-4 turn.

The statistical computations, which are performed in the main program, are relatively straightforward

and should not require explanation. Data for ranges greater than 1667 metres are omitted from all calculations. A possibly confusing action is the assignment of -1 to certain elements of the speed array; the elements are those within the EP-4 turn for which the LRV was in motion. The speed values for these elements are computed (meaninglessly) as zero by LUNACPY5; the value of -1 indicates to the plotting routine that each such datum is to be ignored.

Ordering of the data according to increasing speed is accomplished by subroutine BUBBLE, which performs a bubble sort. Rather than interchanging elements within four parallel arrays (one of speeds and three of dB values) the routine uses an integer array (IX) of equivalent size, supplied by the calling program; IX(I) is initialized to I, and the contents of IX are used as indirect addresses to the actual data arrays, and it is these indices which are interchanged. When the sort is complete, the contents of IX indicate the order in which the other arrays should be indexed to obtain the data in order of increasing speeds.

The dB data are plotted versus speed by subroutine TXPLOT, which is entered once for each frequency. For each entry, three sets of labelled axes are plotted (one for each receiving antenna) within an 8.5 by 11 inch area. The appropriate data points are then simply plotted on each set of axes.

VLBI RT

This program is used to compare results from the VLBI experiment with SEP navigation data; the comparison is done on the basis of distance from the SEP transmitter.

The 16 megahertz range data are read from file SCI2, and an array of corresponding times is generated, using a starting value which is read as a

control parameter. A parameter may also be supplied specifying a value to be added to each range value.

The VLBI times are converted from hours, minutes, and seconds to seconds, and each pair of x and y coordinates is converted to a distance. An interpolated SEP range value is computed for each VLBI datum, using the time arrays; the difference between VLBI and interpolated SEP ranges is calculated, and summations are taken of the differences and their squares, which yield the mean difference and standard deviation.

If PLOT is set to true on the namelist control record, subroutine RTPLOT is called. The subroutine plots a set of time and range axes, using the supplied parameter SCALE. The SEP and VLBI ranges are then plotted versus time in two simple loops.

Auxillary Routines

LUNIN, LUNIN2, LUNIN3

These subroutines are used to read data from files SCI1, SCI2, and SCI3 respectively. Floating-point data are returned to the invoking routine in the array DATA (not to be confused with the stack, although most of the programs which invoke these routines use a portion of the stack for passing data); fixed-point data are returned in array IDATA. The label record in SCI1 contains various fields, the contents of which are passed to the invoking routine through COMMON block LUNDAT. The label records of the other files consist solely of text, which is returned in IDATA.

The values in the fixed-point array TIDX are used to identify each record read from the input file. The values are arranged in seventeen groups of three: the first value in each group indicates the number of records of that type which are on the file; the second and third values are used to select alphanumeric identification from arrays TYPE1 and TYPE2 respectively. The alphanumeric identification is returned in TYPE, and the second and third values from TIDX are returned in ITYPE (both TYPE and ITYPE are in LUNDAT).

LUNIN obtains the value of N (the number of values in each record other than the label record) from the label record; the other two routines expect N to have been set by the invoking routine.

The logical variables FIRST and LAST are set to true if the record read is the first or last of its type respectively (e. g. - the first of the twenty-four eight megahertz dB records).

SEPLOT (DATPLT, BASEL)

This routine (written by J. J. Proctor, 1973) has been modified in a number of ways:

- (1) dB values may be plotted versus either wavelengths or metres; the decision is made by examining XSCALE: if it is less than ten it is assumed to be the number of inches per twenty-wavelength segment, while a value of ten or greater is assumed to indicate the number of inches per kilometre.
- (2) Entry point BASEL has been added in order to allow a set of points to be plotted in conjunction with each curve, using the same (relative) plotter origin. This is for the purpose of indicating a set of background values for each curve.
- (3) Low dB values are no longer set to zero. Furthermore, the y-origin for each curve is now equivalent to (relative) zero dB, rather than the integral minimum dB value. This was necessary in order to keep the plot of background values on the page.
- (4) All the labels for individual curves, except the component identification, have been eliminated.
- (5) A reference mark for each curve is plotted on the y-axis, at a dB level set by the invoking routine.

N.B. The entry point (THEPLT) and associated code for plotting theoretical curves have not been changed. However, since the program was modified, no attempt has been made to verify the integrity of this feature.

Most of the code for the subroutine is concerned with setting up the axes and labels, and with placing

a particular curve on the graph. The range axis markings depend on the value of XSCALE, as described above, and extend from zero to the first multiple of the chosen increment greater than the highest range value in the data. A displacement is computed such that curves on the graph will be equally spaced vertically. All these operations are performed on entry at DATPLT, the entry point for plotting data curves, if a new graph (not just a new curve) is to be plotted.

The curves themselves are plotted in a straightforward manner, and a label is plotted at the right-hand end of each, to provide component identification. After a component has been plotted, the parameters defining the position of the curve are left unchanged until the next entry at DATPLT; therefore an entry at BASEL will result in the baseline points being plotted on the same set of relative axes as the associated data curve. (If BASEL is invoked at any other time, it will not function properly.)

INTPOL

This is a general linear interpolation routine. It accepts "input" arrays XIN and YIN, and an array XOUT, of values on the same scale as XIN, for which interpolated values for YIN are required. The results are placed in array YOUT, and parameters NSTART and NPLOT are set to indicate which values in YOUT are the result of successful interpolation, and which are undefined due to the corresponding elements of XOUT being out of the range of XIN.

FILTER

This is a subroutine which accepts array A, and applies to its contents the filter whose coefficients are contained in array F. Array B is used for

accumulation of sums, and its contents are copied into array A before return to the calling program.

PLINIT

This subroutine is used to systematize the initialization of the University of Toronto CALCOMP software package. Presumably it will be of interest only to users of that installation.

File Formats

SCI1, SCI2, SCI3

Each of these files begins with a label record. The format of this record on file SCI1 is given in table 5. The label records for the other two files are 2316 characters, consisting of 27 segments of 84 characters each, followed by 48 characters of padding. Each 84-character segment contains one card image and four padding characters.

The next record in each file is the mode array, in format 386I6. Each element is a three (decimal) digit number, MAR; the significance of the values of the digits is indicated in table 6. (The notations "f1" and "f2" in the table refer to transmitter-off and calibration data descriptions given in table 7.)

The format of all remaining records is 386F6.1; the contents of these records are given in table 7. (N.B. - file SCI3 contains no range data.)

<u>Format</u>	<u>Contents</u>
A6	run identification
A6	site identification
A6	traverse direction
A6	forward/reverse traverse
14A6	title
I6	number of values in each succeeding block

Table 5 - Format of SCI1 label record.

<u>Digit</u>	<u>Value</u>	<u>Significance</u>
M	1	receiver in data acquisition mode
	2	receiver in synchronization acquisition mode
A	1	f1 = 32.1 MHz.; f2 = 16 MHz.
	2	f1 = 8.1 MHz.; f2 = 4 MHz.
	3	f1 = 2.1 MHz.; f2 = 1 MHz.
R	0	synchronization not received
	1	synchronization received

Table 6 - Interpretation of mode data.

<u>Number</u>	<u>Rec.</u>	<u>Tx.</u>	<u>Freq.</u>	<u>Contents</u>
1				label
1				mode
1				temperature
1	x		f1	transmitter-off
1	y		f1	transmitter-off
1	z		f1	transmitter-off
1	x		f2	transmitter-off
1	y		f2	transmitter-off
1	z		f2	transmitter-off
1			f1	calibration - grounded input
1			f1	calibration - amplified noise
1			f1	calibration - noise
1			f2	calibration - grounded input
1			f2	calibration - amplified noise
1			f2	calibration - noise
1			1	range *
1	x	EW	1	dB
1	y	FW	1	dB
1	z	EW	1	dB
1	x	NS	1	dB
1	y	NS	1	dB
1	z	NS	1	dB
1			2.1	range *
1	x	EW	2.1	dB
1	y	EW	2.1	dB
1	z	EW	2.1	dB
1	x	NS	2.1	dB
1	y	NS	2.1	dB
1	z	NS	2.1	dB

Table 7 - Record contents on files SCI1, SCI2, and SCI3.

* not present on SCI3

<u>Number</u>	<u>Rec.</u>	<u>Tx.</u>	<u>Freq.</u>	<u>Contents</u>
2			4	range *
2	x	EW	4	dB
2	y	EW	4	dB
2	z	EW	4	dB
2	x	NS	4	dB
2	y	NS	4	dB
2	z	NS	4	dB
4			8.1	range *
4	x	EW	8.1	dB
4	y	EW	8.1	dB
4	z	EW	8.1	dB
4	x	NS	8.1	dB
4	y	NS	8.1	dB
4	z	NS	8.1	dB
8			16	range *
8	x	EW	16	dB
8	y	EW	16	dB
8	z	EW	16	dB
8	x	NS	16	dB
8	y	NS	16	dB
8	z	NS	16	dB
13			32.1	range *
13	x	EW	32.1	dB
13	y	EW	32.1	dB
13	z	EW	32.1	dB
13	x	NS	32.1	dB
13	y	NS	32.1	dB
13	z	NS	32.1	dB

Table 7 - Record contents on files SCI1, SCI2, and SCI3 (continued).
 * not present on SCI3

SCI1A, SCI2A, SCI3A

These files are written in binary form (i.e. - without format control), and contain dB data equivalent to that in files SCI1, SCI2, and SCI3 respectively. Each file begins with the same label information as its parent file.

The remainder of each file consists of thirty-six blocks of dB data; each block is of the form given in table 8. The dB data are interpolated values, at intervals of 0.1 wavelength, beginning at zero wavelengths; the maximum value of NPTS is one thousand.

<u>Position</u>	<u>Contents</u>
1	eight characters: frequency
2	floating-point: frequency
3	integer: NSTART
4	integer: NPTS
5 through NSTART+3	floating-point: 0.0
NSTART+4 through NPTS+4	floating-point: dB values

Table 8 - dB data on files SCI1A, SCI2A, and SCI3A.

STAT1

The six blocks which comprise this binary file are summarized in table 9(a). The index I for the first two blocks selects temperatures and corresponding ranges at successive intervals of 6.48 seconds.

For the remaining blocks, the index I selects the various data at successive times (at varying intervals). Values of one through six for J select data for frequencies 1.0 through 32.1 megahertz respectively. The significance of K is indicated in table 9(b).

<u>Block</u>	<u>Contents</u>	<u>Indexing</u>
1	TEMP(I)	I<=386
2	RANGE(I)	I<=386
3	CAL(I, J, K), NCAL(J, K)	I<=NCAL(J, K) J<=6 K<=3
4	TXOFF(I, J, K), NTXOFF(J, K)	I<=NTXOFF(J, K) J<=6 K<=3
5	RANGE2(I, J), NR(J')	I<=NR(J') J<=6 J'=1+(J-1)/2
6	SPEED(I, J)	I<=NR(J') J<=6

Table 9(a) - Contents of file STAT1.

K	TXOFF	CAL
1	x antenna	grounded input
2	y antenna	noise diode +20dB amplification
3	z antenna	noise diode

Table 9(b) - Function of the index K for transmitter-off and calibration arrays on file STAT1.

EP4

This file contains range and dB data from file SCI3, only for the region of the turn at FP-4 (specifically for the range values on the interval 490 to 535 metres, inclusive. There are 36 records on the file, all of the same form, but of varying length. The form of a record is summarized in table 10.

<u>Name</u>	<u>Words</u>	<u>Contents</u>
F	1	frequency in megahertz
NCOMP	1	component identification: 1: rho endfire 2: phi endfire 3: zed endfire 4: rho broadside 5: phi broadside 6: zed broadside
YMIN	1	minimum and
YMAX	1	maximum dB values
N	1	number of range-dB pairs
RANGE	N	range data
DB	N	dB data

Table 10 - Record format for file EP4.

NAV1

This is actually a part of the card input data to ANTENNA0. Any number of cards may be included. Each card contains three values: a time in seconds, and corresponding right-front- and left-rear-wheel odometer counts, in format (3F10.).

Program Listings


```

C
REAL*4 X(600), Y(600), TIME(600)
REAL*4 TZERO(6)
REAL*4 DT(6) / 6.48, 6.48, 3.24, 1.62, .91, .49848 /
REAL*8 PROGNM(2) / 'OQGP.ANT', 'ENNA' /
INTEGER*2 BOUND(6, 6), BOUNDS(6)
LOGICAL*1 DECIDE(6, 6) / 36 * .TRUE. /, NAVDAT / .FALSE. /
INTEGER*2 COMP(6) / 212, 222, 232, 211, 221, 231 /
INTEGER*2 ICOMP(6)

C
NAMELIST / CNTL / IFREQ, ICOMP, AQ, ARANGE, TIMEO, BOUNDS, NAVDAT
C
C SET UP CONTROL INFORMATION
C
AQ = 3.14159
ARANGE = 6.28318
DO 100 I=1,6

C
DO 10 J=1,6
    ICOMP(J) = 0
    BOUNDS(J) = 0
10 CONTINUE

C
C GET FREQUENCY INDICATOR AND COMPONENTS TO PLOT
C
READ(5, CNTL)
TZERO(IFREQ + 1) = TIMEO
DO 50 J = 1,6

C
    IF A COMPONENT IS NOT TO BE PLOTTED,
    RESET ITS MATRIX ENTRY.
C
    IC = COMP(J)
    DO 30 K = 1,6
        IF(IC .EQ. ICOMP(K))
            GO TO 50
30 CONTINUE
    DECIDE(IFREQ + 1, J) = .FALSE.
50 CONTINUE

C
DO 80 J = 1,6
    BOUND(J, IFREQ + 1) = BOUNDS(J)
80 CONTINUE
100 CONTINUE

C
C INITIALIZE THE PLOTTER
C
CALL PLINIT(PROGNM)

```

```

CALL PLOT( 4.25, 5.0, -3)
C
C LOOP THROUGH FREQUENCIES
C
DO 140 IFFREQ = 1,6
C
C SET UP THE TIME ARRAY
C
IF1 = BOUND(1, IFFREQ)
IF6 = BOUND(6, IFFREQ)
DO 110 I = IF1, IF6
    TIME(I) = TZPRC(IFFREQ) + DT(IFFREQ) * (I - IF1)
110 CONTINUE
C
C LOOP THROUGH COMPONENTS
C
DO 130 JCOMP = 1,6
    READ(1) FREQ, NCOMP, YMIN, YMAX, N,
    .      (X(I), I = 1, N), (Y(I), I = 1, N)
    .      IF(.NOT. DECIDE(IFFREQ, JCOMP))
    .      GO TO 130
    CALL ANTPAT(TIME, Y, N, FREQ, JCOMP, BOUND(1, IFFREQ),
    .          AO, ARANGE, NAVDAT)
130 CONTINUE
140 CONTINUE
C
CALL PLOTND
C
RETURN
END

```

```

*****
*
* ANTPAT
*
*****

```

```

SUBROUTINE ANTPAT(T, H, N, F, JC, B, AO, ARANGE, NAVDAT)
C
C ROUTINE TO PLOT THE ANTENNA PATTERN
C THROUGH THE TURN AT FP-4
C
C PARAMETERS ARE:
C
C T - TIME ARRAY
C

```

```

C      P - V.C.O. ARRAY
C
C      N - NUMBER OF POINTS IN P OR H
C
C      F - FREQUENCY
C
C      JC - COMPONENT IDENTIFIER:
C
C              ENDFIRE  BROADSIDE
C
C              RHO      1          4
C              PHI      2          5
C              ZED      3          6
C
C      P - BOUNDS WITHIN H; THE VALUES H(B(1)) - H(B(2)) INCLUSIVE
C              H(B(3)) - H(B(4)) INCLUSIVE
C              AND H(B(5)) - H(B(6)) INCLUSIVE ARE PLOTTED.
C
C      AO - INITIAL ANGLE - DEFAULTS TO PI
C
C      ARANGE - RANGE OF ANGLES - DEFAULTS TO 2*PI
C
C
C      REAL*8 LAB(6) / 'RHO END.', 'PHI END.', 'ZED END.',
C              'RHO BRD.', 'PHI BRD.', 'ZED BRD'
C      REAL*4 H(N), T(N)
C      INTEGER*4 COMP(6,2) / Z3B, Z24, Z69, Z3B, Z24, Z69,
C              3 * 'END', 3 * 'BRD'
C      INTEGER*2 B(6)
C      LOGICAL*1 NAVDAT
C
C      WRITE(6,1000) F, LAB(JC), N, P
C
C      PLOT X AND Y AXES FOR REFERENCE
C
C      CALL PLOT(-4., 0., 3)
C      CALL PLOT( 4., 0., 2)
C      CALL PLOT( 0., 4., 3)
C      CALL PLOT( 0., -4., 2)
C
C      PLOT LABELS: FREQUENCY AND COMPONENT
C
C      CALL NUMBER(-1.12, -5., .14, F, 0., 1)
C      CALL SYMBOL( 999., 999., .14, 7H MHZ. H, 0., 7)
C      CALL SYMBOL( 999., 999., .14, COMP(JC,1), 0., -1)
C      CALL SYMBOL( 999., 999., .14, COMP(JC,2), 0., 3)
C
C      INITIALIZE THE VALUE OF THE ANGLE

```

```

C      AND PLOT THE FIRST POINT
C
20    DA = ARANGE / ((B(2) - B(1)) + (B(4) - B(3)) + (B(6) - B(5)) + 2)
      A=A0
      X = H(B(1)) * COS(A) * 0.1
      Y = H(B(1)) * SIN(A) * 0.1
      WRITE(6,2000) B(1), H(B(1)), A, X, Y
      CALL SYMBOL(X, Y, .07, 10, 0., -1)
      I = B(1) + 1
      IF(I .EQ. B(2) + 1) I = B(3)
      IF(.NOT. NAVDAT) GO TO 30
      ODCMIN = ODCINT(T(B(1)))
      ODCPAN = ODCINT(T(B(6))) - ODCMIN
C
C      DECREMENT THE ANGLE (ROTATION IS CLOCKWISE)
C      AND PLOT THE NEXT POINT
C
30    IF(NAVDAT) GO TO 40
      A = A - DA
      GO TO 45
40    A = A0 - ARANGE * (ODCINT(T(I)) - ODCMIN) / ODCPAN
45    IF(A .LT. -3.14159) A = A + 6.28318
      X = H(I) * COS(A) * 0.1
      Y = H(I) * SIN(A) * 0.1
      WRITE(6,3000) I, H(I), A, X, Y
      CALL SYMBOL(X, Y, .07, 10, 0., -2)
      I = I + 1
      IF(I .EQ. B(2) + 1) I = B(3)
      IF(I .EQ. B(4) + 1) I = B(5)
      IF(I .EQ. B(6) + 1) GO TO 90
C
      GO TO 30
C
C      REDEFINE THE PLOTTER ORIGIN FOR A POSSIBLE NEXT ENTRY;
C      THEN RETURN
C
90    CALL PLOT(8.50, 0.0, -3)
      RETURN
1000  FORMAT('0', F5.1, ' MHZ.', ' ', A8, ' COMPONENT' / 1X, I5, ' POINTS'
.      // 1X, 'POINTS', I5, ' TO ', I5 /
.      // 7X, I5, ' TO ', I5 /
.      // 4X, 'AND', I5, ' TO ', I5, ' WILL BE PLOTTED')
2000  FORMAT('0 # DB', 7X, 'ANGLE', 2X, 'Y (PLOT) Y' /
.      // '0', I4, 2X, F6.2, F10.5, 5X, 2F9.3)
3000  FORMAT(1Y, I4, 2X, F6.2, F10.5, 5X, 2F9.3)
      END

```

```
*****
*
*                               BUBBLE
*
*****
```

```

SUBROUTINE BUBBLE(X, IX, N)
C
REAL*4 X(N)
INTEGER*4 IX(N)
C
C
DO 10 I = 1, N
    IX(I) = I
10 CONTINUE
C
C
    NN = N - 1
C
C
DO 50 I = 1, NN
    IF(X(IX(I)) .LE. X(IX(I + 1))) GO TO 50
C
C
        SWITCH
C
        IT          = IX(I)
        IX(I)       = IX(I + 1)
        IX(I + 1)   = IT
C
C
        II = I - 1
        JJ = I
C
C
        BUBBLE
C
DO 40 J = 1, II
    JJ = JJ - 1
    IF(X(IX(JJ)) .LE. X(IX(JJ + 1))) GO TO 50
C
C
        SWITCH
C
        IT          = IX(JJ)
        IX(JJ)       = IX(JJ + 1)
        IX(JJ + 1)   = IT
40 CONTINUE
50 CONTINUE
C
```



```

N = 0
C
C      LOOP THROUGH THE RANGE ARRAY FOR THIS FREQUENCY.
C
DO 10 I = 1, 140
C
C      FIND TEMPERATURE VALUE FOR CRANGE(I)
C
IF(CRANGE(I, IFREQ) .GT. 1667.) GO TO 70
IF(CRANGE(I, IFREQ) .GT. TRANGE(1)) GO TO 20
T(I) = TEMP(1)
GO TO 50
20 CONTINUE
DO 40 J = 1, 385
IF(CRANGE(I, IFREQ) .GT. TRANGE(J + 1)) GO TO 40
T(I) = TEMP(J) + (TEMP(J + 1) - TEMP(J))
      * (CRANGE(I, IFREQ) - TRANGE(J))
      / (TRANGE(J + 1) - TRANGE(J))
GO TO 50
40 CONTINUE
50 EBN(I) = AN(IFREQ) * (T(I) - 66.) + BN(IFREQ)
EBNPA(I) = ANPAMP(IFREQ) * (T(I) - 66.) + BNPAMP(IFREQ)
DN(I) = CAL(I, IFREQ, 3) - EBN(I)
DNPAMP(I) = CAL(I, IFREQ, 2) - EBNPA(I)
N = N + 1
EMN = EMN + DN(I)
EPAMN = EPAMN + DNPAMP(I)
EGMN = EGMN + CAL(I, IFREQ, 1)
60 CONTINUE
70 EMN = EMN / N
EPAMN = EPAMN / N
EGMN = EGMN / N
DO 90 I = 1, N
DDN = ABS(DN(I) - EMN)
DDNPA = ABS(DNPAMP(I) - EPAMN)
ESIG = ESIG + DDN * DDN
EPASIG = EPASIG + DDNPA * DDNPA
WRITE(6, 2000) CRANGE(I, IFREQ), T(I), CAL(I, IFREQ, 3),
      EBN(I), DN(I), DDN, CAL(I, IFREQ, 2),
      EBNPA(I), DNPAMP(I), DDNPA
CIG = CAL(I, IFREQ, 1) - EGMN
EGSIG = EGSIG + CIG * CIG
WRITE(8, 4000) CRANGE(I, IFREQ), CAL(I, IFREQ, 1), CIG
90 CONTINUE
ESIG = SORT(ESIG / (N - 1))
EPASIG = SORT(EPASIG / (N - 1))
WRITE(6, 3000) EMN, EPAMN, ESIG, EPASIG
EGSIG = SORT(EGSIG / (N - 1))

```



```

1  B(I)=B(J)+A(ISUB+J)*F(J)
C
2  CONTINUE
C
C..COPY B BACK INTO A.
   DO 3 I=K,L
3  A(I)=B(I)
C
   WRITE(6,4) M,N,F
4  FORMAT('0',I3,'-POINT FILTERING COMPLETED ON',I4,' POINTS. FILTER
*COEFFICIENTS WERE:'/(7X,14F9.4/))
C
   RETURN
C
5  WRITE(6,6) M,N
6  FORMAT('C***ERROR*** ATTEMPT TO USE',I4,'-POINT FILTER ON',I4,
* ' POINTS: FILTER REQUEST IGNORED.'/)
   RETURN
   END

```

```

*****
*
*
*
*
*
*****

```

GAPLOT

```

SUBROUTINE GAPLOT(FREQ, X, Y, N, YMIN, YMAX, NCOMP)
C
C  PLOT RANGE VS. RECORD NUMBER AND V.C.O. VS. RECORD NUMBER FOR
C  DATA IN THE AREA OF THE TURN AT EP-4
C
C  PARAMETERS ARE:
C
C      FREQ  - FREQUENCY
C
C      X      - RANGE ARRAY
C
C      Y      - V.C.O. ARRAY
C
C      N      - NUMBER OF VALUES IN X OR Y
C
C      YMIN   - MINIMUM V.C.O. VALUE
C
C      YMAX   - MAXIMUM V.C.O. VALUE
C
C      NCOMP  - COMPONENT IDENTIFIER:

```

C
C
C
C
C
C
C
C

ENDFIRE BROADSIDE

RHO	1	4
PHI	2	5
ZFD	3	6

```

REAL*4 X(N),Y(N)
INTEGER*4 ICOMP(6,2) / Z3B, Z24, Z60, Z3R, Z24, Z60,
                3 * 'END', 3 * 'END' /
WRITE (6,100) FREQ, NCOMP, N, YMIN, YMAX
WRITE(6,200) X
WRITE(6,300) Y
    
```

C
C
C
C

N POINTS ARE TO BE PLOTTED OVER A RANGE IN (PLOTREP) Y OF 8.589 INCHES.

```

DX = 8.589 / N
CALL PLOT(0., 8.,3)
CALL PLOT(0., 0.,2)
    
```

C
C
C
C
C
C

SET AN INTEGRAL MINIMUM V.C.O. VALUE; IF THE RANGE OF V.C.O. VALUES IS GREATER THAN 35 DB. ADJUST THE DATA TO FIT WITHIN A 7 INCH PLOTTER RANGE - OTHERWISE THE SCALE IS FIXED AT 5 DB. / INCH.

```

YMINN=AINT(YMIN)
DY= AMAX1(5., AINT((AINT(YMAX - YMINN) + 1.) / 7.) + 1.)
WRITE (6, 150) DY
DY = 1. / DY
DO 20 I = 1, 7
YY = I
YYY = YMINN + YY / DY
CALL SYMBOL(0.,YY,.07,13,90.,-1)
CALL NUMBER(-.1,YY-.05,.07,YYY,90.,-1)
20 CONTINUE
    
```

20 CONTINUE

C
C
C
C

FINISH LABELLING THE AXIS; THEN PLOT A LABEL GIVING FREQUENCY AND COMPONENT

```

CALL SYMBOL(-.25, 6.25, .14, 5HV C O, 90., 5)
CALL NUMBER(4.5,0.,.14,FREQ,0.,1)
CALL SYMBOL(999.,999.,.14,7H MHZ. H,0.,7)
CALL SYMBOL(999.,999.,.14,ICOMP(NCOMP,1),0.,-1)
CALL SYMBOL(999.,999.,.14,ICOMP(NCOMP,2),0., 3)
    
```

C
C

PLOT A SYMBOL FOR EACH V.C.O. VALUE.

```

C
  DO 40 I=1,N
  XX=DX*I
  YY = DY * (Y(I) - YMINN)
  CALL SYMBOL(XX,YY,.07,10,0.,-1)
40 CONTINUE
C
C   REDEFINE THE ORIGIN FOR THE NEXT PLOT:
C   THEN RETURN.
C
  CALL PLOT(2.5, 0., -3)
  RETURN
100 FORMAT('OFRFQ.=',F6.1,' COMPONENT',I2/I6,' POINTS'/
.      ' MIN. V.C.O.=', F6.1 / ' MAX. V.C.O.=', F6.1)
150 FORMAT('OSCALE =', F6.1, ' DE / INCH')
200 FORMAT('ORANGE ARRAY:'/100(1X,10F10.1/))
300 FORMAT('OV.C.O. ARRAY:'/100(1X,10F10.1/))
  END

```

```

*****
*
*                               INTPOL
*
*****

```

```

SUBROUTINE INTPOL (XIN,YIN,N,XOUT,YOUT,NSTART,NPLOT)
C
C   LINEAR INTERPOLATION OF YIN VS XIN AT POINTS YOUT. FEB 18/73.
C
C   INPUT:
C   XIN = INPUT X ARRAY
C   YIN = INPUT Y ARRAY
C   N = DIMENSION OF XIN AND YIN
C   YOUT = POINTS AT WHICH YIN WILL BE INTERPOLATED
C   NPLOT = DIMENSION OF XOUT AND YOUT
C
C   OUTPUT:
C   YOUT = INTERPOLATED VALUES OF YIN AT POINTS XOUT
C   NSTART = NUMBER OF FIRST POINT INTERPOLATED
C   NPLOT = NUMBER OF LAST POINT INTERPOLATED
C
  DIMENSION XOUT(NPLOT),YOUT(NPLOT),XIN(N),YIN(N)
  NSTART=1
  I=1
C
C   DO LOOP TO INTERPOLATE YOUT AT EACH XOUT POINT.

```

```

C     CHECKS ARE MADE FOR BEGINNING AND END OF YIN.
C
C     DO 50 J=1,NPLOT
40    IF (XIN(I)-XOUT(J)) 10,20,30
C
10    IF (J.EQ.N) GO TO 60
      I=I+1
      GO TO 40
C
20    YOUT(J)=YIN(I)
      GO TO 50
C
30    IF (I.EQ.1) GO TO 33
      YOUT(J)=YIN(I-1) + (XOUT(J)-XIN(I-1)) * (YIN(I)-YIN(I-1)) / (XIN(I)-
      XIN(I-1))
      GO TO 50
C
33    NSTART=J+1
50    CONTINUE
C
      RETURN
60    NPLOT=J-1
      RETURN
      END

```

```

*****
*
*                               LUNACOPY
*
*****

```

```

REAL*8    TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4    DATA(12000), RANGE(1000), VCO(1000)
REAL*4    FREQ(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4  IDATA(400)
INTEGER*2  ITYPE(2)
LOGICAL*4  FIRST, LAST
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
      ITYPE, N, FIRST, LAST

```

```

C
C
C     READ LUNAR SEP FILE (#1) AND PRODUCE A FILE OF V.C.O. DATA
C     INTERPOLATED AT INTERVALS OF 0.1 WAVELENGTH
C
C     THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".

```

```

C      IORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
C      MAY BE STORED.  IXX IS THE INDEX OF THE FIRST RANGE VALUE,
C      AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C
C      READ AND WRITE THE LABEL RECORD.
C      THIS RECORD CONTAINS N - THE NUMBER OF
C      VALUES IN EACH SUBSEQUENT RECORD.
C
C      CALL LUNIN(DATA, IDATA, 8980, 8990)
C      WRITE(6, 3000) TYPE
C      WRITE(6, 1000) RUN, SITE, DIRECT, FORREV, TITLE, N
C      WRITE(3)      RUN, SITE, DIRECT, FORREV, TITLE, N
C
C      INITIALIZE THE STACK
C
C      10 IORG=1
C         M=0
C         L=0
C
C      CHECK FOR STACK OVEPFLOW BEFORE READING THE NEXT RECORD.
C
C      20 IF(IORG+N .GT. 12000) GO TO 970
C         CALL LUNIN(DATA(IORG), IDATA, 8980, 8990)
C         IF(ITYPE(1) .GE. 6) GO TO 40
C         WRITE(6, 2000) TYPE
C         GO TO 20
C
C      THIS SECTION IS ENTERED ONLY FOR
C      RANGE AND V.C.O. RECORDS.
C
C      40 WRITE(6, 3000) TYPE
C         IF(ITYPE(2) .EQ. 6) GO TO 60
C
C      FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE
C      LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).
C
C      IF(FIRST) IXX=IORG
C      IORG=IORG+N
C      M=M+1
C      IF(.NOT. LAST) GO TO 20
C
C      AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY,
C      FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING
C      TO 0.1 WAVELENGTH INCREMENTS;  THEN COMPUTE THE NUMBER OF VALUES.
C

```



```

      DWL=29.97925/FREQ(ITYPE(1)-5)
      DO 50 I=1, 1000
      RANGE(I)=DWL*FLOAT(I-1)
50  CONTINUE
      NPTSIN=M*N
      GO TO 20

```

C
C
C
C
C
C

TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (L) EQUALS N.

```

60  IF(FIRST) IXI=IORG
      IORG=IOFG+N
      L=L+1
      IF(L .LT. M) GO TO 20

```

C
C
C
C

CALL INTPOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS; THE NEW VALUES ARE RETURNED IN ARRAY "VCO".

```

      NSTART=1
      NPLOT=1000
      CALL INTPOL(DATA(IXI), DATA(IYI), NPTSIN, RANGE, VCO, NSTART, NPLOT)

```

C
C
C

SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.

```

      NSTM1=NSTART-1
      IF(NSTM1 .LE.0) GO TO 80
      DO 70 I=1, NSTM1
      VCO(I)=0.0
70  CONTINUE
80  NPLTP1=NPLOT+1
      IF(NPLTP1 .GT. 1000) GO TO 100
      DO 90 I=NPLTP1, 1000
      VCO(I)=0.0
90  CONTINUE
100 NPTS=NPLOT-NSTM1

```

C
C
C

WRITE HEADER INFORMATION AND ARRAY "VCO".

```

      WRITE(6,4000) TYPE(1), FREQ(ITYPE(1)-5), NSTART, NPTS,
      .             (VCO(I), I=1, NPTS)
      WRITE(3)      TYPE(1), FREQ(ITYPE(1)-5), NSTART, NPTS,
      .             (VCO(I), I=1, NPTS)

```

C
C
C
C

IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHERWISE READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA

```

C     ARE RETAINED).
C
C
C     IF(LAST) GO TO 10
C     IORG=IXY
C     L=0
C     GO TO 20
C
C
C     STACK OVERFLOW MESSAGE
C
C
C     970 WRITE(6,7000)
C     GO TO 999
C
C
C     END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED
C
C
C     980 WRITE(6, 5000)
C     GO TO 999
C
C
C     PROCESSING COMPLETED NORMALLY
C
C
C     990 WRITE(6, 6000) TYPE
C     999 END FILE 3
C     RETURN
C
C
C     1000 FORMAT('ORUN ',A6/'OSITE ',A6/'ODIRECTION ',A6/
C     .          'O',A6,' TRANSMITTER'/'O',10A8,A4/'O',I4,' POINTS')
C     2000 FORMAT('O',2A8,' RECORD SKIPPED')
C     3000 FORMAT('O',2A8,' RECORD READ')
C     4000 FORMAT('1LABEL="',A8,'" / 'OFREQ.= ',F5.1,' MHZ.' /
C     .          'OFIRST POINT=',I4/ 'O# OF POINTS=',I4/
C     .          'O',10F10.3/99(1X,10F10.3/))
C     5000 FORMAT('ONORMAL END OF JOB')
C     6000 FORMAT('OEND OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
C     .          2A8,' RECORD')
C     7000 FORMAT('-*** INSUFFICIENT SPACE ON STACK ***')
C
C
C     END

```

```
*****
*
*                               LUNACPY2
*
*****
```

```
REAL*8      TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4      DATA(12000), RANGE(1000), VCO(1000)
REAL*4      FREQ(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4   IDATA(400)
INTEGER*2   ITYPE(2)
LOGICAL*4   FIRST, LAST
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
          ITYPE, N, FIRST, LAST
```

```
C
C
C   READ LUNAR SEP FILE (#2) AND PRODUCE A FILE OF V.C.O. DATA
C   INTERPOLATED AT INTERVALS OF 0.1 WAVLENGTH
C
C   THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
C   IORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
C   MAY BE STORED.  IXX IS THE INDEX OF THE FIRST RANGE VALUE,
C   AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C
C   READ AND WRITE THE LABEL RECORD.
C   THIS RECORD CONTAINS N - THE NUMBER OF
C   VALUES IN EACH SUBSEQUENT RECORD.
C
C   N=386
C   CALL LUNIN2(DATA,          IDATA, 8980, 8990)
C   WRITE(6, 3000) TYPE
C   WRITE(6, 1000) (IDATA(J), I=1, 297)
C
C   INITIALIZE THE STACK
C
10  IORG=1
    M=0
    L=0
C
C   CHECK FOR STACK OVERFLOW BEFORE READING THE NEXT RECORD.
C
20  IF(IORG+N .GT. 12000) GO TO 970
    CALL LUNIN2(DATA(IORG), IDATA, 8980, 8990)
    IF(ITYPE(1) .GE. 6) GO TO 40
    WRITE(6, 2000) TYPE
```

```

GO TO 20

C
C
C THIS SECTION IS ENTERED ONLY FOR
C RANGE AND V.C.O. RECORDS.
C
C
C
40 WRITE(6, 3000) TYPE
   IF(ITYPE(2) .EQ. 6) GO TO 60

C
C FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE
C LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).
C
   IF(FIRST) IXX=IORG
   IORG=IORG+N
   M=M+1
   IF(.NOT. LAST) GO TO 20

C
C AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY,
C FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING
C TO 0.1 WAVELENGTH INCREMENTS; THEN COMPUTE THE NUMBER OF VALUES.
C
   DWL=29.97925/FREQ(ITYPE(1)-5)
   DO 50 I=1, 1000
   RANGE(I)=DWL*FLOAT(I-1)
50 CONTINUE
   NPTSIN=M*N
   GO TO 20

C
C
C TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES
C HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (L) EQUALS M.
C
C
60 IF(FIRST) IXY=IORG
   IORG=IORG+N
   L=L+1
   IF(L .LT. M) GO TO 20

C
C CALL INTPOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS;
C THE NEW VALUES ARE RETURNED IN ARRAY "VCO".
C
   NSTART=1
   NPLOT=1000
   CALL INTPOL(DATA(IXX), DATA(IXY), NPTSIN, RANGE, VCO, NSTART, NPLOT)

C
C SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.

```

```

C
  NSTM1=NSTART-1
  IF(NSTM1 .LE.0) GO TO 30
  DO 70 J=1, NSTM1
  VCO(I)=0.0
70 CONTINUE
80 NPLTP1=NPLOT+1
  IF(NPLTP1 .GT. 1000) GO TO 100
  DO 90 I=NPLTP1, 1000
  VCO(I)=0.0
90 CONTINUE
100 NPTS=NPLOT-NSTM1
C
C   WRITE HEADER INFORMATION AND ARRAY "VCO".
C
  DO 120 I=NSTART, NPLOT
  VCO(I)=VCO(I)+135.0
120 CONTINUE
  WRITE(6,4000) TYPE(1),FREQ(ITYPE(1)-5),NSTART,NPTS,
  .           (VCO(I),I=1,NPTS)
  WRITE(3)    TYPE(1),FREQ(ITYPE(1)-5),NSTART,NPTS,
  .           (VCO(I),I=1,NPTS)
C
C   IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHERWISE
C   READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA
C   ARE RETAINED).
C
C
  IF(LAST) GO TO 10
  IORG=IXY
  L=0
  GO TO 20
C
C
C   STACK OVERFLOW MESSAGE
C
C
970 WRITE(6,7000)
  GO TO 999
C
C
C   END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED
C
C
980 WRITE(6,5000)
  GO TO 999
C

```

```

C
C   PROCESSING COMPLETED NORMALLY
C
C
C   990 WRITE(6, 6000) TYPE
C   999 END FILE 3
C       RETURN
C
C
C   1000 FORMAT(27(1X,11A4/))
C   2000 FORMAT('0',2A8,' RECORD SKIPPED')
C   3000 FORMAT('0',2A8,' RECORD READ')
C   4000 FORMAT('1LABEL=',A8,'"/ 'OPREQ.= ',F5.1,' MHZ.'/
C       .      'OFIRST POINT=',I4/ 'O# OF POINTS=',I4/
C       .      '0',10F10.3/99(1X,10F10.3/))
C   5000 FORMAT('ONORMAL END OF JOB')
C   6000 FORMAT('OEND OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
C       .      2A8,' RECORD')
C   7000 FORMAT('-*** INSUFFICIENT SPACE ON STACK ***')
C
C
C   END

```

```

*****
*
*                               LUNACPY3
*
*****

```

```

REAL*8      TYPE(2), FUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4      DATA(12000), RANGE(1000), VCO(1000)
REAL*4      FREQ(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4   IDATA(400)
INTEGER*2   ITYPE(2)
LOGICAL*4   FIRST, LAST
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, FUN, SITE, DIRECT, FORREV, TYPE,
.           ITYPE, N, FIRST, LAST

```

```

C
C
C   READ LUNAR SEP FILE (#3) AND PRODUCE A FILE OF V.C.O. DATA
C   INTERPOLATED AT INTERVALS OF 0.1 WAVELENGTH
C   RANGE DATA ARE TAKEN FROM FILE #2
C
C   THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
C   IORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA

```

```

C     MAY BE STORED.  IXV IS THE INDEX OF THE FIRST RANGE VALUE,
C     AND IXV IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C
C     READ AND WRITE THE LABEL RECORD.
C     THIS RECORD CONTAINS N - THE NUMBER OF
C     VALUES IN EACH SUBSEQUENT RECORD.
C
C     N=385
C     CALL LUNIN2(DATA,          IDATA, 8980, 8990)
C     CALL LUNIN3(DATA,          IDATA, 8980, 8990)
C     WRITE(6, 3000) TYPE
C     WRITE(6, 1000) (IDATA(I), I=1, 297)
C
C     INITIALIZE THE STACK
C
C     10 IORG=1
C        M=0
C        L=0
C
C     CHECK FOR STACK OVERFLOW BEFORE READING THE NEXT RECORD.
C
C     20 IF(IORG+N .GT. 12000) GO TO 970
C        CALL LUNIN2(DATA(IORG), IDATA, 8980, 8990)
C        IF(ITYPE(2) .NE. 5)
C          CALL LUNIN3(DATA(IORG), IDATA, 8980, 8990)
C        IF(ITYPE(1) .GE. 6) GO TO 40
C        WRITE(6, 2000) TYPE
C        GO TO 20
C
C
C     THIS SECTION IS ENTERED ONLY FOR
C     RANGE AND V.C.O. RECORDS.
C
C
C     40 WRITE(6, 3000) TYPE
C        IF(ITYPE(2) .EQ. 6) GO TO 60
C
C     FOR RANGE DATA - MOVE IORG TO POINT ONE LOCATION BEYOND THE
C     LAST VALUE, AND INCREMENT THE COUNT OF RANGE BLOCKS (M).
C
C     IF(FIRST) IXV=IORG
C     IORG=IORG+N
C     M=M+1
C     IF(.NOT. LAST) GO TO 20
C
C
C     AFTER READING THE LAST RANGE BLOCK FOR THIS FREQUENCY,
C     FILL ARRAY "RANGE" WITH DISTANCES IN METERS CORRESPONDING

```

```

C      TO 0.1 WAVELENGTH INCREMENTS; THEN COMPUTE THE NUMBER OF VALUES.
C
      DWL=29.97925/FREQ(ITYPE(1)-5)
      DO 50 J=1, 1000
      RANGE(I)=DWL*FLOAT(I-1)
50 CONTINUE
      NPTSIN=M*N
      GO TO 20

C
C
C      TREATMENT OF V.C.O. DATA IS SIMILAR; ONE SET OF V.C.O. VALUES
C      HAS BEEN ACCUMULATED WHEN THE COUNT OF V.C.O. BLOCKS (L) EQUALS M.
C
C
60 IF(FIRST) IXI=IORG
      IORG=IORG+N
      L=L+1
      IF(L .LT. M) GO TO 20

C
C      CALL INTPOL TO OBTAIN V.C.O. VALUES AT EQUAL RANGE INTERVALS;
C      THE NEW VALUES ARE RETURNED IN ARRAY "VCO".
C
      NSTART=1
      NPLOT=1000
      CALL INTPOL(DATA(IXI), DATA(IXY), NPTSIN, RANGE, VCO, NSTART, NPLOT)

C
C      SET TO ZERO V.C.O. VALUES WHICH HAVE NOT BEEN INTERPOLATED.
C
      NSTM1=NSTART-1
      IF(NSTM1 .LE. 0) GO TO 80
      DO 70 I=1, NSTM1
      VCO(I)=0.0
70 CONTINUE
80 NPLTP1=NPLOT+1
      IF(NPLTP1 .GT. 1000) GO TO 100
      DO 90 I=NPLTP1, 1000
      VCO(I)=0.0
90 CONTINUE
100 NPTS=NPLOT-NSTM1

C
C      WRITE HEADER INFORMATION AND ARRAY "VCO".
C
      DO 120 I=NSTART, NPLOT
      VCO(I)=VCO(I)+135.0
120 CONTINUE
      WRITE(6,4000) TYPE(1), FREQ(ITYPE(1)-5), NSTART, NPTS,
      .           (VCO(I), I=1, NPTS)
      WRITE(3)     TYPE(1), FREQ(ITYPE(1)-5), NSTART, NPTS,

```



```

      (VCO(I),I=1,NPTS)
C
C
C   IF LAST IS TRUE THEN READ A NEW SET OF RANGE VALUES; OTHERWISE
C   READ V.C.O. DATA FOR THE NEXT COMPONENT (THE CURRENT RANGE DATA
C   ARE RETAINED).
C
C   IF(LAST) GO TO 10
C   IORG=IXY
C   L=0
C   GO TO 20
C
C   STACK OVERFLOW MESSAGE
C
C   970 WRITE(6,7000)
C   GO TO 999
C
C   END OF FILE ON INPUT WHEN MORE DATA WERE EXPECTED
C
C   980 WRITE(6,5000)
C   GO TO 999
C
C   PROCESSING COMPLETED NORMALLY
C
C   990 WRITE(6,6000) TYPE
C   999 END FILE 3
C   RETURN
C
C   1000 FORMAT(27(1X,11A4/))
C   2000 FORMAT('0',2A8,' RECORD SKIPPED')
C   3000 FORMAT('0',2A8,' RECORD READ')
C   4000 FORMAT('1 LABEL="'',A8,'"/ '0 FREQ. = ',F5.1,' MHZ.'/
C   .       '0 FIRST POINT='',I4/ '0# OF POINTS='',I4/
C   .       '0',10F10.3/99(1X,10F10.3/))
C   5000 FORMAT('0 NORMAL END OF JOB')
C   6000 FORMAT('0 END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
C   .       2A8,' RECORD')
C   7000 FORMAT('0 -*** INSUFFICIENT SPACE ON STACK ***')
C
C
C

```



```

LOGICAL*1 DECIDE(6,6) / 36 * .TRUE. /
NAMLIST / CNTL / IREQ, ICOMP
EQUIVALENCE (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
      ITYPE, N, FIRST, LAST

```

```

C
C   BEGIN BY SETTING DEFAULT VALUES FOR COMPONENT SELECTION
C   (NO COMPONENTS), READING CONTROL CARDS, AND SETTING
C   CONTROL PARAMETERS
C

```

```

      DO 10500 I=1,6
      DO 10100 J=1,6
10100  ICOMP(J)=0
      READ(5,CNTL,END=10600)
      IDY=IREQ+1
      DO 10120 J=1,6
      IC=COMP(J)
      DO 10110 K=1,6
      IF(IC.EQ.ICOMP(K)) GO TO 10120
10110  CONTINUE
      DECIDE(IDY,J)=.FALSE.
10120  CONTINUE
10500  CONTINUE
10600  CONTINUE
      N=386

```

```

C
C   SKIP THE LABEL BLOCK
C
      CALL LUNIN2(DATA,      IDATA, 8980, 8990)
      CALL LUNIN3(DATA,      IDATA, 8980, 8990)

```

```

C
C   INITIALIZE THE STACK
C

```

```

10  IORG=1
      M=0
      L=0
20  IF(IORG+N .GT. 12000) GO TO 970
      CALL LUNIN2(DATA(IORG), IDATA, 8980, 8990)
      IF(ITYPE(2) .NE. 5)
      . CALL LUNIN3(DATA(IORG), IDATA, 8980, 8990)
      IF(ITYPE(1) .GE. 6) GO TO 40
      GO TO 20

```

```

C
C
40  CONTINUE
      IF(ITYPE(2) .EQ. 6) GO TO 60
C
C   ACCUMULATE RANGE BLOCKS

```

```

C
  IF(FIRST) IYX=IORG
  IORG=IORG+N
  M=M+1
  IF(.NOT. LAST) GO TO 20
  NPTSIN=N*M
  IGX=IXX
  IGXEND=IXX

C
C  FIND THE POINTS WHICH LIE BETWEEN 490 AND 535 MPPRPS;
C  THESE WILL BE COPIED
C
  DO 50 I=IXX,NPTSIN
  IF(DATA(I).LE.490.) IGX=IGX+1
  IF(DATA(I) .LE. 535.) IGXEND = IGXEND + 1
50 CONTINUE
  NCOMP=0
  GO TO 20

C
C  ACCUMULATE VCO BLOCKS
C
60 IF(FIRST) IYV=IORG
  IORG=IORG+N
  L=L+1
  IF(L .LT. M) GO TO 20
  IF(NCOMP .GT. 0) GO TO 65

C
C  ISOLATE THE POINTS TO BE COPIED
C
  IGY=IYV+IGX-IXX
  IGYEND=IYV+IGXEND-IXX
  NPTS=IGYEND-IGX
65 CONTINUE
  NCOMP=NCOMP+1
  IF(.NOT. DECIDE(ITYPE(1)-5,NCOMP)) GO TO 150
  YMIN=DATA(IGY)+135.
  YMAX=YMIN
  IY=IGY

C
C  ADJUST THE DATA VALUES TO RELATIVE DB, AND FIND
C  MAXIMUM AND MINIMUM VALUES
C
  DO 70 I=1,NPTS
  DATA(IY)=DATA(IY)+135.
  IF(DATA(IY) .LT. YMIN) YMIN=DATA(IY)
  IF(DATA(IY) .GT. YMAX) YMAX=DATA(IY)
  IY=IY+1
70 CONTINUE

```

```

      IF(NCOMP .GT. 1) GO TO 75
      IGXEND = IGXEND - 1
      IGYEND = IGYEND - 1
75  CONTINUE
C
C   WRITE OUT THE ACCUMULATED DATA
C
      WRITE(1) FREQ(ITYPE(1)-5), NCOMP, YMIN, YMAX, NPTS,
      .         (DATA(I), I=IGX,IGXEND), (DATA(I), I=IGY,IGYEND)
C
C   IF THIS WAS THE SIXTH COMPONENT FOR THIS FREQUENCY, READ
C   RANGE DATA FOR THE NEXT FREQUENCY; OTHERWISE READ
C   VCO DATA FOR THE NEXT COMPONENT
C
150  IF(LAST) GO TO 10
      IORG=IXY
      L=0
      GO TO 20
C
C   STACK ARRAY TOO SMALL
C
970  WRITE(6,7000)
      GO TO 999
C
C   ALL PROCESSING COMPLETED NORMALLY
C
980  WRITE(6, 5000)
      GO TO 999
C
C   PREMATURE END OF INPUT FILE
C
990  WRITE(6, 6000) TYPE
999  END FILE 1
      RETURN
C
C
1000 FORMAT(27(1X,11A4/))
2000 FORMAT('0',2A8,' RECORD SKIPPED')
3000 FORMAT('0',2A8,' RECORD READ')
4000 FORMAT('1 LABEL="',A8,'" / 'OFREQ.= ',F5.1,' MHZ.' /
      .         'OFIRST POINT=',I4 / 'O# OF POINTS=',I4 /
      .         '0',10F10.3/99(1X,10F10.3/))
5000 FORMAT('ONORMAL END OF JOB')
6000 FORMAT('OEND OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
      .         2A8, ' RECORD')
7000 FORMAT('O-*** INSUFFICIENT SPACE ON STACK ***')
C
C

```

END

```
*****
*
*                               LUNACPY5
*
*****
```

```
C      PROGRAM TO EXTRACT TEMPERATURE, CALIBRATION,
C      TRANSMITTEE-OFF, AND SELECTED RANGE INFORMATION FROM
C      LUNAR SEP FILE # 2.
```

```
C      THE RANGE DATA ASSOCIATED WITH THE TEMPERATURE DATA ARE
C      A DIRECT COPY OF THE 1 MHZ. RANGE ARRAY.
```

```
C      A SECOND ARRAY CONTAINS RANGE VALUES MATCHED WITH THE
C      CALIBRATION AND TXOFF DATA BY SELECTING EVERY 13-TH POINT
C      FROM THE 32 MHZ. RANGE ARRAY, BEGINNING WITH THE 7-TH.
```

```
C      THE ARRAY OF TEMPERATURE DATA IS COPIED DIRECTLY OFF THE
C      INPUT FILE. THE CALIBRATION AND TXOFF DATA ARE IN A
C      MULTIPLEXED FORM ON THE INPUT FILE (C.F. - I. WATTS NOTES
C      - 13.I.74). THE PROGRAM DEMULTIPLEXES THIS INFORMATION
C      AND STORES IT IN TWO ARRAYS:
```

```
C      CAL(I, IFRQ, J), AND
C
C      TXOFF(I, IFRQ, K),
```

```
C      WHERE I INDICATES THE I-TH VALUE IN SEQUENCE AND IFRQ IS
C      THE (INTEGRAL) BASE-2 LOGARITHM OF THE FREQUENCY.
C      J = 1, 2, 3 CORRESPOND TO CALIBRATION FOR GROUND,
C      NOISE DIODE + 20 DB, AND NOISE DIODE SOURCES RESPECTIVELY.
C      K = 1, 2, 3 INDICATE TXOFF INFORMATION FOR THE X, Y,
C      AND Z ANTENNAE RESPECTIVELY.
```

```
C
C      REAL*4 RANGE(386), DATA(5018), RANGE2(140, 6)
C      REAL*4 CAL(140, 6, 3), TXOFF(140, 6, 3)
C      REAL*4 SPEED(140, 6)
C      INTEGER*4 MODE(386)
C      INTEGER*4 NCAL(6, 3) / 18 * 0 /
C      INTEGER*4 NTXOFF(6, 3) / 18 * 0 /
C      INTEGER*4 NR(3) / 3 * 0 /
C      INTEGER*2 ITYPE(2)
```

```

LOGICAL*4 FIRST, LAST
C
COMMON / LUNDAT / JUNK(34), ITYPE, N, FIRST, LAST
C
C
N = 386
IORG = 1
DO 15 K = 1, 6
  DO 10 J = 1, 140
    RANGE2(J, K) = 0.
    DO 5 L = 1, 3
      CAL(J, K, L) = 0.
      TXOFF(J, K, L) = 0.
5    CONTINUE
10   CONTINUE
15   CONTINUE
C
C
20 CALL IUNIN2(DATA(IORG), MODE, 8900, 8900)
C
IT = ITYPE(1)
GO TO (20, 40, 100, 200, 300, 400, 20, 20, 20, 20, 500), IT
C
C
C      DELETE THE 'R' DIGIT FROM EACH ELEMENT OF THE MODE ARRAY.
C
40 CONTINUE
DO 60 I = 1, N
  MODE(I) = MODE(I) / 10
60 CONTINUE
GO TO 20
C
C
C      THE TEMPERATURE ARRAY IS WRITTEN OUT IMMEDIATELY.
C
100 WRITE(3)      (DATA(I), I = 1, N)
WRITE(6,1000) (DATA(L), I = 1, N)
GO TO 20
C
C
C      DEMULTIPLEX TXOFF DATA INTO ARRAY TXOFF;
C      NTXOFF(IFREQ, M) CONTAINS THE MAXIMUM K FOR
C      TXOFF(K, IFREQ, M).
C
200 IF(FIRST) MM = 0
M = MOD(MM, 3) + 1
MM = MM + 1
L = 0

```

```

IF(MM .GT. 3) L = 1
DO 250 I = 1, N
  IFREQ = 2 * (4 - MOD(MODE(I), 10)) - L
  NTXOFF(IFREQ, M) = NTXOFF(IFREQ, M) + 1
  TXOFF(NTXOFF(IFREQ, M), IFREQ, M) = DATA(I)
250 CONTINUE
GO TO 20

```

C
C
C
C
C

FOLLOW THE SAME PROCEDURE TO DEMULTIPLEX THE
CALIBRATION DATA.

```

300 IF(FIRST) MM = 0
M = MOD(MM, 3) + 1
MM = MM + 1
L = 0
IF(MM .GT. 3) L = 1
DO 350 I = 1, N
  IFREQ = 2 * (4 - MOD(MODE(I), 10)) - L
  NCAL (IFREQ, M) = NCAL (IFREQ, M) + 1
  CAL ( NCAL(IFREQ, M), IFREQ, M) = DATA(I)
350 CONTINUE
GO TO 20

```

C
C
C
C
C

THE RANGE ARRAY FOR 1 MHZ. IS PAIRED WITH THE
TEMPERATURE ARRAY.

```

400 IF(ITYPE(2) .EQ. 6) GO TO 20
WRITE(3) (DATA(I), I = 1, N)
WRITE(6,1020) (DATA(I), I = 1, N)
GO TO 20

```

C
C
C
C

ACCUMULATE 32 MHZ. RANGE BLOCKS

```

500 IF(FIRST) NN = 0
NN = NN + N
IORG = IORG + N
IF(.NOT. LAST) GO TO 20

```

C
C
C
C
C

DEMULITPLEX THE RANGE DATA TO MATCH THE CALIBRATION
AND TXOFF ARRAYS.

```

DO 600 I = 7, NN, 13
  II = (I - 7) / 13 + 1
  IFREQ = 4 - MOD(MODE(II), 10)

```



```

      NR(IFREQ) = NR(IFREQ) + 1
      RANGE2(NR(IFREQ), 2 * IFREQ      ) = DATA(I)
      RANGE2(NR(IFREQ), 2 * IFREQ - 1) = DATA(J)
      SPEED(NR(IFREQ), 2 * IFREQ) =
      .   1.234568 * (DATA(I + 1) - DATA(I - 1))
      SPEED(NR(IFREQ), 2 * IFREQ - 1) =
      .   SPEED(NR(IFREQ), 2 * IFREQ)
600 CONTINUE
C
C
C           WRITE AND LIST THE CALIBRATION, TXOFF, AND
C           ASSOCIATED RANGE INFORMATION.
C
      WRITE(3) CAL, NCAL
      WRITE(3) TXOFF, NTXOFF
      WRITE(3) RANGE2, NR
      WRITE(3) SPEED
C
      DO 700 IFREQ = 1, 6
        IFR = 2 ** (IFREQ - 1)
        NN = NR((IFREQ - 1) / 2 + 1)
        WRITE(6,1050) IFR, ( RANGE2(L, IFREQ),
      .
      .           (CAL(L, IFREQ, K), K = 1, 3),
      .           (TXOFF(L, IFREQ, J), J = 1, 3), L = 1, NN)
700 CONTINUE
C
C
C 900 RETURN
C
C
C 1000 FORMAT(' TEMPERATURE' // 26(1X, 15F8.1 / ))
C
C 1020 FORMAT('ORANGES FOR TEMPERATURE ARRAY' // 26(1X, 15F8.1 / ))
C
C 1050 FORMAT('1', I3, ' MHZ.', 29X, 'CALIBRATION', 37X,
      .   'TRANSMITTER-OFF' / 11X, 'RANGE', 14X, 'GROUND', 6X,
      .   'NOISE +20', 10X, 'NOISE', 19X, 'X', 14X, 'Y', 14X, 'Z'
      .   // 10(1X, F15.1, 2(5X, 3F15.1) / ))
C
      END

```

```
*****
*
*                               LUNALIST
*
*****
```

```
C
C   PROGRAM TO LIST LUNAR DATA
C
C   REAL*8   TITLE(11), RUN, SITE, DIRECT, FORREV, TYPE(2)
C   REAL*4   DATA(825)
C   INTEGER*4 IDATA(825)
C   INTEGER*2 ITYPE(2)
C   INTEGER*2 ITYSAV / 0 /, LINCNT / 0 /
C
C   COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV,
C   .           TYPE, ITYPE, N
C
C   FLAGS FOR TEMPORARY TRAP
C
C   LOGICAL*1 TRAP / .FALSE. /, SKIP / .FALSE. /
C
C   BEGIN EXECUTABLE CODE
C
C   GET (NEXT) INPUT RECORD
C
20 CALL LUNIN (DATA, IDATA, 8400, 8500)
21 CONTINUE
    IF(.NOT.SKIP)
C   .       GO TO 30
C
C   ELSE
C   .       IF(ITYPE(1) .NE. 1)
C   .           GO TO 20
C   .       ELSE
C   .           IF(LINCNT .LE. 44)
C   .               GO TO 22
C   .           ELSE
C   .               WRITE(6, 35) TYPE
C   .               LINCNT = 0
C   .               GO TO 28
22          WRITE(6, 25) TYPE
25          FORMAT('0<<< ', 2A8, ' >>>' / 2X)
28          LINCNT = LINCNT + 16
C   .               GO TO 80
30 IF(ITYPE(1) .LE. 3 .OR. ITYPE(2) .NE. ITYSAV)
C   .       WRITE(6, 35) TYPE
35          FORMAT('1<<< ', 2A8, ' >>>' / 2X)
```

```

                LINCNT = 3
                ITYSAV = ITYPE(1)
C
C      CHOOSE APPROPRIATE OUTPUT FORMAT
C
C      80  IF(ITYPE(1) - 2) 100, 200, 300
C
C          HEADER
C
C      100  WRITE(6, 105) PUN, SITE, DIRECT, FORREV, TITLE, N
C      105  FORMAT ('ORUN ', A6 / 'SITE ', A6 / 'DIRECTION ', A6 /
C          .      '0', A6, ' TRANSMITTER' / '0', 10A8, A4 /
C          .      '0', I6, ' POINTS' )
C
C          CHECK FOR ARRAY OVERFLOW
C
C          IF(N .GT. 825)
C          .      N = 825
C
C          COMPUTE NUMBER OF LINES REQUIRED FOR LISTING
C
C          NL = MAX0(N / 15 , N / 15 + 1) + 2
C
C          TEMPORARY TRAP TO RESTRICT PRINTOUT
C
C          IF(TRAP)
C          .      SKIP = .TRUE.
C          .      TRAP = .TRUE.
C          .      GO TO 20
C
C          MODE
C
C      200  CONTINUE
C
C          TRAP
C
C          IF(SKIP)
C          .      GO TO 20
C
C          WRITE(6, 1000)
C          WRITE(6, 225) (IDATA(I), I=1,N)
C      225  FORMAT(54 (1X, 15I7 / ), 1X, 15I7)
C          GO TO 20
C
C          ALL OTHER DATA
C
C          TRAP
C

```

```

300     CONTINUE
        IF (SKIP)
            GO TO 20
C
        IF (LINCNT + NL .LE. 60)
            GO TO 320
C
        ELSE
            WRITE (6, 35) TYPE
            LINCNT = 3
320     WRITE (6, 1000)
        WRITE (6, 350) (DATA(I), I=1,N)
350     FORMAT (54 (1X, 15F7.1 / ), 1X, 15F7.1)
        LINCNT = LINCNT + NL
        GO TO 20
C
C     RETURN POINTS FOR END OF FILE CONDITIONS
C
400     WRITE (6, 410)
410     FORMAT ('-NORMAL END OF FILE DETECTED')
        GO TO 900
C
500     WRITE (6, 510)
510     FORMAT ('-ABNORMAL END OF FILE DETECTED')
C
900     RETURN
C
1000    FORMAT ('0 ')
C
        END

```

```

*****
*
*                               LUNALST2
*
*****

```

```

C
C     PROGRAM TO LIST LUNAR DATA
C
        REAL*8    TITLE(11), RUN, SITE, DIRECT, FORREV, TYPE(2)
        REAL*4    DATA(825)
        INTEGER*4  JDATA(825)
        INTEGER*2  JTYPE(2)
        INTEGER*2  ITYSAV / 0 /, LINCNT / 0 /
C
        COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV,

```

```

      .           TYPE,   ITYPE,  N
C
C     FLAGS FOR TEMPORARY TRAP
C
C     LOGICAL*1 TRAP / .FALSE. /,  SKIP / .FALSE. /
C
C     BEGIN EXECUTABLE CODE
C
C     GET (NEXT) INPUT RECORD
C
      N=386
20 CALL LUNIN2(DATA, IDATA, 8400, 8500)
21 CONTINUE
      IF(.NOT.SKIP)
      .           GO TO 30
C     ELSE
      .           IF(ITYPE(1) .NE. 1)
      .           .           GO TO 20
C     ELSE
      .           .           IF(LINCNT .IE. 44)
      .           .           .           GO TO 22
C     ELSE
      .           .           .           WRITE(6, 35) TYPE
      .           .           .           LINCNT = 0
      .           .           .           GO TO 28
22          .           .           WRITE(6, 25) TYPE
25          .           .           FORMAT('0<<< ', 2A8, ' >>>' / 2X)
28          .           .           LINCNT = LINCNT + 16
      .           .           .           GO TO 30
30          .           IF(ITYPE(1) .LE. 3 .OR. ITYPE(2) .NE. ITYSAV)
      .           .           WRITE(6, 35) TYPE
35          .           .           FORMAT('1<<< ', 2A8, ' >>>' / 2X)
      .           .           .           LINCNT = 3
      .           .           .           ITYSAV = ITYPE(1)
C
C     CHOOSE APPROPRIATE OUTPUT FORMAT
C
C     80  IF(ITYPE(1) - 2) 100, 200, 300
C
C     HEADER
C
100          WRITE(6, 105) (IDATA(J), I=1, 297)
105          FORMAT (27(1X, 11A4 /), 2X)
C
C     CHECK FOR ARRAY OVERFLOW
C
      IF(N .GT. 825)
      .           N = 825

```

```

C
C      COMPUTE NUMBER OF LINES REQUIRED FOR LISTING
C
C      NL = MAYO(N / 15 , N / 15 + 1 ) + 2
C
C      TEMPORARY TRAP TO RESTRICT PRINTOUT
C
C      IF (TRAP)
C          .      SKIP = .TRUE.
C          TRAP = .TRUE.
C          GO TO 20
C
C      MODE
C
C      200  CONTINUE
C
C      TRAP
C
C      IF (SKIP)
C          .      GO TO 20
C
C      WRITE(6, 1000)
C      225  WRITE(6, 225) (IDATA(I), I=1,N)
C          FORMAT(54 (1X, 15I7 / ), 1X, 15I7)
C          GO TO 20
C
C      ALL OTHER DATA
C
C      TRAP
C
C      300  CONTINUE
C          IF (SKIP)
C              .      GO TO 20
C
C          IF (LINCNT + NL .LE. 60)
C              .      GO TO 320
C              ELSE
C                  WRITE(6, 35) TYPE
C                  LINCNT = 3
C      320  WRITE(6, 1000)
C          WRITE(6, 350) (DATA(I), I=1,N)
C      350  FORMAT(54 (1X, 15F7.1 / ), 1X, 15F7.1)
C          LINCNT = LINCNT + NL
C          GO TO 20
C
C      RETURN POINTS FOR END OF FILE CONDITIONS
C
C      400  WRITE(6, 410)

```

```

410  FORMAT('-NORMAL END OF FILE DETECTED')
      GO TO 900
C
500  WRITE(6, 510)
510  FORMAT('-ABNORMAL END OF FILE DETECTED')
C
900  RETURN
C
1000 FORMAT('0 ')
C
      END

```

```

*****
*
*                               LUNALST3
*
*****

```

```

C
C   PROGRAM TO LIST LUNAR DATA
C
      REAL*8   TITLE(11), RUN, SITE, DIRECT, FORREV, TYPE(2)
      REAL*4   DATA(825)
      INTEGER*4 IDATA(825)
      INTEGER*2 ITYPE(2)
      INTEGER*2 ITYSAV / 0 /, LIMCNT / 0 /
C
      COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV,
      .           TYPE, ITYPE, N
C
      FLAG FOR TEMPORARY TRAP
C
      LOGICAL*1 TRAP / .FALSE. /, SKIP / .FALSE. /
C
      BEGIN EXECUTABLE CODE
C
      GET (NEXT) INPUT RECORD
C
      N=386
20  CALL LUNIN2(DATA, IDATA, 8400, 8500)
      IF(ITYPE(2) .NE. 5)
      .   CALL LUNIN3(DATA, IDATA, 8400, 8500)
21  CONTINUE
      IF(.NOT.SKIP)
      .   GO TO 30
C
      ELSE

```

```

      IF (ITYPE(1) .NE. 1)
        GO TO 20
C      ELSE
C      IF (LINCNT .LE. 44)
C      GO TO 22
C      ELSE
C      WRITE(6, 35) TYPE
C      LINCNT = 0
C      GO TO 28
22      WRITE(6, 25) TYPE
25      FORMAT('0<<< ', 2A8, ' >>>' / 2X)
28      LINCNT = LINCNT + 10
C      GO TO 30
30      IF (ITYPE(1) .LE. 3 .OR. ITYPE(2) .NE. ITYSAV)
C      WRITE(6, 35) TYPE
35      FORMAT('1<<< ', 2A8, ' >>>' / 2X)
C      LINCNT = 3
C      ITYSAV = ITYPE(1)
C
C      CHOOSE APPROPRIATE OUTPUT FORMAT
C
C      30      IF (ITYPE(1) - 2) 100, 200, 300
C
C      HEADER
C
C      100      WRITE(6, 105) (JDATA(I), I=1, 297)
C      105      FORMAT (27(1X, 11A4 /), 2X)
C
C      CHECK FOR ARRAY OVERFLOW
C
C      IF (N .GT. 825)
C      N = 825
C
C      COMPUTE NUMBER OF LINES REQUIRED FOR LISTING
C
C      NL = MAX0(N / 15 , N / 15 + 1) + 2
C
C      TEMPORARY TRAP TO RESTRICT PRINTOUT
C
C      IF (TRAP)
C      SKIP = .TRUE.
C      TRAP = .TRUE.
C      GO TO 20
C
C      MODE
C
C      200      CONTINUE
C

```



```

C          TRAP
C
C          IF (SKIP)
C              GO TO 20
C
C          WRITE(6, 1000)
C          WRITE(6, 225) (IDATA(I), I=1,N)
225      FORMAT(54 (1X, 15I7 / ), 1X, 15I7)
C          GO TO 20
C
C          ALL OTHER DATA
C
C          TRAP
C
C          300      CONTINUE
C          IF (SKIP)
C              GO TO 20
C
C          IF (LINCNT + NL .LE. 60)
C              GO TO 320
C          ELSE
C              WRITE(6, 35) TYPE
C              LINCNT = 3
C          320      WRITE(6, 1000)
C          WRITE(6, 350) (DATA(I), I=1,N)
350      FORMAT(54 (1X, 15F7.1 / ), 1X, 15F7.1)
C          LINCNT = LINCNT + NL
C          GO TO 20
C
C          RETURN POINTS FOR END OF FILE CONDITIONS
C
C          400      WRITE(6, 410)
C          410      FORMAT(' -NORMAL END OF FILE DETECTED')
C          GO TO 900
C
C          500      WRITE(6, 510)
C          510      FORMAT(' -ABNORMAL END OF FILE DETECTED')
C
C          900      RETURN
C
C          1000     FORMAT('0 ')
C
C          END

```

```

*****
*
*                               LUNAPLOT                               *
*
*****

```

```

C
C  ROUTINE TO PLOT LUNAR DATA
C  FROM FILE #1
C  AFTER INTERPOLATION BY THE COPY PROGRAM
C
C  SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
C  BELOW, AND ONE PLOT CARD AS DESCRIBED IN PLOTIT.
C
C  NAMELIST / FREQ /
C
C    IPRFQ - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
C           NO DEFAULT
C
C    XMIN  - MINIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 0.0
C
C    XMAX  - MAXIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 100.0
C
C    YMAX  - MAXIMUM (RELATIVE) DB VALUE TO BE PLOTTED, DEFAULT 67.5
C
C    ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
C           THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROS
C           CODES FOR THE COMPONENTS ARE:
C
C           ENDFIRE   BROADSIDE
C
C           RHO      212          211
C           PHI      222          221
C           ZED      232          231
C
C    FILT  - (LOGICAL) FILTERING REQUIRED, DEFAULT .FALSE.
C
C    COEFF  - FILTER COEFFICIENTS, OR ZEROS TO PAD THE ARRAY
C           TO 100 ELEMENTS (COEFFICIENTS SHOULD BE LEFT-JUSTIFIED
C           IN THE ARRAY, FOR DEFAULTS SEE DECLARATION OF COEFF
C
C    NCOEFF - NUMBER OF FILTER COEFFICIENTS, DEFAULT 11
C
C    REF   - RELATIVE DB VALUE AT WHICH A REFERENCE MARK IS TO BE
C           PLOTTED ON THE Y AXIS, DEFAULT 45.0
C
C

```

```

C      ABSREF - ABSOLUTE DB VALUE CORRESPONDING TO REF, DEFAULT 45.0
C
C      NOTES - UP TO 32 CHARACTERS OF ANNOTATION, NO DEFAULT
C
C
C
REAL*8      TITLE(2) / 2*'      ' /
REAL*4      VCO(1000), RANGE(1000), WORK(1000)
REAL*4      NOTES(8) / 8*'      ' /
REAL*4      XLIM(6,2) / 6*0.0, 16.0, 32.0, 60.0, 3*100.0 /
REAL*4      YMAXS(6) / 6*100.0 /
REAL*4      REF
REAL*4      XMIN, YMAX, YMAX
REAL*4      COEFF(100) /-.0023, .0041, .0445, .1239, .2078,
      .2440, .2078, .1239, .0445, .0041, -.0023,
      80*0.0
INTEGER*4    COMP(6) / 212, 222, 232, 211, 221, 231 /
INTEGER*4    IREFQ, ICOMP(6), CPERG
INTEGER*4    NCOEFF/ 11 /
LOGICAL*1    DECIDE(6,6) /36*.TRUE./, FILTRE(6)/6*.FALSE./
LOGICAL*1    BOTH, FILT
NAMELIST /PREQ/ IREFQ, XMIN, XMAX, YMAX, ICOMP, FILT, COEFF,
      NCOEFF, REF, NOTES, ABSREF
C
C      INITIALIZE RANGE ARRAY
C
C      DO 5 I=1,1000
      RANGE(I)=0.1*FLOAT(I-1)
5 CONTINUE
C
C      SKIP THE LABEL RECORD
C
C      READ(3)
      DO 500 I=1,6
C
C      INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES , IF ANY
C
C      XMIN=0.
      XMAX=100.
      FILT=.FALSE.
      YMAX=67.5
      REF=45.0
      ABSREF=45.0
      DO 100 J=1,6
100 ICOMP(J)=0
C
C      READ PLOTTING PARAMETERS
C

```

```

      READ(5,FRFO,END=520)
      IDX=IFRFO+1
      DO 120 J=1,6
      IC=COMP(J)
      DO 110 K=1,6
      IF(IC .EQ. ICOMP(K)) GO TO 120
110  CONTINUE
      DECIDE(IDX,J)=.FALSE.
120  CONTINUE
      IF(XMIN .GT. XLIM(IDX,1)) XLIM(IDX,1)=XMIN
      IF(XMAX .LT. XLIM(IDX,2)) XLIM(IDX,2)=XMAX
      IF(YMAX .LT. YMAXS(IDX)) YMAXS(IDX)=YMAX
      FILTRE(IDX)=FILT
500  CONTINUE
520  DX=0.
C
C      INITIALIZE PLOTTER
C
C      CALL PLINIT('OOP.JCR.LUNAR ')
C
C      LOOP THROUGH FREQUENCIES
C
C      DO 900 I=1,6
C
C      DETERMINE NUMBER OF CURVES PER GRAPH
C
      NA=0
      NB=0
      XMIN=XLIM(I,1)
      XMAX=XLIM(I,2)
      DO 550 J=1,3
      IF(DECIDE(I,J)) NA=NA+1
      IF(DECIDE(I,J+3)) NB=NB+1
550  CONTINUE
      BOTH=.FALSE.
      CPERG=NA
      IF(NA+NB .GT. 3) GO TO 560
      BOTH=.TRUE.
      CPERG=CPERG+NB
560  CONTINUE
      MST=IFIX(XMIN*10.0+1.5)
      MPT=IFIX(YMAX*10.0+0.5)
C
C      LOOP THROUGH COMPONENTS
C
      DO 580 J=1,6
      IF(DECIDE(I,J)) GO TO 563
      READ(3,END=999)

```

```

GO TO 580
563 READ(3,FND=999) TITLE(1), F,NST,NPT, (VCO(K),K=1,NPT)
DO 565 K=1,NPT
IF (VCO(K) .GT. YMAXS(I)) NST=K+1
565 CONTINUE
C
C COMPUTE FIRST POINT AND NUMBER OF POINTS TO BE PLOTTED
C
NST=MAX0(NST,MST)
NPT=MIN0(NPT,MPT)-NST+1
C
C FILTER IF REQUESTED
C
IF (FILTPE(I)) CALL FILTER(VCO(NST),NPT,COEFF,NCOEFF,WORK)
C
C PLOT THE CURVE
C
CALL DATPLT(TITLE,NOTES,CPERG,6.18,15.0,COMP(J),VCO(LST),
RANGE(NST),NPT,1,17.0,1.1,F,PEF,ABSREF)
IF (ROTH .OR. J .NE. 4) GO TO 580
CPERG=NB
580 CONTINUE
900 CONTINUE
C
C TERMINATE THE PLOT WHEN EOF IS DETECTED ON TAPE
C
999 CALL PLOTND
RETURN
END

```

```

*****
*
* LUNAPLT2
*
*****

```

```

C
C ROUTINE TO PLOT LUNAR DATA
C FROM FILE #2
C AFTER INTERPOLATION BY THE COPY PROGRAM
C
C SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
C BELOW, AND ONE PLTID CARD AS DESCRIBED IN PLINT.
C
C
C NAMELIST / PREQ /

```



```

      INTEGER*4 IFRQ, ICOMP(6), CPEFG
      INTEGER*4 NCOEFF/ 11 /
      LOGICAL*1 DECIDE(6,6) /36*.TRUE./, FILTRF(6)/6*.FALSE./
      LOGICAL*1 ROTH, FIIT
      NAMLIST /FRQ/ IFRQ, XMIN, XMAX, YMAX, ICOMP, FILT, COEFF,
      .           NCOEFF, REF, NOTES, ABSREF
C
C   INITIALIZE RANGE ARRAY
C
      DO 5 I=1,1000
      RANGE(I)=0.1*FLOAT(I-1)
5 CONTINUE
C
      DO 500 I=1,6
C
C   INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES , IF ANY
C
      XMIN=0.
      XMAX=100.
      FIIT=.FALSE.
      YMAX=67.5
      REF=45.0
      ABSREF=45.0
      DO 100 J=1,6
100 ICOMP(J)=0
C
C   READ PLOTTING PARAMETERS
C
      READ(5,FRQ,END=520)
      IDX=IFRQ+1
      DO 120 J=1,6
      IC=COMP(J)
      DO 110 K=1,6
      IF(IC .EQ. ICOMP(K)) GO TO 120
110 CONTINUE
      DECIDE(IDX,J)=.FALSE.
120 CONTINUE
      IF(XMIN .GT. XLIM(IDX,1)) XLIM(IDX,1)=XMIN
      IF(XMAX .LT. XLIM(IDX,2)) XLIM(IDX,2)=XMAX
      IF(YMAX .LT. YMAXS(IDX)) YMAXS(IDX)=YMAX
      FILTRF(IDX)=FIIT
500 CONTINUE
520 DY=0.
C
C   INITIALIZE PLOTTER
C
      CALL PLINIT('OQGP.JCR.LUNAR ')
C

```

```

C     LOOP THROUGH FREQUENCIES
C
C     DO 990 J=1,6
C
C     DETERMINE NUMBER OF CURVES PER GRAPH
C
      NA=0
      NB=0
      XMIN=XLIM(I,1)
      XMAX=XLIM(J,2)
      DO 550 J=1,3
      IF(DECIDE(I,J)) NA=NA+1
      IF(DECIDE(I,J+3)) NB=NB+1
550 CONTINUE
      BOTH=.FALSE.
      CPERG=NA
      IF(NA+NB .GT. 3) GO TO 560
      BOTH=.TRUE.
      CPERG=CPERG+NB
560 CONTINUE
      MST=IFIX(XMIN*10.0+1.5)
      MPT=IFIX(XMAX*10.0+0.5)
C
C     LOOP THROUGH COMPONENTS
C
      DO 580 J=1,6
      IF(DECIDE(I,J)) GO TO 563
      READ(3,END=999)
      GO TO 580
563 READ(3,END=999) TITLE(1), F,MST,NPT, (VCO(K),K=1,NPT)
      DO 565 K=1,NPT
      IF(VCO(K) .GT. YMAXS(I)) NST=K+1
565 CONTINUE
C
C     COMPUTE FIRST POINT AND NUMBER OF POINTS TO BE PLOTTED
C
      NST=MAX0(NST,MST)
      NPT=MIN0(NPT,MPT)-NST+1
C
C     FILTER IF REQUESTED
C
      IF(FILTRE(I)) CALL FILTER(VCO(NST),NPT,COEFF,NCOEFF,AGRN)
C
C     PLOT THE CURVE
C
      CALL DATPLT(TITLE,NOTES,CPERG,6.18,15.0,COMP(J),VCO(NST),
      . RANGE(NST),NPT,1,17.0,1.1,F,DEF,ANSDEF)
      IF(BOTH .OR. J .NE. 4) GO TO 580

```



```

      CPREG=NR
550 CONTINUE
900 CONTINUE
C
C   TERMINATE THE PLOT WHEN EOF IS DETECTED ON TAPE
C
999 CALL PLOTND
      RETURN
      END

```

```

*****
*
*                               LUNAPL3
*
*****

```

```

C
C
C   ROUTINE TO PLOT LUNAR SEP DATA FROM FILE #2;
C   NO INTERPOLATION;
C   VALUES WITH 510. M. <= RANGE <= 520. M.
C   ARE DELETED BEFORE PLOTTING.
C
C   THE RANGE AND V.C.O. DATA ARE ACCUMULATED IN ARRAY "DATA".
C   IORG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
C   MAY BE STORED.  IXX IS THE INDEX OF THE FIRST RANGE VALUE,
C   AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.
C
C   SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
C   BELOW, AND ONE PLTID CARD AS DESCRIBED IN PLINIT.
C
C   NAMELIST / CNTL /
C
C       IPREQ - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
C             NO DEFAULT
C
C       XMIN  - MINIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 0.0
C
C       XMAX  - MAXIMUM WAVELENGTH TO BE PLOTTED, DEFAULT 100.0
C
C       YMAX  - MAXIMUM (RELATIVE) DB VALUE TO BE PLOTTED, DEFABIT 67.5
C
C       ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OR ZEROS TO PAD
C             THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROS
C             CODES FOR THE COMPONENTS ARE:

```



```

      IDY=IFREQ+1
      DO 10120 J=1,5
      IC=COMP(J)
      DO 10110 K=1,5
      IF(IC.EQ.ICOMP(K)) GO TO 10120
10110 CONTINUE
      DECIDE(IDX,J)=.FALSE.
10120 CONTINUE
C
C     SET MIN AND MAX RANGE, AND MAX VCO FOR THIS COMPONENT
C
      IF(XMIN.GT.XLIM(IDX,1)) XLIM(IDX,1)=XMIN
      IF(YMAX.LT.XLIM(IDX,2)) XLIM(IDX,2)=YMAX
      IF(YMAX.LT.YMAXS(IDX)) YMAXS(IDX)=YMAX
      NA(IDX)=0
      NB(IDX)=0
C
C     DECIDE ON THE NUMBER OF CURVES PER GRAPH
C
      DO 10200 J=1,3
      IF(DECIDE(IDX,J)) NA(IDX)=NA(IDX)+1
      IF(DECIDE(IDX,J+3)) NB(IDX)=NB(IDX)+1
10200 CONTINUE
      BOTH(IDX)=.TRUE.
      IF(NA(IDX)+NB(IDX).GT.3) BOTH(IDX)=.FALSE.
      DO 10300 J=1,3
      IF(BOTH(IDX)) GO TO 10250
      CPERG(IDX,J)=NA(IDX)
      CPERG(IDX,J+3)=NB(IDX)
      GO TO 10300
10250 CPERG(IDX,J)=NA(IDX)+NB(IDX)
      CPERG(IDX,J+3)=CPERG(IDX,J)
10300 CONTINUE
10500 CONTINUE
10600 CONTINUE
C
C     INITIALIZE THE PLOTTER
C
      CALL PLINIT('OOGP.JCR.LUNAR ')
      N=386
C
C     READ THE LABEL RECORD
C
      CALL LUNIN2(DATA, IDATA, 8980, 8990)
      WRITE(6, 3000) TYPE
      WRITE(6, 1000) (IDATA(I), I=1, 297)
C
C     INITIALIZE THE STACK BEFORE READING RANGE DATA

```

```

C
10 IORG=1
   M=0
   L=0
20 IF (IORG+N .GT. 12000) GO TO 970
   CALL IUNIN2 (DATA (IORG), IDATA, 8980, 8990)
   IF (ITYPE (1) .LE. 5) GO TO 20

C
C
   IF (ITYPE (2) .EQ. 6) GO TO 60

C
C
   ACCUMULATE RANGE BLOCKS

   IF (FIRST) IXX=IORG
   IORG=IORG+N
   M=M+1
   IF (.NOT. LAST) GO TO 20
   XMTOWL=FREQ (ITYPE (1) - 5) / 299.7925
   NPTSIN=M*N
   NXBGAP=0
   NXAGAP=0

C
C
   FIND RANGE VALUES TO BE OMITTED

C
   DO 45 I=1, NPTSIN
   IF (DATA (I) .GE. 510.0) GO TO 50
   NXBGAP=NXBGAP+1
   DATA (I) = XMTOWL * (DATA (I) + 3.0)
45 CONTINUE
50 INEXT=NXBGAP+1
   ISTART=INEXT

C
C
   DELETE VALUES BY COMPRESSING THE ARRAY

C
   DO 55 I=ISTART, NPTSIN
   IF (DATA (I) .LE. 520.0) GO TO 55
   DATA (INEXT) = XMTOWL * (DATA (I) + 3.0)
   NXAGAP=NXAGAP+1
   INEXT=INEXT+1
55 CONTINUE

C
C
   RESET THE ORIGIN FOR VCO DATA, AND ZERO THE COMPONENT COUNTER

C
   IORG=NXBGAP+NXAGAP+1
   NCOMP=0
   GO TO 20

C
C

```

```

C      ACCUMULATE VCO BLOCKS
C
C
60  IF (FIRST) IXV=IORG
    IORG=IORG+N
    L=L+1
    IF (I .LT. M) GO TO 20
    IDX=ITYPE(1)-5
    NCOMP=NCOMP+1
    IF (.NOT. DECIDE (IDX, NCOMP)) GO TO 150
C
C      FIND THE LAST VALUE BEFORE THE GAP, AND ADJUST DB VALUES
C      TO A RELATIVE SCALE
C
    IY=IXY+NXBGAP-1
    DO 70 I=IXY, IY
70  CONTINUE
    INEXT=IXY+NXBGAP
    ISTART=IORG-NXAGAP
    IEND=IORG-1
C
C      COMPRESS THE VCO DATA, ADJUSTING
C      TO RELATIVE SCALE IN THE PROCESS
C
    DO 80 I=ISTART, IEND
    DATA (INEXT)=DATA (I) +135.0
    INEXT=INEXT+1
80  CONTINUE
    NSTX=IXX
    NX=IXX
    NPLOT=IXX+NXBGAP+NXAGAP-1
    IY=IXY
C
C      OMIT VALUES OUTSIDE THE OUTER BOUNDS
C
    DO 90 IX=NX, NPLOT
    IF (DATA (IX) .LT. XLIM (IDX, 1) .OR. DATA (IX) .GT. YMAXS (IDX))
      NSTX = IX + 1
    IY=IY+1
90  CONTINUE
    NSTY=IXY+NSTX-IXX
    IX=NPLOT
    NX=NPLOT
    DO 100 I=NSTX, NX
    IF (DATA (IX) .GT. XLIM (IDX, 2)) NPLOT=NPLOT-1
    IX=IX-1
100 CONTINUE
    NPTS=NPLOT-NSTX+1

```

```

C
C   PLOT THE CURVE
C
C   CALL DATPLT(TYPE,NOTES,CPERG(IDX,NCOMP),6.18,15.0,COMP(NCOMP),
.       DATA(NSTY),DATA(NSTX),NPTS,1,17.,1.1,FREQ(IDX),
.       REF,ABSREF)
C
C
C   IF THIS WAS THE SIXTH COMPONENT, GET A NEW RANGE ARRAY, OTHERWISE
C   GET A VCO ARRAY FOR THE NEXT COMPONENT
C
C
C   150 IF(LAST) GO TO 10
C       IORG=IXY
C       L=0
C       GO TO 20
C
C   STACK ARRAY TOO SMALL
C
C   970 WRITE(6,7000)
C       GO TO 999
C
C   NORMAL COMPLETION
C
C   980 WRITE(6,5000)
C       GO TO 999
C
C   PREMATURE END OF FILE ("TAPF" INPUT)
C
C   990 WRITE(6,6000) TYPE
C   999 CALL PLOTND
C       RETURN
C
C
C   1000 FORMAT(27(1X,11A4/))
C   2000 FORMAT('0',2A8,' RECORD SKIPPED')
C   3000 FORMAT('0',2A8,' RECORD READ')
C   4000 FORMAT('1 LABEL="',A8,'" / 'OFREQ.= ',F5.1,' MHZ.' /
.           '0FIRST POINT=',I4/ '0# OF POINTS=',I4/
.           '0',10F10.3/99(1X,10F10.3/))
C   5000 FORMAT('ONORMAL END OF JOB')
C   6000 FORMAT('OEND OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
.           2A8,' RECORD')
C   7000 FORMAT('-*** INSUFFICIENT SPACE ON STACK ***')
C
C
C   END

```

 *
 *
 *
 *
 *

LUNAPLT4

ROUTINE TO PLOT SEP DATA THROUGH THE TURN AT EP-4 VS. RECORD NUMBER

THE RANGE AND V.C.C. DATA ARE ACCUMULATED IN ARRAY "DATA".
 IOPG IS THE INDEX OF THE NEXT FREE LOCATION INTO WHICH DATA
 MAY BE STORED. IXX IS THE INDEX OF THE FIRST RANGE VALUE,
 AND IXY IS THE INDEX OF THE FIRST V.C.O. VALUE.

SEVEN NAMELIST CARDS ARE REQUIRED AS INPUT, SIX AS DESCRIBED
 BELOW, AND ONE PLTID CARD AS DESCRIBED IN PLTID.

NAMELIST / CNTL /

IFREQ - FREQUENCY INDICATOR (BASE 2 LOG OF FREQUENCY)
 NO DEFAULT

ICOMP - ARRAY OF COMPONENTS TO BE PLOTTED, OF ZEROES TO PAD
 THE ARRAY OUT TO 6 ELEMENTS, DEFAULT 6 ZEROES
 CODES FOR THE COMPONENTS ARE:

	ENDFIRE	BROADSIDE
RHO	212	211
PHI	222	221
ZED	232	231

REAL*8 TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
 REAL*8 PROGNM(2) / 'OQGP.JCR', 'GAP' /
 REAL*4 DATA(12000), RANGE(1000), VCO(1000)
 REAL*4 FREQ(6) / 1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
 INTEGER*4 IDATA(400)
 INTEGER*2 ITYPE(2)
 LOGICAL*4 FIRST, LAST
 INTEGER*2 ICOMP(6)
 INTEGER*2 COMP(6) / 212, 222, 232, 211, 221, 231 /
 LOGICAL*1 DECIDE(6,6) / 36 * .TRUE. /
 NAMELIST / CNTL / IFREQ, ICOMP

```

EQUIVALENC (DATA(1), IDATA(1))
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
      ITYPE, N, FIRST, LAST

```

C
C

```
DO 10500 I=1,6
```

C
C
C

```
NO COMPONENTS PLOTTED UNLESS REQUESTED
```

C
C

```
DO 10100 J=1,6
```

10100

```
ICOMP(J)=0
```

C
C
C

```
READ FREQUENCY INDICATOR AND COMPONENTS TO BE PLOTTED
```

```
READ(5,CNTL,END=10600)
```

```
IDX=JFREQ+1
```

```
DO 10120 J=1,6
```

```
IC=COMP(J)
```

```
DO 10110 K=1,6
```

```
IF(IC.EQ.ICOMP(K)) GO TO 10120
```

10110

```
CONTINUE
```

```
DECIDE(IDX,J)=.FALSE.
```

10120

```
CONTINUE
```

10500

```
CONTINUE
```

10600

```
CONTINUE
```

C
C
C

```
INITIALIZE THE PLOTTER
```

```
CALL PLINIT(PROGMM)
```

```
N=386
```

C
C
C

```
SKIP THE LABEL BLOCK
```

```
CALL LUNIN2(DATA, IDATA, 8980, 8990)
```

```
CALL LUNIN3(DATA, IDATA, 8980, 8990)
```

C
C
C

```
INITIALIZE THE STACK
```

```
10 IORG=1
```

```
M=0
```

```
I=0
```

```
20 IF(IORG+N .GT. 12000) GO TO 970
```

```
CALL LUNIN2(DATA(IORG), IDATA, 8980, 8990)
```

```
IF(ITYPE(2) .NE. 5)
```

```
CALL LUNIN3(DATA(IORG), IDATA, 8980, 8990)
```

```
IF(ITYPE(1) .GE. 6) GO TO 40
```

```
GO TO 20
```

C


```

C      ACCUMULATE RANGE BLOCKS
C
40  CONTINUE
    IF (ITYPE(2) .EQ. 6) GO TO 60
    IF (FIRST) IXX=IORG
    IORG=IORG+N
    M=M+1
    IF (.NOT. LAST) GO TO 20
    NPTSIN=N*M
    IGX=IXX
    IGYEND=IXX
C
C      FIND THE GAP
C
    DO 50 I=IXX,NPTSIN
    IF (DATA(I) .LE. 400.) IGX=IGX+1
    IF (DATA(I) .LE. 535.) IGYEND = IGYEND + 1
50  CONTINUE
    NCOMP=0
    GO TO 20
C
C      ACCUMULATE VCO BLOCKS
C
60  IF (FIRST) IXY=IORG
    IORG=IORG+N
    L=L+1
    IF (L .LT. M) GO TO 20
    IGY=IXY+IGY-IXX
    IGYEND=IXY+IGYEND-IXX
    NPTS=IGYEND-IGX
    NCOMP=NCOMP+1
    IF (.NOT. DECIDE (ITYPE(1)-5,NCOMP)) GO TO 150
    YMIN=DATA (IGY)+135.
    YMAX=VMIN
    IY=IGY
C
C      ADJUST DB VALUES TO RELATIVE SCALE, AND FIND MINIMUM AND
C      MAXIMUM VCO THROUGH THE TURN
C
    DO 70 I=1,NPTS
    DATA (IY)=DATA (IY)+135.
    IF (DATA (IY) .LT. YMIN) YMIN=DATA (IY)
    IF (DATA (IY) .GT. YMAX) YMAX=DATA (IY)
    IY=IY+1
70  CONTINUE
C
C      PLOT THE POINTS
C

```

```

      CALL GAPLOT(BPFO (ITYPE(1)-5), DATA (IGX), DATA (IGY),
      .           NPTS, YMIN, YMAX, NCOMP)
C
C   IF THIS WAS THE SIXTH COMPONENT, GET NEW RANGE DATA; OTHERWISE
C   GET NEW VCO DATA
C
150  IF (LAST) GO TO 10
      IORG=IXY
      L=0
      GO TO 20
C
C   STACK ARRAY TOO SMALL
C
970  WRITE (6,7000)
      GO TO 999
C
C   NORMAL COMPLETION
C
980  WRITE (6, 5000)
      GO TO 999
C
C   PREMATURE END OF DATA FILE
C
990  WRITE (6, 6000) TYPE
999  CALL PLOTND
      RETURN
C
C
1000 FORMAT(27(1X,11A4/))
2000 FORMAT('0',2A8,' RECORD SKIPPED')
3000 FORMAT('0',2A8,' RECORD READ')
4000 FORMAT('1 LABEL="',A8,'" / '0 FREQ. = ',F5.1,' MHZ.' /
      .       '0 FIRST POINT = ',I4/ '0 # OF POINTS = ',I4/
      .       '0',10F10.3/99(1X,10F10.3/))
5000 FORMAT('0 NORMAL END OF JOB')
6000 FORMAT('0 END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
      .       2A8,' RECORD')
7000 FORMAT('0 *** INSUFFICIENT SPACE ON STACK ***')
C
C
      END

```



```

C           IN METRES.
C
C           REF      - RELATIVE DB VALUE AT WHICH A REFERENCE MARK IS TO BE
C                     PLOTTED ON THE Y AXIS, DEFAULT 45.0
C
C           ABSREF   - ABSOLUTE DB VALUE CORRESPONDING TO REF, DEFAULT 45.0
C
C           NOTES    - UP TO 32 CHARACTERS OF ANNOTATION, NO DEFAULT
C
C
REAL*8      TYPE(2), RUN, SITE, DIRECT, FORREV, TITLE(11)
REAL*4      DATA(12000), RANGE(1000), VCO(1000)
REAL*4      RANGE2(140, 6), TXOFF(140, 6, 3)
REAL*4      FREQ(6) /1.0, 2.1, 4.0, 8.1, 16.0, 32.1 /
INTEGER*4   IDATA(400)
INTEGER*4   NTXOFF(6, 3), NP(3)
INTEGER*2   ITYPE(2)
LOGICAL*4   FIRST, LAST
EQUIVALENCE (DATA(1), IDATA(1))
REAL*8      NOTES(4) /4*' '/
REAL*4      XLIM(6,2)
REAL*4      YMAXS(6)
INTEGER*4   ICOMP(6), NA(6), NR(6), CPREG(6,6)
INTEGER*4   COMP(6) /212,222,232,211,221,231/
INTEGER*4   NN(6)
LOGICAL*1   DECIDE(6,6) /36*.TRUE./, BOTH(6)
LOGICAL*1   VSWL / .TRUE. /
NAMLIST/CNTL/IFREQ,XMIN,XMAX,YMAX,ICOMP,REF,ABSREF,NOTES,VSWL
COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
           ITYPE, N, FIRST, LAST
C
C
DO 10500 I=1,6
C
C   INITIALIZE PLOTTING PARAMETERS TO DEFAULT VALUES , IF ANY
C
XMIN=0.0
XMAX=100.0
YMAX=67.5
REF=45.0
ABSREF=45.0
DO 10100 J=1,6
10100 ICOMP(J)=0
C
C   READ PLOTTING PARAMETERS
C
READ(5,CNTL,END=10600)
IDX=IFREQ+1

```

```

DO 10120 J=1,6
TC= COMP(J)
DO 10110 K=1,6
IF (TC.EQ. ICOMP(K)) GO TO 10120
10110 CONTINUE
DECIDE (IDX,J) = .FALSE.
10120 CONTINUE
C
C   SET MIN AND MAX RANGE, AND MAX VCO FOR THIS COMPONENT
C
VLIM (IDX,1) = XMIN
XLIM (IDX,2) = XMAX
YMAXS (IDX) = YMAX
NA (IDX) = 0
NB (IDX) = 0
C
C   DECIDE ON THE NUMBER OF CURVES PER GRAPH
C
DO 10200 J=1,3
IF (DECIDE (IDX,J )) NA (IDX) = NA (IDX) + 1
IF (DECIDE (IDX,J+3)) NB (IDX) = NB (IDX) + 1
10200 CONTINUE
BOTH (IDX) = .TRUE.
IF (NA (IDX) + NB (IDX) .GT. 3) BOTH (IDX) = .FALSE.
DO 10300 J=1,3
IF (BOTH (IDX)) GO TO 10250
CPERG (IDX,J ) = NA (IDX)
CPERG (IDX,J+3) = NB (IDX)
GO TO 10300
10250 CPERG (IDX,J ) = NA (IDX) + NB (IDX)
CPERG (IDX,J+3) = CPERG (IDX,J)
10300 CONTINUE
10500 CONTINUE
10600 CONTINUE
XSCALE = 10.
IF (VSWL) XSCALE = 6.18
C
C   READ TRANSMITTER-OFF DATA AND THE ASSOCIATED RANGE VALUES.
C
READ (3)
READ (3)
READ (3)
READ (3) TXOFF, NTXOFF
READ (3) RANGE?, NR
C
C   REMOVE DATA FOR 510 M. <= RANGE <= 520 M.
C
DO 20500 IFR = 1, 6

```

```

INEXT = 1
N = NR((IFR - 1) / 2 + 1)
XMTOWL = FBEC(IFR) / 239.7925
IF(.NOT. VSWL) XMTOWL = 1.
DO 20400 I = 1, N
  IF(RANGE2(I, IFR) .LT. XLIM(IFR, 1) / XMTOWL - 3.) GO TO 20400
  IF(RANGE2(I, IFR) .GT. XLIM(IFR, 2) / XMTOWL - 3.) GO TO 20500
  IF(RANGE2(I, IFR) .GE. 510.) GO TO 20200
  RANGE2(INEXT, IFR) = (RANGE2(I, IFR) + 3.) * XMTOWL
  DO 20100 J = 1, 3
    TXOFF(INEXT, IFR, J) = TXOFF(I, IFR, J) + 135.
20100 CONTINUE
  GO TO 20350
20200 IF(RANGE2(I, IFR) .LE. 520.) GO TO 20400
  RANGE2(INEXT, IFR) = (RANGE2(I, IFR) + 3.) * XMTOWL
  DO 20300 J = 1, 3
    TXOFF(INEXT, IFR, J) = TXOFF(I, IFR, J) + 135.
20300 CONTINUE
20350 NR(IFR) = INEXT
  INEXT = INEXT + 1
20400 CONTINUE
20500 CONTINUE
C
C   INITIALIZE THE PLOTTER
C
  CALL FINIT('DOGP.JCR.LUNAR ')
  N=386
C
C   READ THE LABEL RECORD
C
  CALL LUNIN2(DATA, IDATA, 8980, 8990)
  WRITE(6, 3000) TYPE
  WRITE(6, 1000) (IDATA(I), I=1, 297)
C
C   INITIALIZE THE STACK BEFORE READING RANGE DATA
C
  10 IORG=1
  M=0
  L=0
  20 IF(IORG+N .GT. 12000) GO TO 970
  CALL LUNIN2(DATA(IORG), IDATA, 8980, 8990)
  IF(ITYPE(1) .LE. 5) GO TO 20
C
C
  IF(ITYPE(2) .EQ. 6) GO TO 60
C
C   ACCUMULATE RANGE BLOCKS
C

```

```

IF(FIRST) IXX=IORG
IORG=IORG+N
M=M+1
IF(.NOT. LAST) GO TO 20
XMTOWL=FREQ(ITYPE(1)-5)/299.7925
IF(.NOT. VSWL) XMTOWL = 1.
NPTSIN=M*N
NXBGAP=0
NXAGAP=0

```

```

C
C FIND RANGE VALUES TO BE OMITTED
C

```

```

DO 45 I=1,NPTSIN
IF(DATA(I).GE.510.0) GO TO 50
NXBGAP=NXBGAP+1
DATA(J)=XMTOWL*(DATA(I)+3.0)
45 CONTINUE
50 INEXT=NXBGAP+1
ISTART=INEXT

```

```

C
C DELETE VALUES BY COMPRESSING THE ARRAY
C

```

```

DO 55 I=ISTART,NPTSIN
IF(DATA(I).LE.520.0) GO TO 55
DATA(INEXT)=XMTOWL*(DATA(I)+3.0)
NXAGAP=NXAGAP+1
INEXT=INEXT+1
55 CONTINUE

```

```

C
C RESET THE ORIGIN FOR VCO DATA, AND ZERO THE COMPONENT COUNTER
C

```

```

IORG=NXBGAP+NXAGAP+1
NCOMP=0
GO TO 20

```

```

C
C ACCUMULATE VCO BLOCKS
C

```

```

60 IF(FIRST) IXY=IORG
IORG=IORG+N
L=L+1
IF(L.LT. M) GO TO 20
IDX=ITYPE(1)-5
NCOMP=NCOMP+1
IF(.NOT.DECIDE(IDX,NCOMP)) GO TO 150

```

```

C
C FIND THE LAST VALUE BEFORE THE GAP, AND ADJUST DR VALUES

```

```

C      TO A RELATIVE SCALE
C
      IY=IXY+NXBGAP-1
      DO 70 I=IXY, IY
      DATA (I) =DATA (I) +135.0
70 CONTINUE
      INEXT=IXY+NXBGAP
      ISTART=IOPG-NXAGAP
      IEND=IOPG-1

C
C      COMPRESS THE VCO DATA, ADJUSTING
C      TO RELATIVE SCALE IN THE PROCESS
C
      DO 80 I=ISTART, IEND
      DATA (INEXT) =DATA (I) +135.0
      INEXT=INEXT+1
80 CONTINUE
      NSTX=IXX
      NX=IXX
      NPLOT=IXX+NXBGAP+NXAGAP-1
      IY=IXY

C
C      OMIT VALUES OUTSIDE THE OUTER BOUNDS
C
      DO 90 IX=NX, NPLOT
      IF (DATA (IX) .LT. XLIM (IDX, 1) .OR. DATA (IY) .GT. YFAXS (IDX) )
      .       NSTX = IX + 1
      IY=IY+1
90 CONTINUE
      NSTY=IXY+NSTX-IXX
      IX=NPLOT
      NX=NPLOT
      DO 100 I=NSTX, NX
      IF (DATA (IX) .GT. XLIM (IDX, 2) ) NPLOT=NPLOT-1
      IX=IX-1
100 CONTINUE
      NPTS=NPLOT-NSTX+1

C
C      PLOT THE CURVE
C
      CALL DATPLT (TYPE, NOTES, CPERG (IDX, NCOMP), XSCALE, 15., COMI (NCOMP),
      .           DATA (NSTY), DATA (NSTX), NPTS, 1, 17., 1.1, FREQ (IDX),
      .           REF, ABSREF)

C
C      PLOT THE BASELINE.
C
      CALL PASEL (TXOFF (1, IDX, MOD (NCOMP - 1, 3) + 1),
      .           RANGE2 (1, IDX), NN (IDX) )

```



```

C
C
C   IF THIS WAS THE SIXTH COMPONENT, GET A NEW RANGE ARRAY, OTHERWISE
C   GET A VCO ARRAY FOR THE NEXT COMPONENT
C
C
C 150 IF(LAST) GO TO 10
      IORG=IXY
      L=0
      GO TO 20
C
C   STACK ARRAY TOO SMALL
C
C 970 WRITE(6,7000)
      GO TO 999
C
C   NORMAL COMPLETION
C
C 980 WRITE(6, 5000)
      GO TO 999
C
C   PREMATURE END OF FILE ("TAPE" INPUT)
C
C 990 WRITE(6, 6000) TYPE
C 999 CALL PLOTND
      RETURN
C
C
C 1000 FORMAT(27(1X,11A4/))
C 2000 FORMAT('0',2A8,' RECORD SKIPPED')
C 3000 FORMAT('0',2A8,' RECORD READ')
C 4000 FORMAT('1LABEL="'',A8,'" / '0FREQ.= ',F5.1,' MHZ.' /
      .      '0FIRST POINT=',I4/ '0# OF POINTS=',I4/
      .      '0',10F10.3/99(1X,10F10.3/))
C 5000 FORMAT('0NORMAL END OF JOB')
C 6000 FORMAT('0END OF FILE OCCURRED WHILE ATTEMPTING TO READ ',
      .      2A8,' RECORD')
C 7000 FORMAT('0-*** INSUFFICIENT SPACE ON STACK ***')
C
C
C   END

```



```

C
C   RECORD COUNTERS
C
C   INTEGER*4 IBLK /17/, IREC /78/
C
C   BEGIN EXECUTABLE CODE
C
C   RESET RECORD COUNTER
C
FIRST=.FALSE.
  IREC = IREC + 1
  IF (IREC .LE. TIDX(1,IBLK))
    GO TO 10
  ELSE
    IREC = 1
    FIRST=.TRUE.
    IBLK = IBLK + 1
    IF (IBLK .GT. 17)
      IBLK = 1
      ITYPE(1) = TIDX(2, IBLK)
      ITYPE(2) = TIDX(3, IBLK)
      TYPE(1) = TYPE1(ITYPE(1))
      TYPE(2) = TYPE2(ITYPE(2))
C
C   SELECT APPROPRIATE RECORD TYPE
C
10  IF (IBLK - 2) 100, 200, 300
C
C   HEADER RECORD
C
100  READ(4, 1000, END=999) PUN, SITE, DIRECT, FORREV, TITLE, N
    GO TO 999
C
C   MODE RECORD
C
200  READ(4, 2000, END=995) (IDATA(I), I=1, N)
    GO TO 999
C
C   ALL OTHER TYPES
C
300  READ(4, 3000, END=995) (DATA(I), I = 1, N)
    GO TO 999
C
C   END OF FILE CONDITIONS
C
C   PREDICTABLE
C   ( LABEL RECORD EXPECTED )
C   ( => BEGINNING OF A NEW RUN )

```



```

C      .          '16 MHZ. ', '32 MHZ. ' /
C      REAL*8 TYPE2(6) / '          ', 'URE          ', 'TEP OFF ',
C      .          'ION          ', 'RANGE          ', 'V. C. O.' /
C      REAL*8 RUN, SITE, DIRECT, FORREV, TITLE(11), TYPE(2)
C      INDICES TO TYPE ARRAYS
C      INTEGER*2 TIDX(3,17) /  1,  1,  1,  1,  2,  1,  1,  3,  2,
C      .          6,  4,  3,  6,  5,  4,  1,  6,  5,
C      .          6,  6,  6,  1,  7,  5,  6,  7,  6,
C      .          2,  8,  5, 12,  8,  6,  4,  9,  5,
C      .          24,  9,  6,  8, 10,  5, 48, 10,  6,
C      .          13, 11,  5, 78, 11,  6 /
C      LOGICAL*4 FIRST, LAST
C      TYPE CODE RETURNED
C      INTEGER*2 ITYPE(2)
C      COMMON BLOCK FOR RETURNED DATA
C      COMMON /LUNDAT/ TITLE, RUN, SITE, DIRECT, FORREV, TYPE,
C      .          ITYPE, N, FIRST, LAST
C      RECORD COUNTERS
C      INTEGER*4 IBLK /17/, IREC /78/
C      BEGIN EXECUTABLE CODE
C      RESET RECORD COUNTER
C      FIRST=.FALSE.
C      IREC = IREC + 1
C      IF (IREC .LE. TIDX(1,IBLK))
C      .      GO TO 10
C      ELSE
C      .      IREC = 1
C      .      FIRST=.TRUE.
C      .      IBLK = IBLK + 1
C      .      IF (IBLK .GT. 17)
C      .          IBLK = 1
C      .      ITYPE(1) = TIDX(2, IBLK)
C      .      ITYPE(2) = TIDX(3, IBLK)

```

```

                TYPE(1) = TYPE1(ITYPE(1))
                TYPE(2) = TYPE2(ITYPE(2))
C
C     SELECT APPROPRIATE RECORD TYPE
C
C     10  IF(IBLK - 2) 100, 200, 300
C
C         HEADER RECORD
C
C     100  READ(4,1000,END=990) (IDATA(I), I=1, 297)
C         GO TO 999
C
C         MODE RECORD
C
C     200  READ(4, 2000, END=995) (IDATA(I), I=1, N)
C         GO TO 999
C
C         ALL OTHER TYPES
C
C     300  READ(4, 3000, END=995) (DATA(I), I = 1, N)
C         GO TO 999
C
C     END OF FILE CONDITIONS
C
C     PREDICTABLE
C     ( LABEL RECORD EXPECTED )
C     ( => BEGINNING OF A NEW RUN )
C
C     990  RETURN 1
C
C     UNEXPECTED
C     ( NON-LABEL RECORD EXPECTED )
C     ( => MIDDLE OF A RUN )
C
C     995  WRITE(6, 4000) TYPE, IREC
C         RETURN 2
C
C     RETURN DATA
C
C     999  LAST=.FALSE.
C         IF(IREC .EQ. TIDX(1,IBLK)) LAST = .TRUE.
C         RETURN
C
C
C     1000 FORMAT(27(11A4))
C     2000 FORMAT(5 (200I6) )
C     3000 FORMAT(5 (200F6.1) )
C     4000 FORMAT('0*** END OF FILE FOUND WHILE ATTEMPTING TO READ "')

```

2A8, ' " RECORD ', J3)

C
C

END

```
*****
*
*                               LUNIN3
*
*****
```

SUBROUTINE LUNIN3(DATA, IDATA, *, *)

C
C
C
C
C

READ LUNAR DATA TAPE

TAPE DATA ARRAYS

REAL*4 DATA(1)
INTEGER*4 IDATA(1)

C
C
C

CHARACTER DATA

```
REAL*8 TYPE1(11) / 'LABEL   ', 'MODE    ', 'TEMPERAT',
.                  'TRANSMIT', 'CALIBRAT', ' 1 MHZ. ',
.                  ' 2 MHZ. ', ' 4 MHZ. ', ' 8 MHZ. ',
.                  '16 MHZ. ', '32 MHZ. '      /
```

C

```
REAL*8 TYPE2(6) / '          ', 'TYPE    ', 'TER OFF ',
.                 'ION     ', 'RANGE   ', 'V. C. O.' /
```

C

REAL*8 RUN, SITE, DIRECT, FORREV, TITLE(11), TYPE(2)

C
C
C

INDICES TO TYPE ARRAYS

```
INTEGER*2 TIDX(3,17) /  1,  1,  1,  1,  2,  1,  1,  3,  2,
.                       6,  4,  3,  6,  5,  4,  1,  6,  5,
.                       6,  6,  6,  1,  7,  5,  6,  7,  6,
.                       2,  8,  5, 12,  8,  6,  4,  9,  5,
.                       24,  9,  6,  8, 10,  5, 48, 10,  6,
.                       13, 11,  5, 78, 11,  6      /
```

C

LOGICAL*4 FIRST, LAST

C
C
C
C
C

TYPE CODE RETURNED

```

      INTEGER*2 ITYPE(2)
C
C      COMMON BLOCK FOR RETURNED DATA
C
      COMMON /LUNDAT/ TITLE,  RUN,   SITE,  DIRECT,  FOREEV,  TYPE,
      .           ITYPE,  N,     FIRST,  LAST
C
      RECOFD COUNTERS
C
      INTEGER*4 IBLK /17/,  IREC /78/
C
      BEGIN EXECUTABLE CODE
C
      RESET RECORD COUNTER
C
      FIRST=.FALSE.
      IREC = IREC + 1
      IF(IREC .LE. TIDX(1,IBLK))
      .   GO TO 10
C
      ELSE
          IREC = 1
          FIRST=.TRUE.
          IBLK = IBLK + 1
          IF(TIDX(3,IBLK) .EQ. 5)
          .   IBLK = IBLK + 1
          IF(IBLK .GT. 17)
          .   IBLK = 1
          ITYPE(1) = TIDX(2, IBLK)
          ITYPE(2) = TIDX(3, IBLK)
          TYPE(1) = TYPE1(ITYPE(1))
          TYPE(2) = TYPE2(ITYPE(2))
C
      SELECT APPROPRIATE RECORD TYPE
C
      10  IF( (IBLK - 2) 100, 200, 300
C
          HEADER RECORD
C
      100  READ(2,1000,END=990) (IDATA(I), I=1, 297)
          GO TO 999
C
          MCDF RECORD
C
      200  READ(2,2000,END=995) (IDATA(I), I=1, N)
          GO TO 999
C
          ALL OTHER TYPES
C

```



```

300      READ(2,3000,END=995) ( DATA(I), I=1, N)
      GO TO 999
C
C      END OF FILE CONDITIONS
C
C      PREDICTABLE
C      ( LABEL RECORD EXPECTED      )
C      ( => BEGINNING OF A NEW RUN )
C
990      RETURN 1
C
C      UNEXPECTED
C      ( NON-LABEL RECORD EXPECTED )
C      ( => MIDDLE OF A RUN      )
C
995      WRITE(6, 4000) TYPE, IREC
      RETURN 2
C
C      RETURN DATA
C
999      LAST=.FALSE.
      IF(IREC .EQ. TIDX(1,IBLK)) LAST = .TRUE.
      RETURN
C
C
1000     FORMAT(27(11A4))
2000     FORMAT(5 (200I6) )
3000     FORMAT(5 (200F6.1) )
4000     FORMAT('0***  END OF FILE FOUND WHILE ATTEMPTING TO READ "',
      .
      2A8, '" RECORD ',I3)
C
C
      END

```

```

*****
*
*
*
*****

```

```

      FUNCTION ODCINT(T)
C
      LOGICAL*1 FIRST / .TRUE. /
      REAL*4 TIME(500) , ODC(500)
C
C

```

```

IF(.NOT. FIRST) GO TO 100
FIRST = .FALSE.
N = 0
20 READ(5, 1000, END = 30) TT, ORF, OLP
N = N + 1
TIME(N) = TT
ODC(N) = 0.5 * (ORF + OLR)
GO TO 20
C
30 NN = N / 5
IF(MOD(N, 5) .NE. 0) NN = NN + 1
DO 50 I = 1, NN
IF(MOD(I - 1, 50) .EQ. 0) WRITE(6, 2000)
JJ = 4 * NN + I
IF(JJ .GT. N) JJ = JJ - NN
WRITE(6, 3000) (TIME(I), ODC(I), IJ = I, JJ, NN)
50 CONTINUE
WRITE(6, 4000)
C
100 IF(T .LE. TIME(1)) GO TO 300
IF(T .GE. TIME(N)) GO TO 400
DO 200 I = 2, N
IF(T .GE. TIME(I)) GO TO 200
ODCINT = ODC(I - 1) + (ODC(I) - ODC(I - 1))
.           * (T - TIME(I - 1))
.           / (TIME(I) - TIME(I - 1))
GO TO 500
200 CONTINUE
C
300 ODCINT = ODC(1)
GO TO 500
C
400 ODCINT = ODC(N)
C
500 RETURN
C
1000 FORMAT(3F10.0)
2000 FORMAT('1NAVIGATION DATA ' / '0', 6X, 'TIME OD. CNT.',
.         4(14X, 'TIME OD. CNT.') / '0 ')
3000 FORMAT(1X, 2F10.1, 4(8X, 2F10.1))
4000 FORMAT('- ')
C
END

```

```
*****
*
*                               PLINIT
*
*****
```

```

SUBROUTINE PLINIT(NAME)
C PLOTTER INITIALIZATION AND SETUP
  REAL*4 NAME(4),INIT/' .JCR' /
  LOGICAL ZIP/.TRUE./
  REAL*8 CODE/' EGS1410' /, SETUP(5)/5*' ' /, BLANK/' ' /
  DATA LIMIT,PLTLEN,PAGWID/90, 20., 11. /
  NAMLIST /PLTID/ INIT,CODE,SETUP,LIMIT,PLTLEN,PAGWID,ZIP
  NAMLIST /PLECHO/ LIMIT,PLTLEN,PAGWID,ZIP
  COMMON /PLTCOM/IT,L,II,IL
  IT=12
  ISET=0
  READ(5,PLTID)
  NAME(4)=INIT
  IF (SETUP(1).NE.BLANK) LSET=40
  IZIP=-1822
  IF (ZIP) IZIP=-IZIP
  CALL PLTSET(LIMIT,SETUP,LSET)
  CALL PLOTST (NAME,16,CODE,IZIP)
  WRITE (6,PLECHO)
  CALL PLTPAG(PAGWID)
  CALL PLTXMX(PLTLEN)
  RETURN
END
SUBROUTINE SYMBOL (X,Y,Z,IBCD,ANGLE,N)
C..***** NOTE *****..USE THIS SUBROUTINE ONLY IN PRODUCTION; REMOVE FOR
CALL SYMBOL (X,Y,Z,IBCD,ANGLE,N)
RETURN
END

```

```
*****
*
*                               RTPLOT
*
*****
```

```

SUBROUTINE RTPLOT
C
  REAL*4 SR(3088), ST(3088), VR(2565), VT(2565)
  COMMON / BLOCK / SR, ST, VR, VT, SCALE

```

```

C
C   INITIALIZE THE PLOTTER SOFTWARE AND LOCAL VARIABLES.
C   (TS IS REQUIRED LATER, FOR DRAWING THE TIME AXIS.)
C
CALL PLINIT('OQGP.RANGES.  ')
SC=1. / SCALE
TORG = AINT(AMAX1(ST(3088), VT(2565)) * SC) + 1.
T = AINT(AMIN1(ST(1), VT(1)) * SC)
TS = T

C
C   DRAW THE RANGE AXIS.
C
CALL PLOT(0., TORG - T, 3)
P = AINT(AMAX1(SR(3088), VR(2565)) * SC) + 1.
CALL SYMBOL(R, TORG - T, .07, 6, -90., -2)
N = IFIX(R) - 1
X = R

C
C   AND LABEL IT.
C
DO 20 I = 1, N
  X = X - 1.
  CALL SYMBOL(X, TORG - T, .07, 13, 0., -1)
  CALL NUMBER(X - .14, TORG - T + .07, .07, X * SCALE, 0., -1)
20 CONTINUE
CALL SYMBOL(R * .6, TORG - T + .2, .14, 'RANGE (METRES)', 0., 14)

C
C   IDENTIFY THE TWO PLOTS: SEP IS A SOLID LINE; VLBI IS SOLID
C   AND MARKED BY A SYMBOL AT EVERY 100'TH POINT
C
CALL PLOT(R - 2., TORG - T - 2.4, 3)
CALL PLOT(R - 2., TORG - T - 3.4, 2)
CALL SYMBOL(R - 2.07, TORG - T - 3.6, .14, 'SEP', -90., 3)
CALL SYMBOL(R - 2.2, TORG - T - 2.4, .03, 0, 0., -1)
CALL SYMBOL(R - 2.2, TORG - T - 2.9, .03, 0, 0., -2)
CALL SYMBOL(R - 2.2, TORG - T - 3.4, .03, 0, 0., -2)
CALL SYMBOL(R - 2.27, TORG - T - 3.6, .14, 'VLBI', -90., 4)

C
C   LABEL THE TIME AXIS.
C
CALL SYMBOL(-.34, (TORG - T) * .5, .14, 'TIME (SECONDS)', -90., 14)
N = IFIX(TORG - T) - 1
DO 40 I = 1, N
  T = T + 1.
  CALL SYMBOL(0., TORG - T, .07, 13, 90., -1)
  CALL NUMBER(-.14, TORG - T + .14, .07, T * SCALE, -90., -1)
40 CONTINUE
C

```

```

C      AND THEN DRAW IT.
C
      CALL SYMBOL(0., TORQ - T - 1., .07, 6, 180., -1)
      CALL PLOT(0., TORQ - TS, 2)
C
C      MOVE TO THE FIRST SEP POINT, AND THEN DRAW THE LINE.
C
      CALL PLOT(SR(1) * SC, TORQ - ST(1) * SC, 3)
      DO 60 I = 2, 3088
        CALL PLOT(SR(I) * SC, TORQ - ST(I) * SC, 2)
60    CONTINUE
C
C      PLOT THE VLBI DATA WITH A SYMBOL AT EVERY 100TH POINT.
C
      CALL SYMBOL(VR(1) * SC, TORQ - VT(1) * SC, .03, 0, 0., -1)
      DO 80 I = 2, 2564
        IF(MOD(I, 100) .EQ. 1) GO TO 70
        CALL PLOT(VR(I) * SC, TORQ - VT(I) * SC, 2)
        GO TO 80
70    CALL SYMBOL(VR(I) * SC, TORQ - VT(I) * SC, .03, 0, 0., -2)
80    CONTINUE
      CALL SYMBOL(VR(2565) * SC, TORQ - VT(2565) * SC, .03, 0, 0., -2)
      CALL PLOTND
      RETURN
      END

```

```

*****
*
*                      SEPLOT
*
*****

```

```

SUBROUTINE SEPLOT(TITLE,NOTES,CPERG,XSCALE,YSCALE,COMP,H,P,NIN,
*  INDEX,D,K1,LT1,K2,LT2,SITE,PUN,FREQ,REF,ABSREF)

```

```

C
C      PLOT OF EITHER THEORETICAL OR EXPERIMENTAL SEP DATA.
C      WRITTEN BY J.J.PROCTOR, SPRING 1973.  UNIVERSITY OF TORONTO.
C
C      INPUT:
C      TITLE = PLOT TITLE (16 DIGITS)
C      NOTES = ADDITIONAL NOTES (32 DIGITS)
C      CPERG = CURVES PER GRAPH (<= 6)
C      XSCALE = NUMBER OF INCHES PER 20.0 WL (6.18 IS STANDARD)
C      YSCALE = NUMBER OF INCHES PER CURVE FOR LINEAR PLOTS (< 5.),
C              = DB PER INCH FOR DB PLOTS (=> 5.)
C      COMP = COMPONENT LABEL - A 3-DIGIT INTEGER:

```

```

C          FIRST DIGIT      1=F, 2=H;
C          SECOND DIGIT     1=RHO, 2=PHI, 3=ZED;
C          THIRD DIGIT      1=BROADSIDE, 2=ENDFIRE
C          H = FIELD-STRENGTH ARRAY
C          R = RANGE ARRAY (IN WL)
C          NIN = DIMENSION OF H AND R
C          INDEX = INDEXING THROUGH H AND R ARRAYS (USUALLY =1)
C
C          REAL*4 K1,K2,LT1,LT2
C          INTEGER*4 TITLE(4),NOTES(8)
C          INTEGER*4 COMTAB(7) / 'E', 'H', 'Z3B', 'Z24', 'Z69', 'PRD', 'END' /
C          INTEGER*4 CTR/O/, GCTR/O/, CPERG, COMP
C          LOGICAL*4 DATOLD, DATNEW
C          DIMENSION H(NIN), R(NIN)
C          INTEGER*4 LABELS(3)
C          INTEGER*4 FXA(3) / 'RHO', 'PHI', 'ZED' /
C          RETURN
C
C
C..ENTRY POINT FOR THEORY CURVES
C          ENTRY THEPLT(TITLE,NOTES,CPERG,XSCALE,YSCALE,COMP,H,R,NIN,INDEX,
C          *   D,K1,LT1,K2,LT2)
C          DATNEW=.FALSE.
C          GO TO 2
C
C
C..ENTRY POINT FOR DATA TYPE CURVES
C          ENTRY DATPLT(TITLE,NOTES,CPERG,XSCALE,YSCALE,COMP,H,R,NIN,INDEX,
C          *   SITE,RUN,FREQ,REF,ABSREF)
C          DATNEW=.TRUE.
C
C
C          2   CTR=CTR+1
C
C..IF THIS IS THE FIRST CURVE ON THE GRAPH, PLOT GRAPH OUTLINE
C          IF(CTR.EQ.1) GO TO 10
C
C..TEST FOR A FULL GRAPH
C          IF(CTR.IE.CPERG) GO TO 70
C
C..FULL-GRAPH LOGIC
C          CTR=1
C          CALL PLOT(XXX+4.20,0.,-3)
C
C..GRAPH OUTLINE
C          10   GCTR=GCTR+1
C
C..SET RANGE AND FIELD STRENGTH ARRAY DIMENSION ON INDEX BOUNDARY

```

```

      N = ((NIN-1) / INDEX) * INDEX + 1
      NFIRST = N
C
C.. CONVERT X-AXIS SCALE TO INCHES-PER-WAVELENGTH
C   OR INCHES-PER-METER
      XSCALD = YSCALE / 20.
      IF (XSCALE .GE. 10.) XSCALD = XSCALE / 1000.
C
C.. DISTANCE BETWEEN 4 WL SEGMENTS
      XSPACE = 4. * XSCALD
      IF (XSCALE .GE. 10.) XSPACE = 200. * XSCALD
C
C.. ROUND DOWN FIRST RANGE POINT TO DETERMINE GRAPH ORIGIN
      IRIST = 0
C
C.. NUMBER OF 4 WL SEGMENTS IN THE RANGE VALUES
      IF (XSCALE .LT. 10.) NUMR = (IFIX(R(N)) - IRIST) / 4 + 1
      IF (XSCALE .GE. 10.) NUMR = (IFIX(R(N)) - IRIST) / 200 + 1
C
C.. LAST SEGMENT NUMBER + ONE
      ILAST = NUMR + 1
C
C.. HALFWAY POINT IN THE NUMBER OF SEGMENTS
      IHALF = ILAST / 2
C
C.. CO-ORDINATE OF Y-AXIS LABEL
      XLABEL = 2 * NUMR * XSCALD - .5
      IF (XSCALE .GE. 10) XLABEL = 100 * NUMR * XSCALD - .5
C
C.. CO-ORDINATE OF GRAPH TITLE
      XTITLE = AMAX1(.5, XLABEL - 1.3)
C
C.. DRAW Y-AXIS
      CALL SYMBOL(0., 10., .1, 6, 0., -2)
C
C.. TITLE GRAPH AND PLOT ANY NOTES (EXPLANATION, ETC)
      CALL NUMBER(XTITLE, 10., .15, FREQ, 0., 1)
      CALL SYMBOL(999., 999., .15, 5H MHZ., 0., 5)
      CALL SYMBOL(999., 999., .15, 14H APOLLO 17, 0., 14)
      CALL SYMBOL(XTITLE, 9.8, .07, NOTES, 0., 32)
C
C.. SET *XXX* VARIABLE WHICH IS RIGHTMOST POSITION OF X-AXIS SEGMENTS
      XXX = NUMR * XSPACE
C
C.. DRAW X-AXIS (BACKWARDS)
      YAX = XXX + XSPACE
      CALL SYMBOL(YAX, 0., .1, 6, 270., -1)
      DO 35 I = 1, ILAST

```

```

XAX=XAX-XSPACE
35 CALL SYMBOL(XAX,0.,.05,13,0.,-2)
C
C..NUMBER FIRST HALF OF X-AXIS
DNUM=IRIST-3.9
IF(XSCALE .GE. 10.) DNUM = -199.9
XNUM=-XSPACE-.05
DO 36 I=1,IHALF
IF(XSCALE .LT. 10.) DNUM = DNUM + 4.
IF(XSCALE .GE. 10.) DNUM = DNUM + 200.
XNUM=XNUM+XSPACE
36 CALL NUMBER(XNUM,-.15,.07,DNUM,0.,-1)
C
C..LABEL X-AXIS
CALL SYMBOL(XLABEL,-.3,.1,6HPANGE,0.,6)
IF(XSCALE .GE. 10.) CALL SYMBOL(999.,999.,.1,2HP.,0.,2)
IF(XSCALE .LT. 10.) CALL SYMBOL(999.,999.,.14,41,0.,-1)
C
C..NUMBER SECOND HALF OF X-AXIS
DO 37 I=IHALF,NUMF
IF(XSCALE .LT. 10.) DNUM = DNUM + 4.
IF(XSCALE .GE. 10.) DNUM = DNUM + 200.
XNUM=XNUM+XSPACE
37 CALL NUMBER(XNUM,-.15,.07,DNUM,0.,-1)
C
C..LABEL Y-AXIS
IF(YSCALE.GT.5.) GO TO 38
CALL SYMBOL(-.15,4.5,.1,6HLINEAR,90.,6)
GO TO 39
38 CALL NUMBER(-.15,4.5,.1,YSCALE,90.,-1)
CALL SYMBOL(999.,999.,.1,3H PB,90.,3)
CALL SYMBOL(-.1,4.25,.06,13,90.,-1)
CALL SYMBOL(-.1,4.27,.04,6,180.,-1)
CALL SYMBOL(-.1,5.23,.04,6,0.,-2)
CALL SYMBOL(-.1,5.25,.06,13,90.,-1)
CALL SYMBOL(-.15,6.0,.1,7HREF. AT,90.,7)
CALL NUMBER(-.15,6.8,.1,ABSREF,90.,1)
CALL SYMBOL(999.,999.,.1,4H DEM,90.,4)
C
C..END OF GRAPH VARIABLE SET-UPS
39 DATOLD=.NOT.DATNEW
CPEFG=MIN0(6,CPERG)
SHIFT=6./(CPERG-1.)
ORIGIN=6.+SHIFT
GO TO 71
C
C
C..ENTRY POINT FOR PLOTTING A CURVE

```



```

70 N= ((NIN-1)/INDEX)*INDEX+1
   NFIRST=N

```

C

C..SET THE ORIGIN FOR THIS CURVE

```

71 ORIGIN=ORIGIN-SHIFT
   IC3=5+MOD(COMP,10)
   IC2=2+MOD(COMP/10,10)
   IC1=COMP/100
   LABELS(1) = COMTAB(IC1)
   LABELS(2) = RXA(IC2 - 2)
   LABELS(3) = COMTAB(IC3)
   WRITE(6, 90050) FREQ, LABELS

```

```

90050 FORMAT('O PLOTTING ', F6.1, 3Y, A2, 2A4)

```

C

C..FIND MAXIMUM AND MINIMUM FIELD STRENGTH VALUES

```

   YMAX=H(1)
   YMIN=YMAX
   DO 97 I=1,N,INDEX
   YMIN=AMIN1(H(I),YMIN)
97   YMAX=AMAX1(H(I),YMAX)

```

C

C..TEST FOR LINEAR PLOTS

```

   IF(YSCALE.LT.5.) GO TO 700

```

C

C..CONVERT DE VALUES TO INCHES AND ZERO LOW VALUES

```

   DO 98 I=1,N,INDEX
98   H(I) = H(I) / YSCALE
   GO TO 99

```

C

C..CONVERT LINEAR VALUES TO INCHES

```

700 HDELTA=YSCALE/YMAX
   DO 701 I=1,N,INDEX
701   U(I)=H(I)*HDELTA

```

C

C..PLOT THE CURVE

```

99   CALL PLOT(-.05, REF / YSCALE + ORIGIN, 3)
   CALL PLOT(.05, REF / YSCALE + ORIGIN, 2)
   CALL PLOT((R(1)-IFIRST)*XSCALD, H(1)+ORIGIN, 3)
   NST=1+INDEX
   DO 800 I=NST,N,INDEX
800  CALL PLOT((R(I)-IFIRST)*XSCALD, H(I)+ORIGIN, 2)

```

C

C..IF FIRST CURVE ON GRAPH, PLOT *COMP* AND *MAX* HEADINGS

```

   YYNO=U(N)+ORIGIN
   YYHD=YYNO+.15
   IF(CTR.EQ.1) CALL SYMBOL(XXX+.035,YYHD,.070,4HCOMP,0.,4)

```

C

C..PLOT COMPONENT AND MAXIMUM

```

CALL SYMBOL (XXX+.035,YYNO,.070,COMTAB(IC1),0.,1)
CALL $YMBOL(999.,999.,.070,47,0.,-1)
CALL $YMBOL(999.,999.,.070,COMTAB(IC2),0.,-1)
CALL $YMBOL(999.,999.,.070,46,0.,-1)
CALLSYMBOL(999.,999.,.070,COMTAB(IC3),0.,3)

```

C
C

```

C..CURVE LABELLING TESTS FOLLOW
  IF (DATNEW) GO TO 40

```

C

```

C..THEORY CURVE
  IF (.NOT.DATOLD) GO TO 41

```

C

```

C..THEORY HEADINGS
  CALL SYMBOL (XXX+1.015,YYHD,.07,32HDEPTH   K1      LT1      F2      IT2
*,0.,32)

```

C

```

C..THEORY VARIABLES
41  CALL NUMBER (XXX+1.015,YYNO,.070,D      ,0.,3)
    CALL NUMBER (XXX+1.505,YYNO,.070,K1    ,0.,2)
    CALL NUMBER (XXX+1.925,YYNO,.070,LT1   ,0.,4)
    CALL NUMBER (XXX+2.555,YYNO,.070,K2    ,0.,2)
    CALL NUMBER (XXX+2.975,YYNO,.070,LT2   ,0.,3)

```

C

```

40  CONTINUE
999  DATOLD=DATNEW
    RETURN
    ENTEY BASEL(H, R, NTN)
    DO 90100 I = 1, NIN
    CALL SYMBOL ((P(I)-IRIST)*XSCALD, H(I)/YSCALEP+ORIGIN,.07,11,0.,-1)
90100 CONTINUE
    RETURN
    END

```

```

*****
*
*
*
*
*****

```

STOPT

LOGICAL FUNCTION STOPT*1 (I, IFR)

C
C
C
C
C

RETURNS .TRUE. IF THE LRV WAS STOPPED DURING THE EP-4
TURN, .FALSE. OTHERWISE. (THIS DECISION IS BASED ON THE
VALUES PLACED IN THE ARRAY "B" BY THE CALLING ROUTINE.)

```

C
  INTEGER*2 B(6)
  INTEGER*2 C(6) / 33, 33, 20, 20, 7, 7 /
C
  COMMON /BOUNDS/ B
C
C
  J = 13 * I + C(IFR)
  STOBT = (      J .GT. B(2)
.         .AND. J .LT. B(3) )
.         .OR. (      J .GT. B(4)
.         .AND. J .LT. B(5) )
  RETURN
  END

```

```

*****
*
*                               TXOSTAT
*
*****

```

```

C
C
C           PROGRAM TO COMPARE TRANSMITTER-OFF DATA
C           WITH APPROXIMATE LRV SPEED.
C
C           ONE INPUT CARD IS REQUIRED, CONTAINING SIX INTEGER VALUES IN
C           FORMAT 6I5; THESE VALUES SHOULD BE THE SAME AS THE "BOUNDS" FOR
C           THE 32.1 MHZ. INPUT TO ANTENNA0.
C
C

```

```

  REAL*4 SST(3), SMO(3), SSTSQ(3), SMOSQ(3)
  REAL*4 RANGE(140, 6), SPEED(140, 6), TXOFF(140, 6, 3)
  REAL*4 TXOFS(140, 6, 3), SMOFS(3), SSTFS(3)
  REAL*4 SSTFSQ(3), SMOFSQ(3)
  INTEGER*4 NTXOFF(6, 3), NR(3), INDEX(140)
  INTEGER*2 BOUND(6)
  LOGICAL*1 STOBT
  LOGICAL*1 SWITCH

```

```

C
  COMMON /BOUNDS/ BOUND
C
C
  CALL PLOTST('OOGP.JCR.TXOFF ', 16, 'PGS1410 ')
C
  READ(3)

```

```

READ(3)
READ(3)
READ(3) TXOFF, NTXOFF
READ(3) RANGE, NR
READ(3) SPEED
READ(5, 3000) BOUND

```

```

                LOOP THROUGH FREQUENCIES

```

```

DO 100 IFR = 1, 6
  IFRPO = 2 ** (IFR - 1)
  I = 0
  N = 0
  SWITCH = .FALSE.
  NST = 0
  NMO = 0
  DO 5 J = 1, 3
    SST(J) = 0.
    SMO(J) = 0.
    SMOFS(J) = 0.
    SSTFS(J) = 0.
    SSTSQ(J) = 0.
    SMOSQ(J) = 0.
    SSTFSQ(J) = 0.
    SMOFSQ(J) = 0.
5  CONTINUE
10  WRITE(6, 1000) IFRPO
    LIN = 0
20  I = I + 1
    IF(RANGE(I, IFR) .GT. 1667.) GO TO 50
    N = N + 1
    IF(RANGE(I, IFR) .EQ. 513.0 .AND. .NOT. STOPT(I, IFR)) GO TO 20
    DO 23 J = 1, 3
      TXOFS(I, IFR, J) = 10. ** (.05 * TXOFF(I, IFR, J))
23  CONTINUE
    LIN = LIN + 1
    WRITE(6, 2000) RANGE(I, IFR), SPEED(I, IFR),
      (TXOFF(I, IFR, J), J = 1, 3),
      (TXOFS(I, IFR, J), J = 1, 3)
    IF(SPEED(I, IFR) .EQ. 0.) GO TO 30
    NMO = NMO + 1
    DO 25 J = 1, 3
      SMO(J) = SMO(J) + TXOFF(I, IFR, J)
      SMOFS(J) = SMOFS(J) + TXOFS(I, IFR, J)
25  CONTINUE
    GO TO 40
30  NST = NST + 1
    DO 35 J = 1, 3

```

```

SST(J) = SST(J) + TXOFF(I, IFR, J)
SSTFS(J) = SSTFS(J) + TXOFS(I, IFR, J)
35 CONTINUE
40 IF(LIN .LT. 55) GO TO 20
GO TO 10

```

C
C
C
C
C
C
C
C
C
C
C
C
C
C
C

```

DO A SORT ON THE APPROPRIATE SEGMENT OF THE SPEED ARRAY.
SUBROUTINE BUBBLE RETURNS THE VECTOR INDEX CONTAINING
INDICES TO THE DATA ARRAYS SUCH THAT IF I < J, THEN
SPEED(INDEX(I), IFR) < SPEED(INDEX(J), IFR)

50 CALL BUBBLE(SPEED(1, IFR), INDEX, N)

LIST THE SPEED VALUES IN ASCENDING ORDER WITH THE CORRESPONDING
TXOFF VALUES FOR EACH ANTENNA.

```

```

LIN = 0
DO 60 J = 1, 3
  SMO(J) = SMO(J) / NMO
  SST(J) = SST(J) / NST
  SMOFS(J) = SMOFS(J) / NMO
  SSTFS(J) = SSTFS(J) / NST
60 CONTINUE
DO 80 I = 1, N
  IF(IIN .EQ. 0) WRITE(6, 1010) I, IFR
  IX = INDEX(I)
  IF(RANGE(IX, IFR) .NE. 513.9 .OR. STOPT(IX, IFR)) GO TO 70
  SPEED(IX, IFR) = -1.
  GO TO 80
70 IF(SWITCH .OR. SPEED(IX, IFR) .EQ. 0.) GO TO 75
DO 72 J = 1, 3
  SSTSQ(J) = SORT(SSTSQ(J) / (NST - 1))
  SSTFSQ(J) = SORT(SSTFSQ(J) / (NST - 1))
72 CONTINUE
WRITE(6, 4000) NST, SST, SSTFS, SSTSQ, SSTFSQ
LIN = LIN + 5
SWITCH = .TRUE.
75 CONTINUE
DO 79 J = 1, 3
  IF(SWITCH) GO TO 77
  A = TXOFF(IX, IFR, J) - SST(J)
  B = TXOFS(IX, IFR, J) - SSTFS(J)
  SSTSQ(J) = SSTSQ(J) + A * A
  SSTFSQ(J) = SSTFSQ(J) + B * B
  GO TO 79
77 A = TXOFF(IX, IFR, J) - SMO(J)
  B = TXOFS(IX, IFR, J) - SMOFS(J)
  SMOSQ(J) = SMOSQ(J) + A * A

```



```
*****
*
*
*
*
*****
```

TXPLOT

```
SUBROUTINE TXPLOT(S, TXX, TXY, TXZ, N, IFR)
```

```
REAL*4 S(N), TXX(N), TXY(N), TXZ(N)
REAL*4 TXORG(3) / 0., 3., 6. /
REAL*4 SCALE
SCALE(ARG) = .1 * (ARG + 135.)
```

```
DRAW THE SPEED AXES
```

```
X = 6.
DO 5 I = 1, 3
  CALL PLOT(0., TXORG(I), 3)
  CALL SYMBOL(X, TXORG(I), .07, 6, -90., -2)
5 CONTINUE
```

```
AND LABEL THEM
```

```
DO 10 I = 1, 5
  X = X - 1.
  DO 8 J = 1, 3
    CALL SYMBOL(X, TXORG(J), .07, 13, 0., -1)
8 CONTINUE
  CALL NUMBER(X - .105, -.2, .07, X, 0., 1)
10 CONTINUE
CALL SYMBOL(1.5, -.5, .14, 22HLRV SPEED (M. / SEC.), 0., 22)
```

```
DRAW THE DB AXES
```

```
DO 30 J = 1, 3
  CALL PLOT(0., TXORG(J), 3)
  CALL SYMBOL(0., TXORG(J) + 2.5, .07, 6, 0., -2)
```

```
AND LABEL THEM
```

```
DO 20 I = 1, 3
  CALL SYMBOL(0., TXORG(J) + 3 - I, .07, 13, 90., -1)
  CALL NUMBER(-.13, TXORG(J) + 2.79 - I, .07, -105. - 10. * I,
    90., 1)
20 CONTINUE
30 CONTINUE
```


C ALSO A PLTID NAMELIST CARD IS REQUIRED BY PLINTJ (IF PLOT = 'TRUE').
 C

```

    INTEGER*4 H, M, S
    REAL*4 X(5), Y(5)
    REAL*4 SR(3088), ST(3088), VP(2565), VT(2565)
    REAL*4 TO / 1452. /, RO / 0. /
    LOGICAL*1 STAT / .TRUE. /, PLOT / .FALSE. /
    COMMON / BLOCK / SR, ST, VP, VT, SCALE
    NAMELIST / CNTL / TO, RO, STAT, PLOT, SCALE
    
```

C SET AND READ CONTROL PARAMETERS, AND SKIP OVER UNWANTED SEP DATA.
 C

```

    SCALE = 500.
    READ(5, CNTL)
    DO 10 I = 1, 71
        READ(3, 1000)
    10 CONTINUE
    
```

C READ 16 MHZ. RANGE DATA.
 C

```

    K = 1
    L = 386
    DO 20 I = 1, 8
        READ(3, 2000) (SR(J), J = K, L)
        K = K + 386
        L = L + 386
    20 CONTINUE
    
```

C ADJUST SEP RANGE AND DB DATA USING SUPPLIED PARAMETERS
 C

```

    DO 30 I = 1, 3088
        ST(I) = .81 * (I - 1) + TO
        SR(I) = SR(I) + RO
    30 CONTINUE
    
```

C READ VLBI DATA IN GROUPS OF ONE TIME AND FIVE X-Y PAIRS, CONVERT
 C TIME TO SECONDS AND X-Y PAIRS TO RANGES, AND STORE.
 C

```

    DO 50 I = 1, 2565, 5
        READ(4, 100) H, M, S, (X(J), Y(J), J = 1, 5)
        T = S + 60 * (M + 60 * H)
        DO 40 J = 1, 5
            II = J - 1
            JJ = I + II
            VT(JJ) = T + II
            VR(JJ) = SQRT(X(J) * X(J) + Y(J) * Y(J))
        40 CONTINUE
    50 CONTINUE
    
```

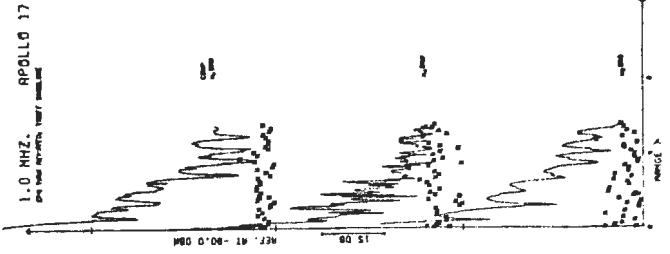
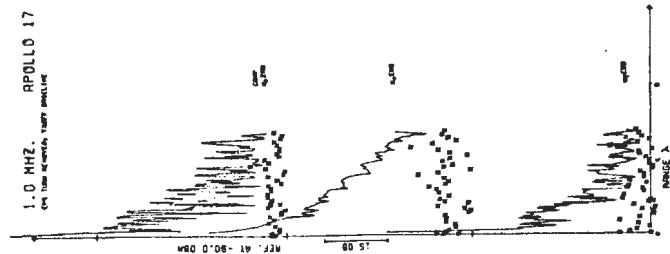
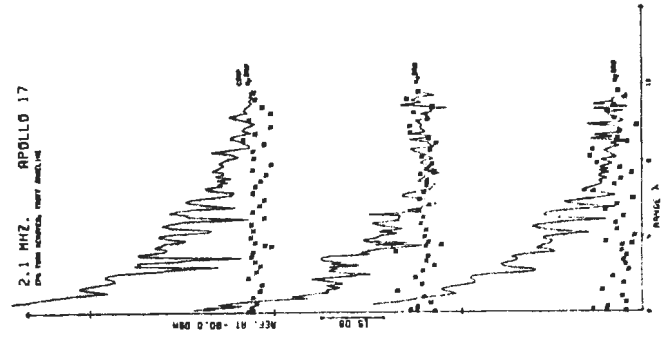
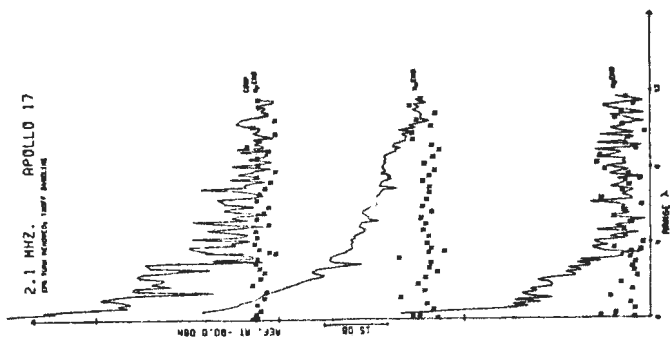
```

C
  IF(PLOT) CALL RTPLOT
  IF(.NOT. STAT) GO TO 99
  JST = 1
C
C   SKIP ALL VLBI TIMES WHICH ARE LESS THAN THE FIRST SEP TIME.
C
  DO 60 J = 1, 2565
    IF(VT(J) .GE. ST(1)) GO TO 65
    JST = JST + 1
60 CONTINUE
C
65 LIN = 0
  E = 0.
  S = 0.
  I = 1
C
C   FOR EACH VLBI TIME-RANGE PAIR
C
  DO 90 J = 1, 2565
C
C   FIND THE PAIR OF SEP TIMES WHICH BRACKET THE VLBI TIME.
C
  DO 70 K = I, 3088
    IF(VT(J) .LT. ST(K)) GO TO 80
70 CONTINUE
  GO TO 95
80 I = K
  II = I - 1
C
C   COMPUTE AN INTERPOLATED SEP RANGE, AND THE DIFFERENCE BETWEEN
C   IT AND THE VLBI RANGE, AND INCREMENT THE SUM OF DIFFERENCES AND
C   THE SUM OF SQUARES OF DIFFERENCES.
C
  R = SR(II) + (SR(I) - SR(II)) * (VT(J) - ST(II))
    / (ST(I) - ST(II))
  D = VR(J) - R
  F = E + D
  S = S + D * D
  IF(MOD(LIN, 50) .EQ. 0) WRITE(6, 200)
  WRITE(6, 300) VT(J), VR(J), R, D
  LIN = LIN + 1
90 CONTINUE
C
C   COMPUTE THE MEAN AND STANDARD DEVIATION.
C
95 F = E / LIN
  S = SQRT(S / FLOAT(LIN - 1))

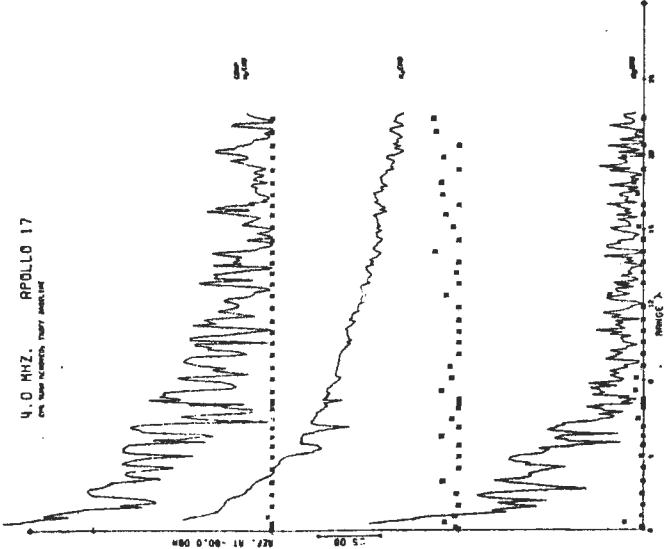
```

```
      WRITE(6, 400) E, S
C
  99 RETURN
C
 100 FORMAT(3(I2,1X), 1X, 10F5.0)
 200 FORMAT('1' / '- ', 33X, 'INTERPOLATED', 5X, 'DIFFERENCE' /
.       7X, 'VIRT TIME', 5X, 'VLBI RANGE',
.       6X, 'SFP RANGE', 7X, 'IN RANGE' / 1X )
 300 FORMAT(10X, F6.0, 9X, F6.0, 8X, F7.2, 8X, F7.2)
 400 FORMAT('-SUM(DIFF.) / N = ', F7.2 /
.       'OSQRT(SUM(DIFF. ** 2) / (N - 1)) = ', F7.3)
1000 FORMAT(200A6, 186A6)
2000 FORMAT(200F6.1, 186F6.1)
C
      END
```

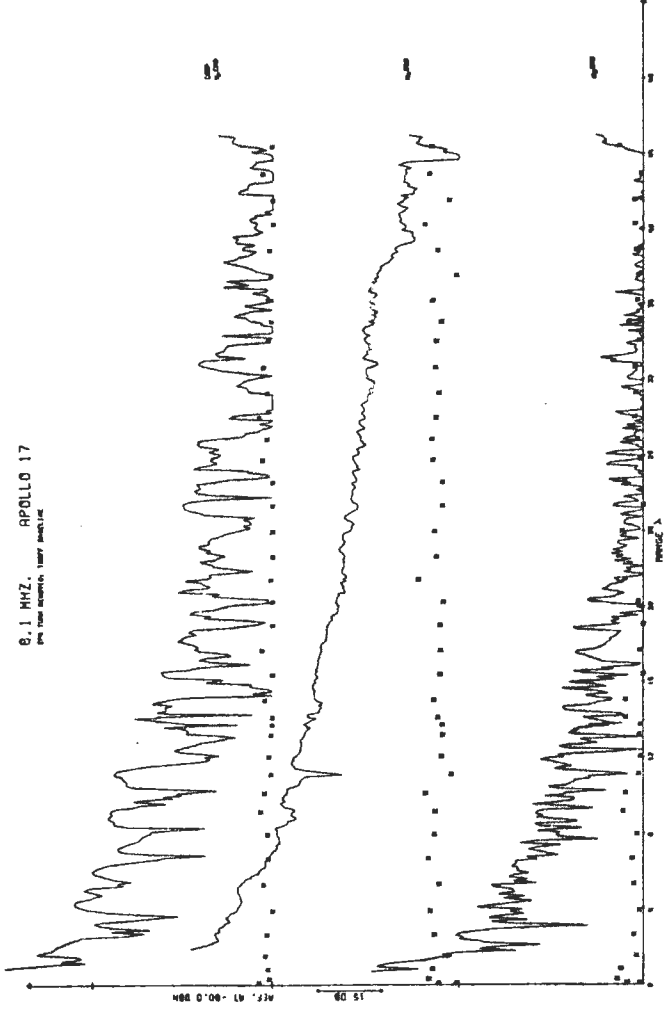
SCI2B Plots



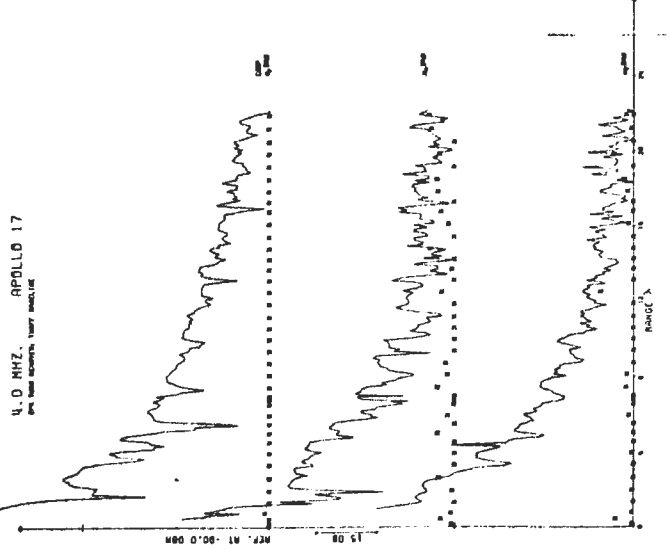
4.0 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY BRADLEY



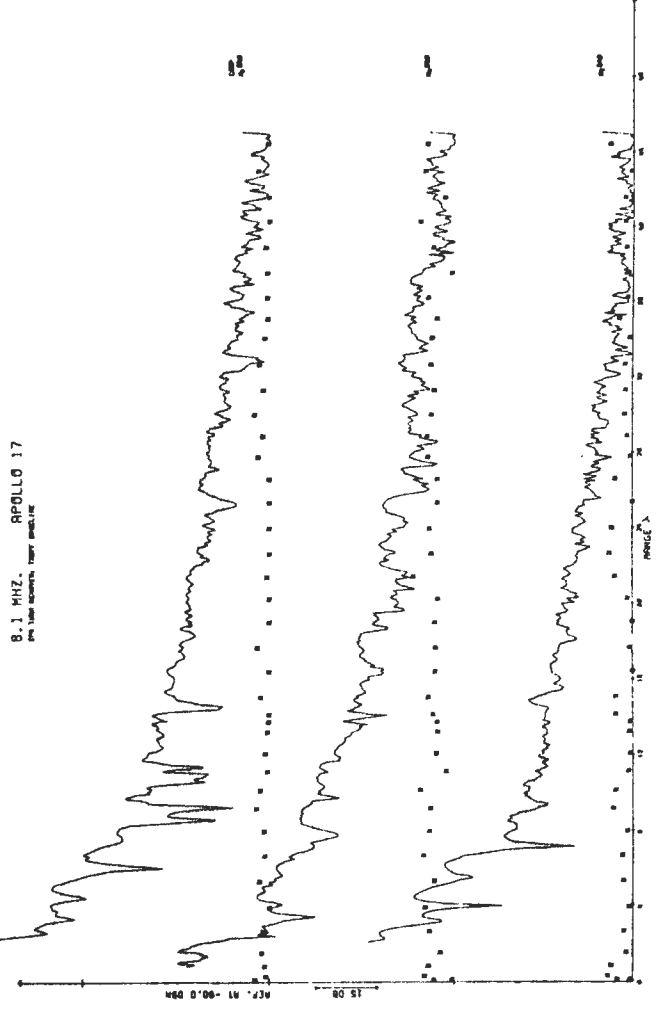
8.1 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY BRADLEY



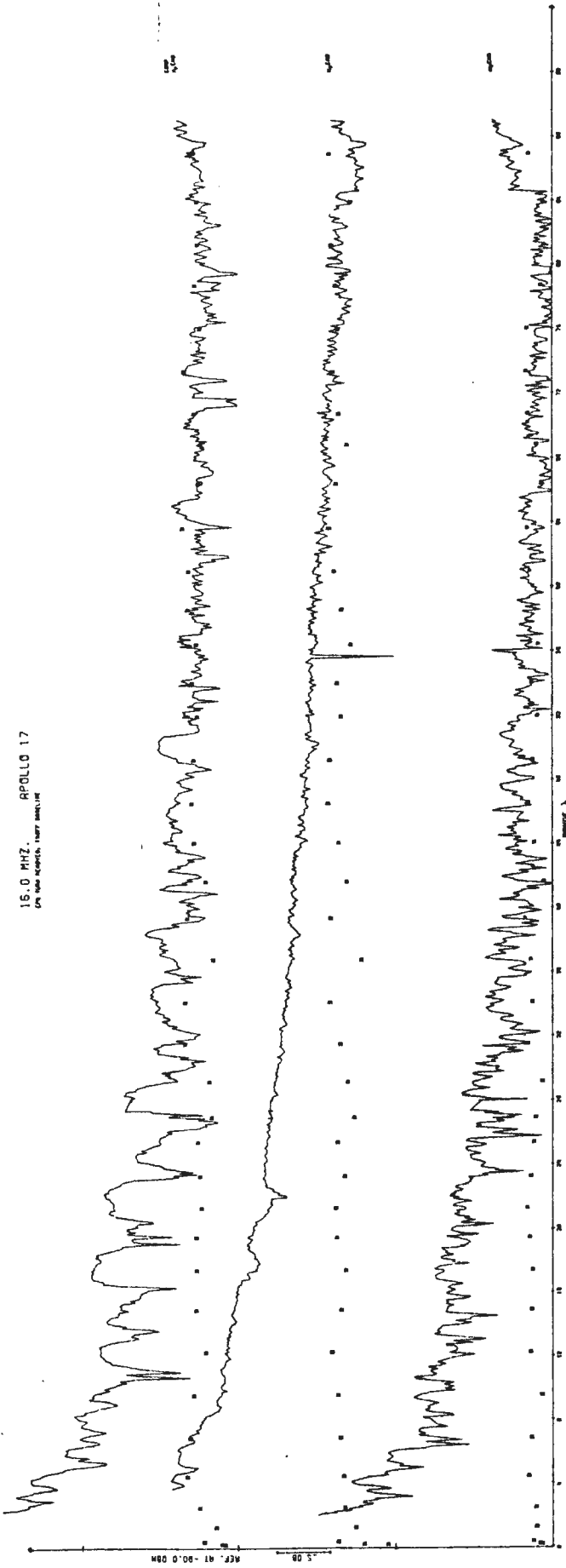
4.0 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY BRADLEY



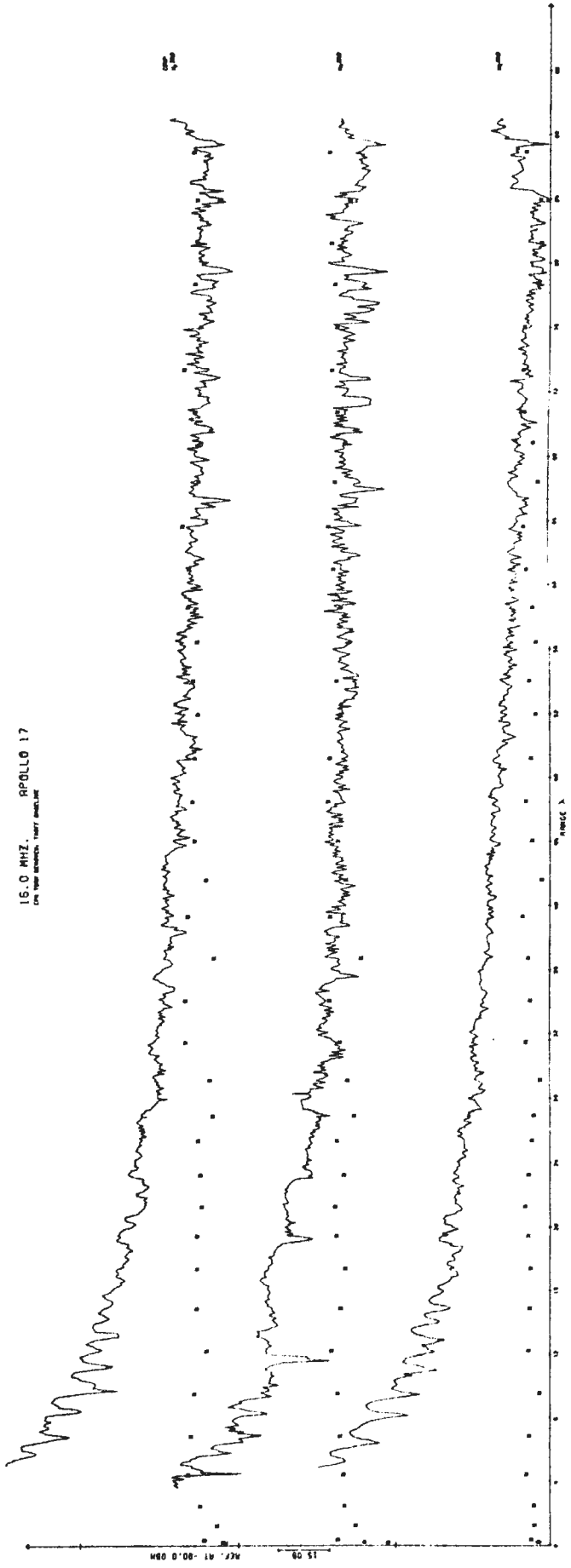
8.1 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY BRADLEY



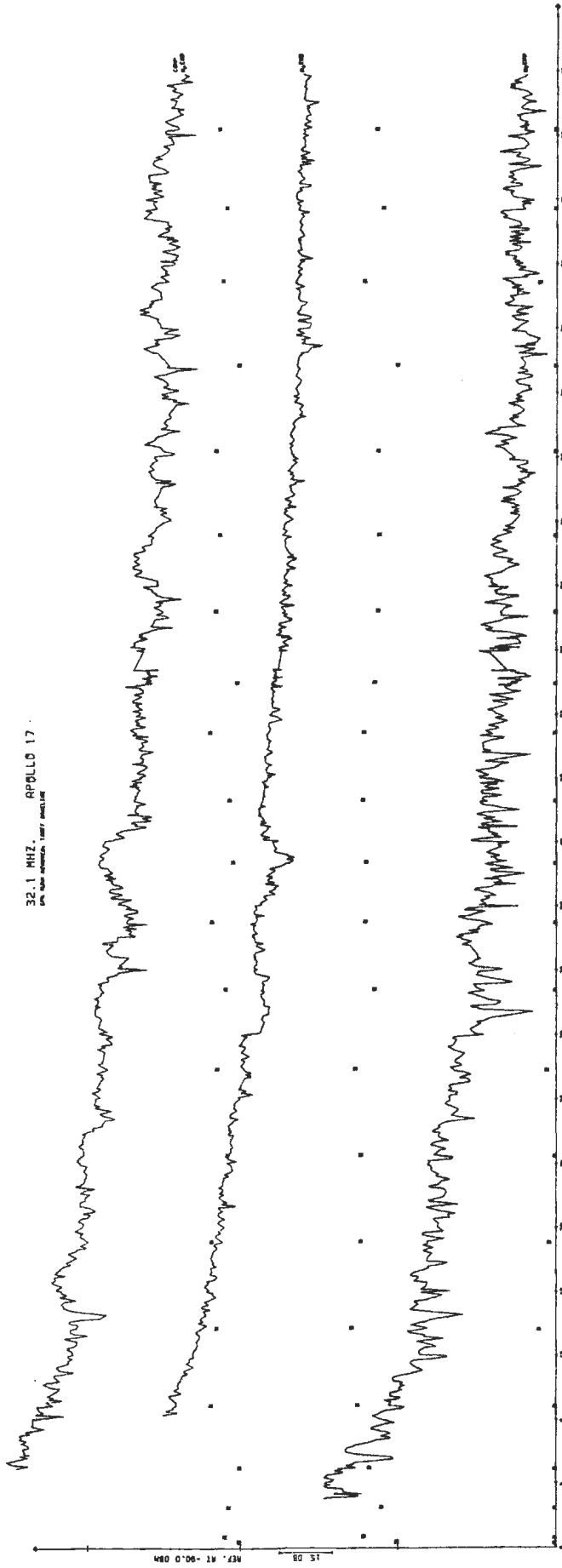
15.0 MHZ. APOLLO 17
ON THE SURFACE. TAPE ANALYSIS



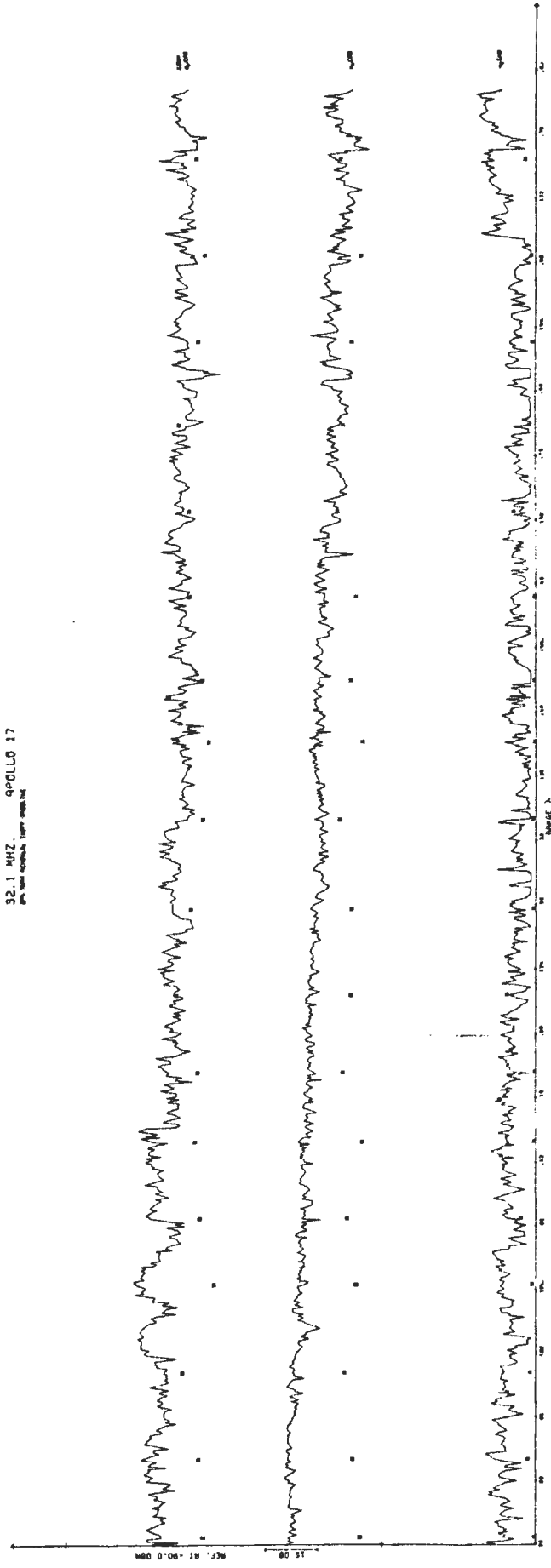
15.0 MHZ. APOLLO 17
ON THE SURFACE. TAPE ANALYSIS



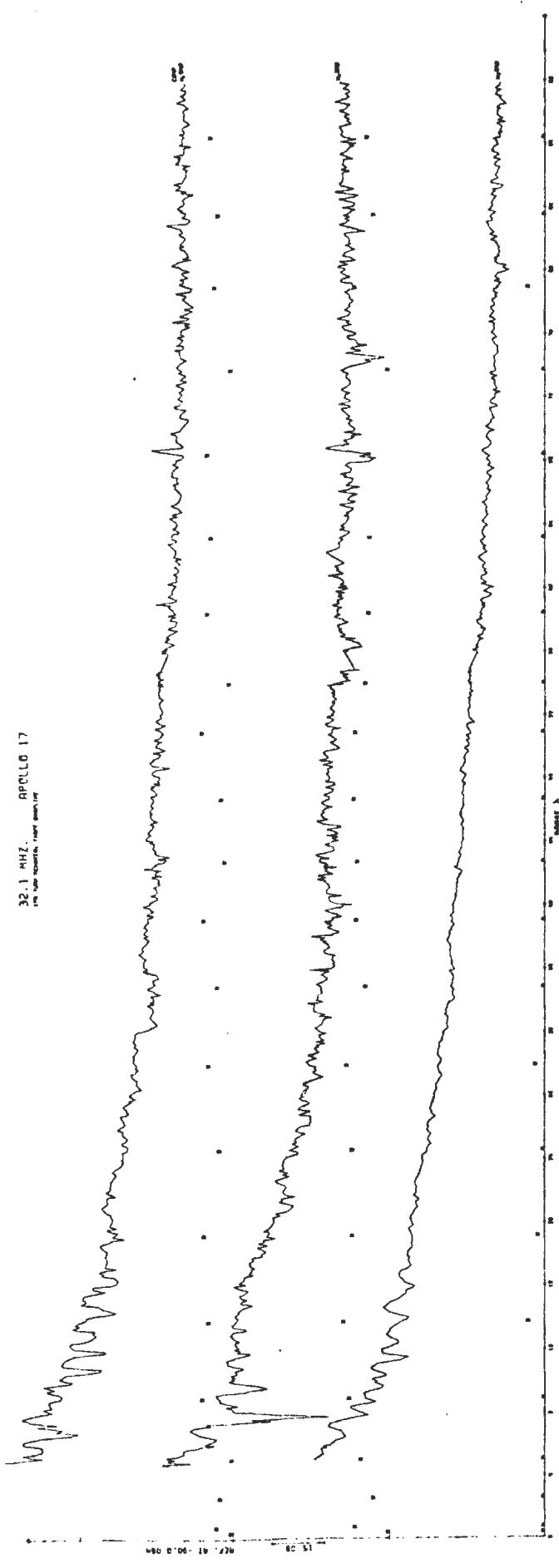
32.1 MHZ. RPOLLO 17
ON LINE MONITOR, TERRY, 08/11/68



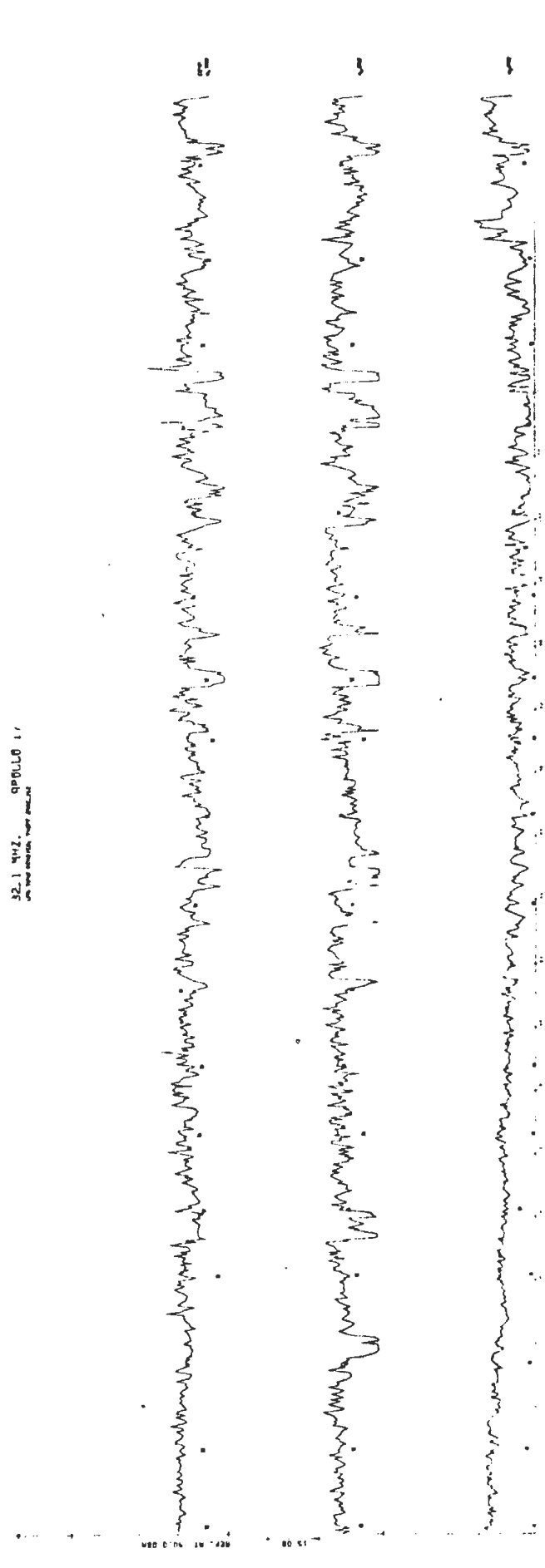
32.1 MHZ. QPOLLO 17
ON LINE MONITOR, TERRY, 08/11/68



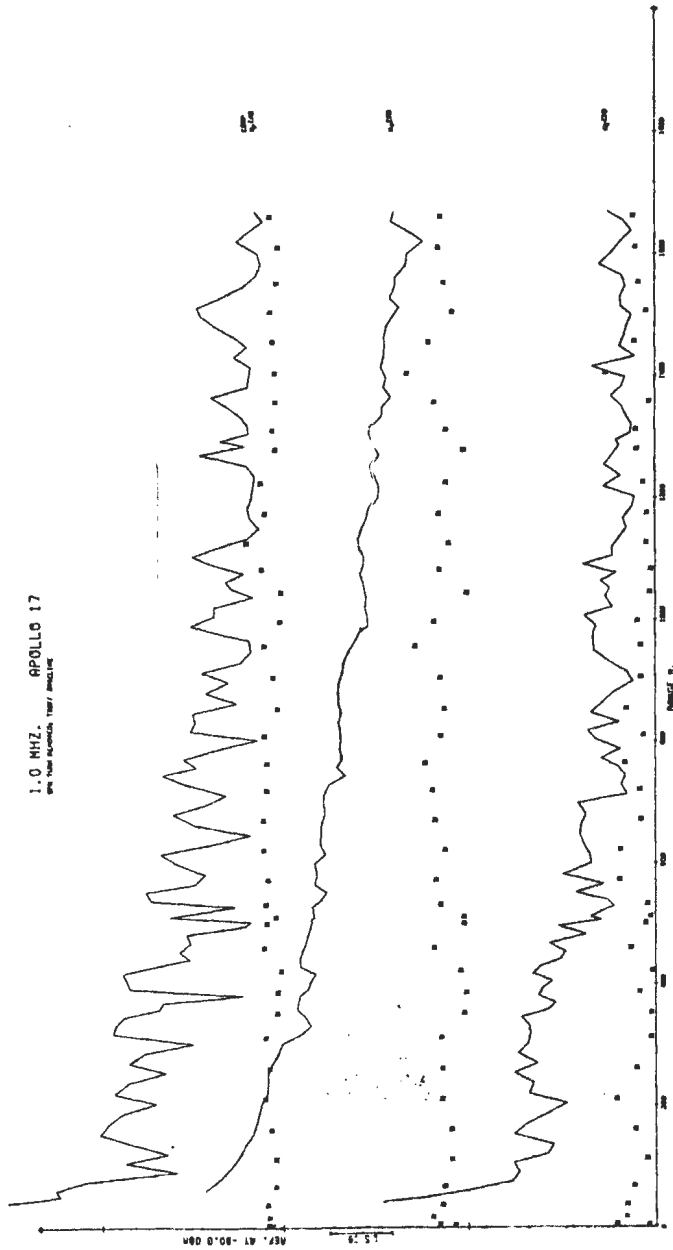
32.1 MHz. APOLLO 17
in the Apollo 17 lunar module



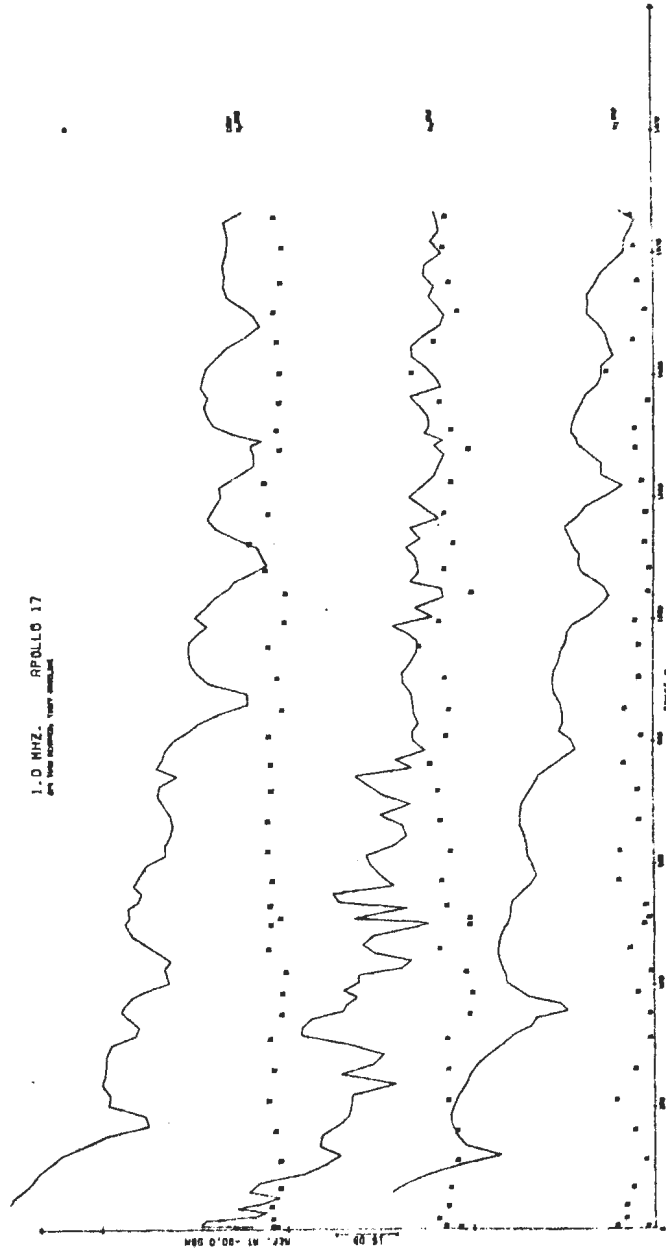
32.1 MHz. APOLLO 17
in the Apollo 17 lunar module



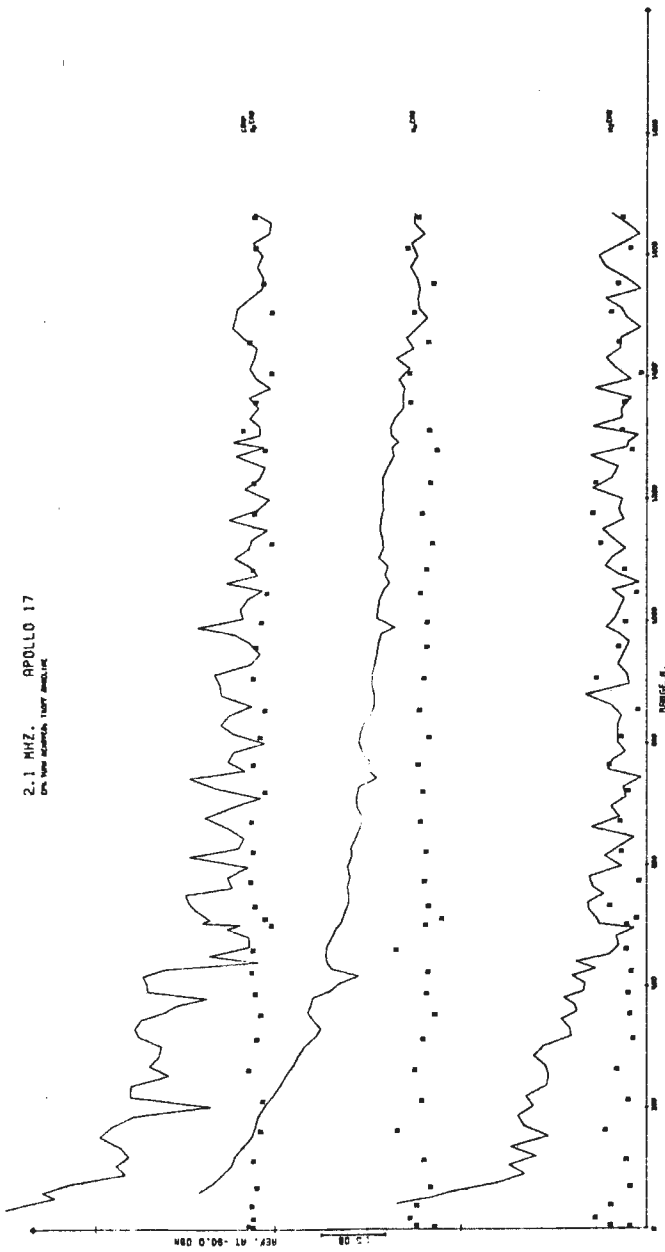
1.0 MHZ. APOLLO 17
ONE YEAR RECORD, YEAR AVERAGE



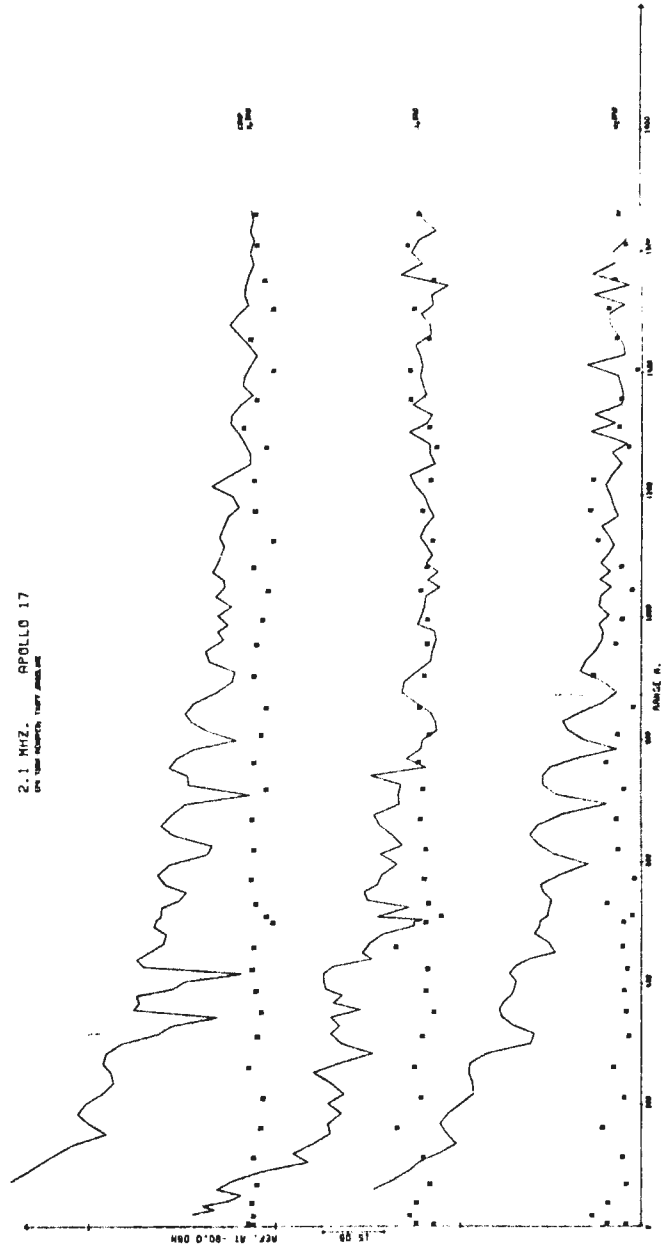
1.0 MHZ. APOLLO 17
ONE YEAR RECORD, YEAR AVERAGE

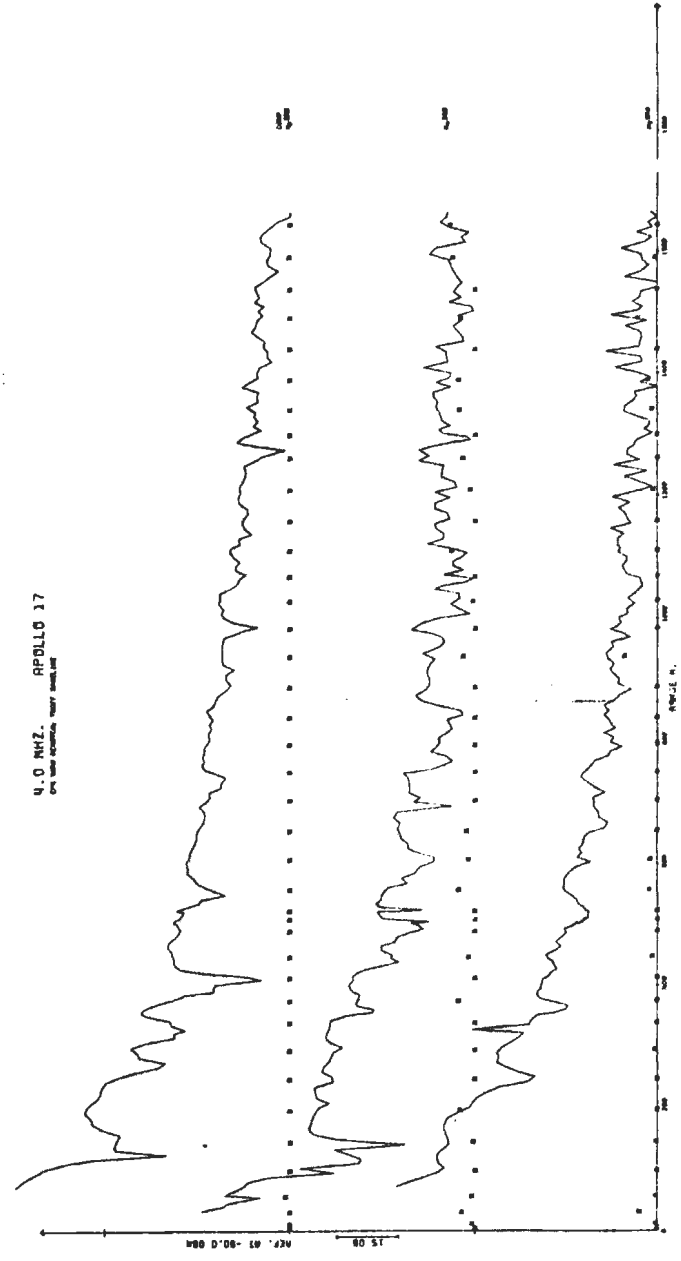
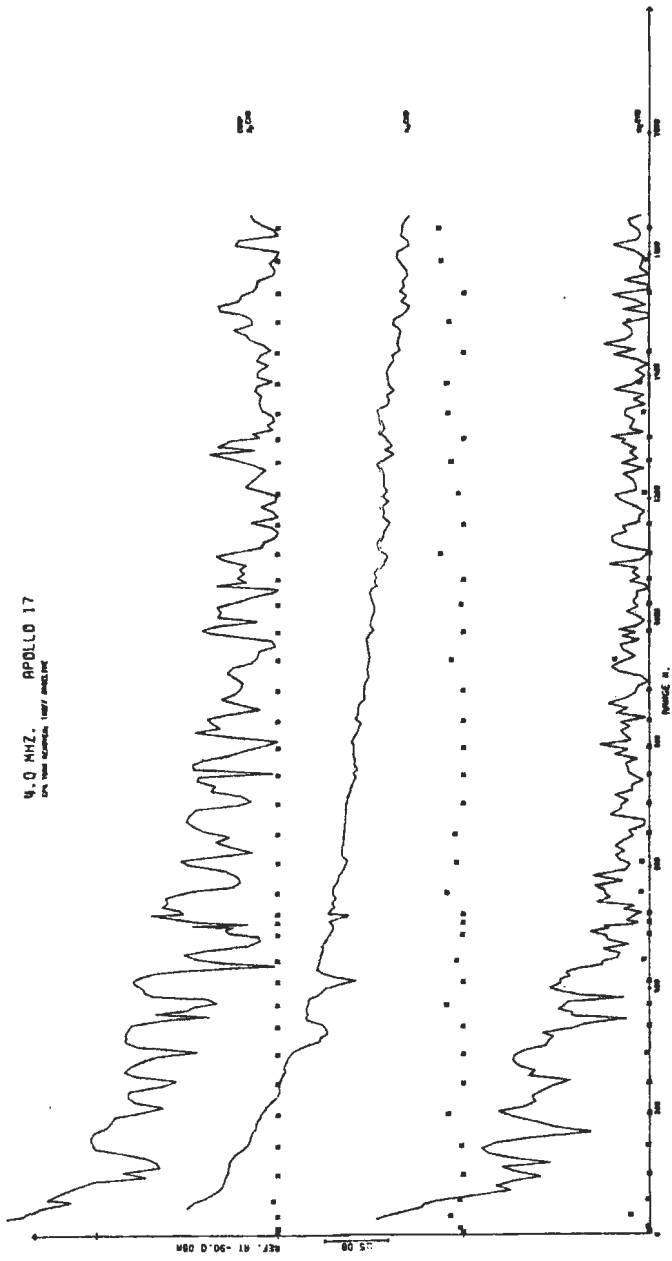


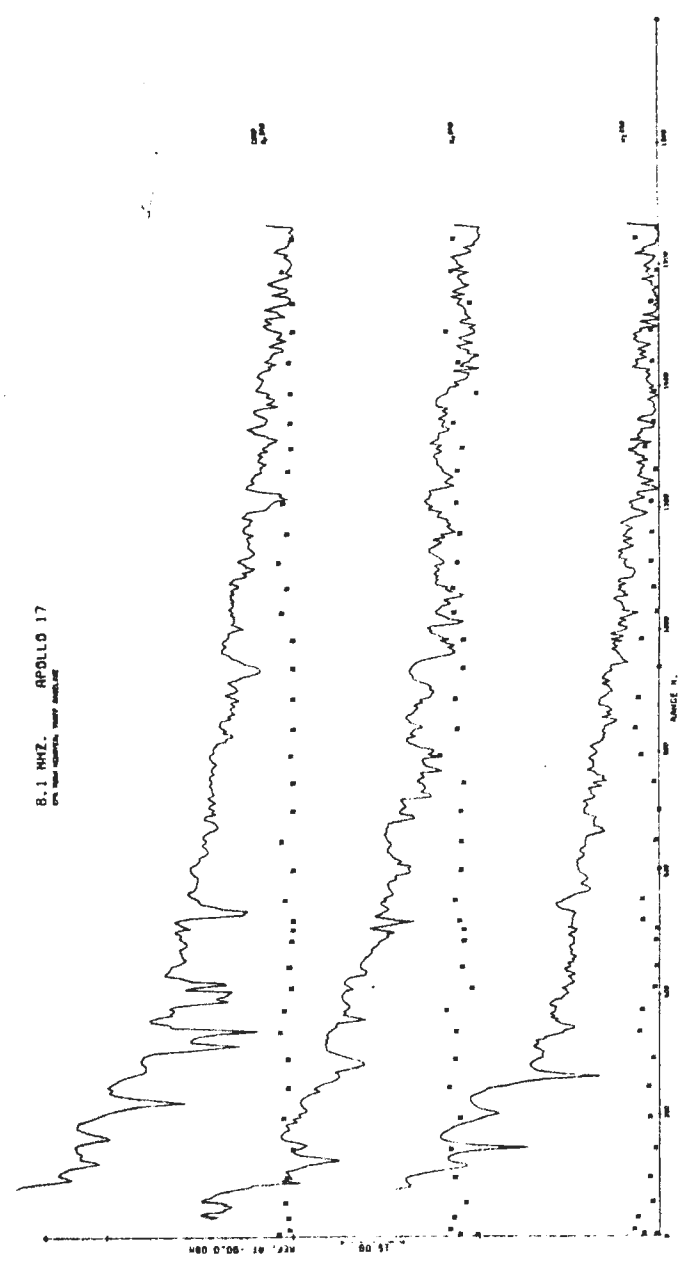
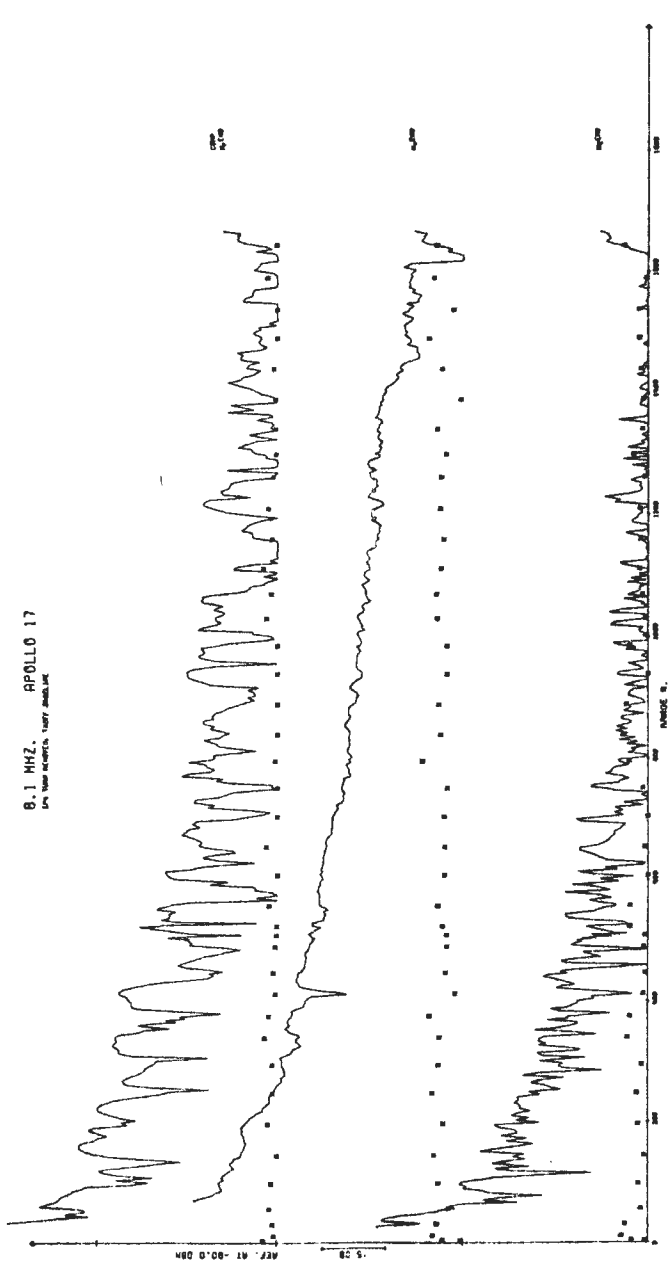
2.1 MHZ. APOLLO 17
ONE YEAR RECORDS, 1967-1968



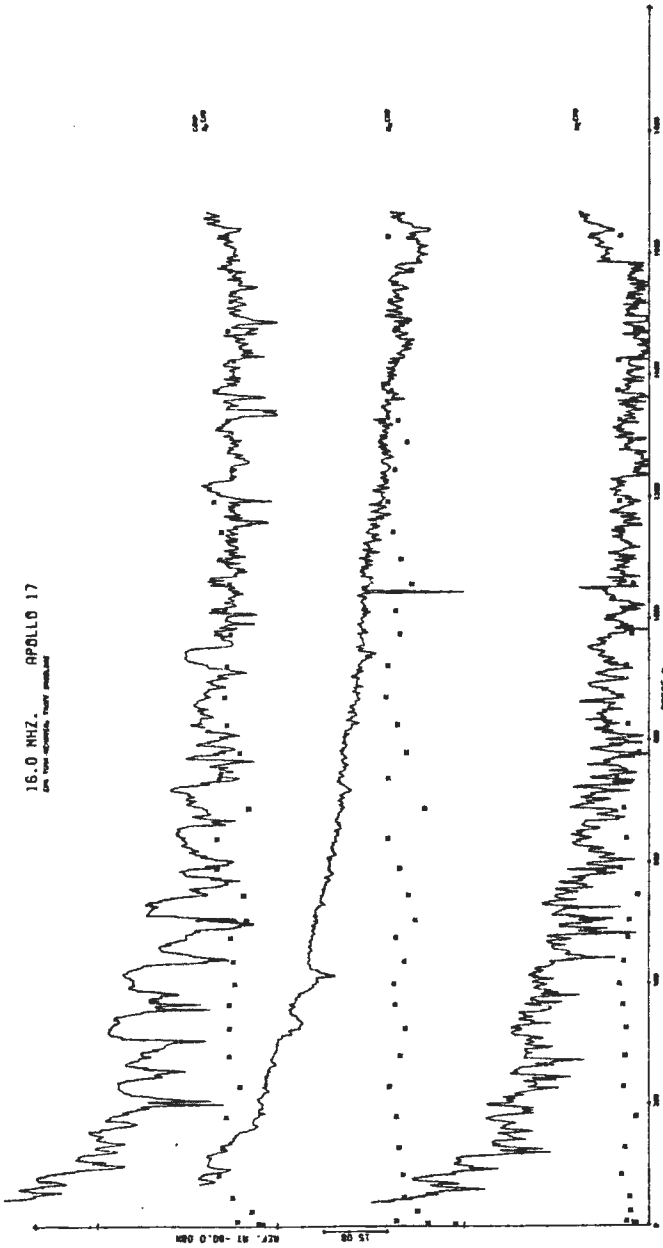
2.1 MHZ. APOLLO 17
ONE YEAR RECORDS, 1967-1968



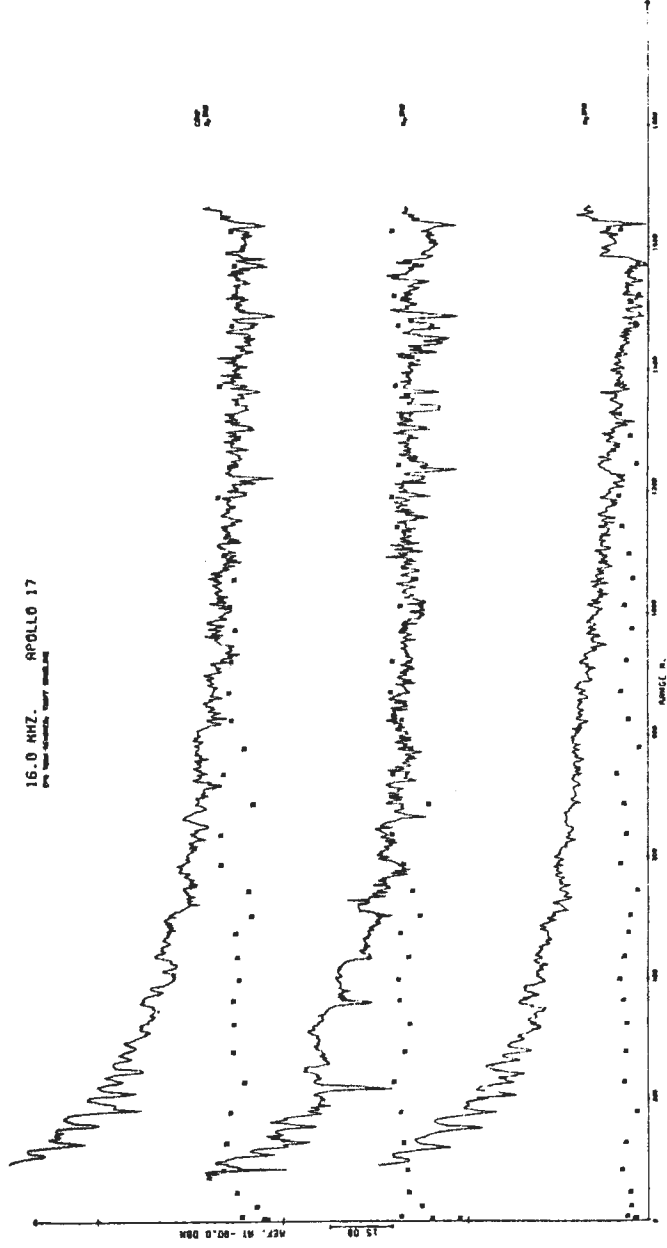




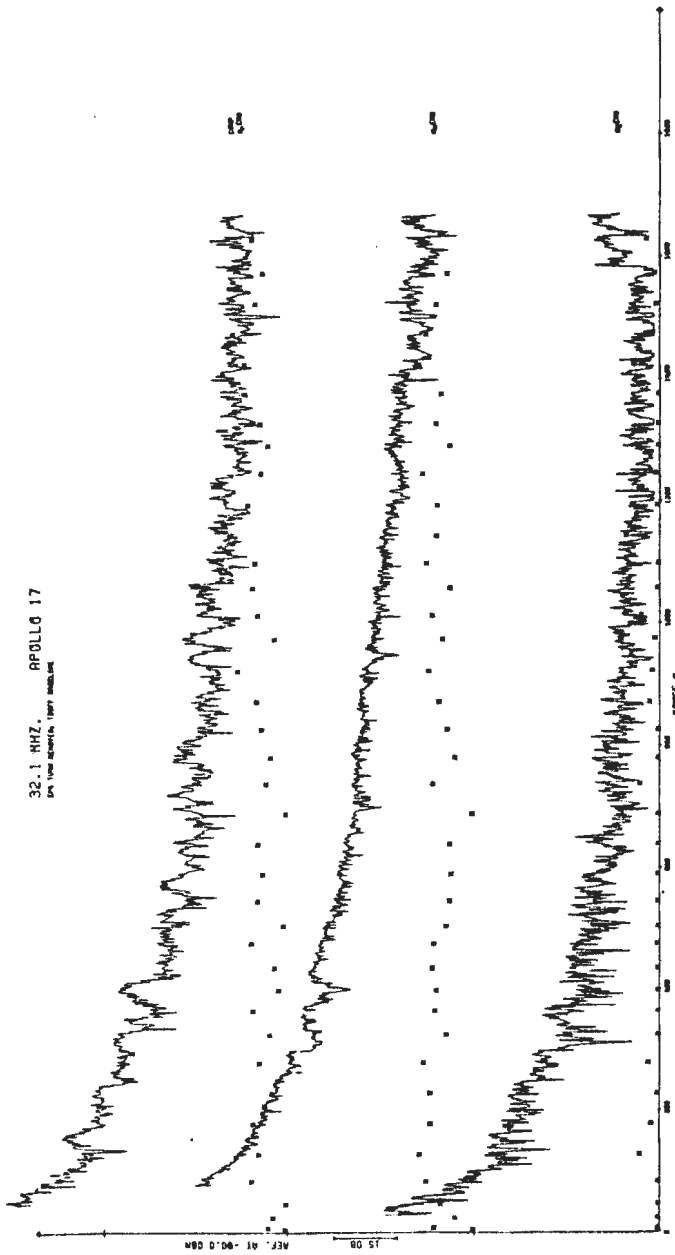
16.0 MHZ. APOLLO 17



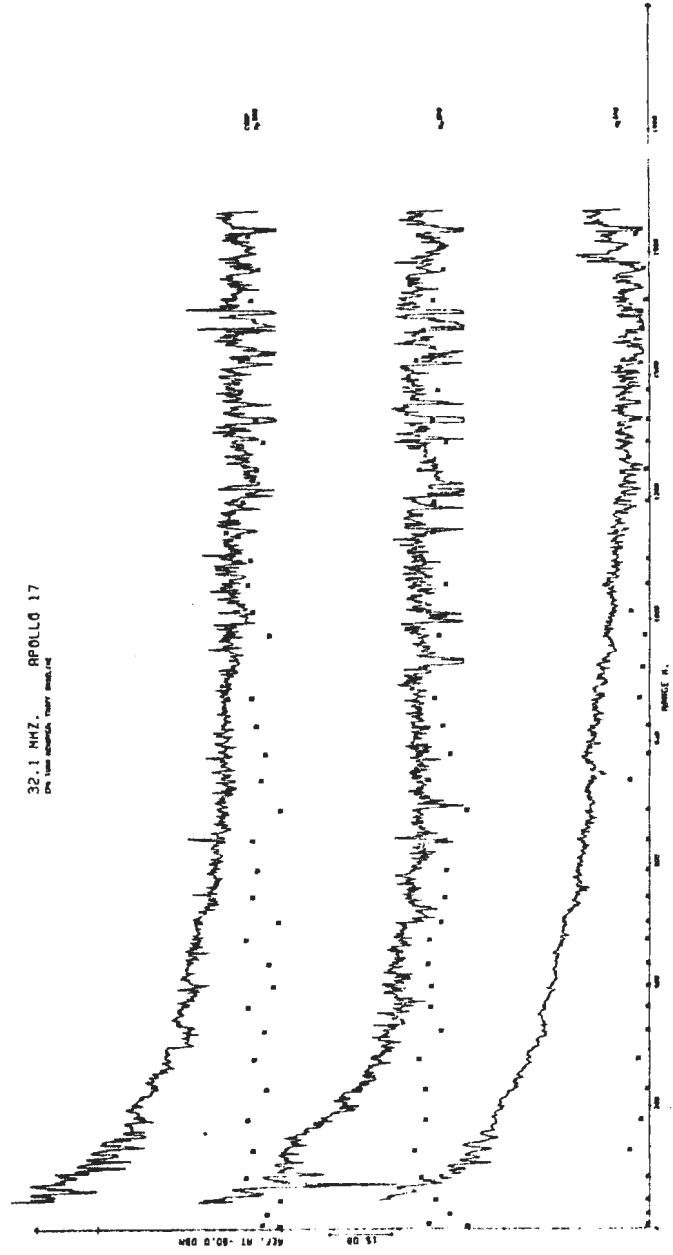
16.0 MHZ. APOLLO 17



32.1 MHZ. APOLLO 17
ON LINE SIGNAL, TERRY CHANNEL



32.1 MHZ. APOLLO 17
ON LINE SIGNAL, TERRY CHANNEL



A-7 (..)

MEMORANDUM

July 29, 1974

TO: Distribution

FPCM: J. C. Bylaarsdam

SUBJECT: Modifications to data on tape SEP09

As described in Watts' memorandum of July 2, 1974, a processing error during generation of data tapes SEP07 through SEP10 resulted in the loss of small amounts of dB data for 4, 8.1, 16, and 32.1 megahertz. These losses lead to erroneous correlation of the dB data with the range information, which was processed correctly. This memorandum describes a procedure for producing a set of data which is correctly matched, by removing range data corresponding to the dB data which were lost.

In the context of my report (Apollo 17 SEP Data Processing - July 1974) the processing is performed by program LUNACPY6, using file SCI2 as input. The modified data are designated as file SCI2M; this file is of exactly the same format as file SCI2, and contains all the same data, except for the changes described. File SCI2M is intended as a replacement for file SCI2 in any of the processing functions described in the report. (However, since some of the missing data occurred during the turn at EP-4, no processing by LUNAPLT4, LUNACPY4, or ANTENNA0 was attempted, and none is recommended using the modified data.)

The following is a recapitulation of Watts' description of the error, including one item which was not explained in his memorandum.

For each frequency, $M * 400$ data words were assembled in memory for each component, where $M = 1, 1, 2, 4, 8,$ and 13 for frequencies of $1, 2.1, 4, 8.1, 16,$ and 32.1 megahertz respectively. Then M blocks of length 387 words were defined, with origins of 1,

401, ..., $(M - 1) * 400 + 1$; the origins should have been 1, 388, ..., $(M - 1) * 387 + 1$. What is not made clear by Watts' memorandum is that the first M words of the first block did not contain data, and were discarded; hence M blocks, each of length 386 words, were written on the output tape. Then output block i would contain

- (a) the last $387 - (M - i + 1)$ words of block i , followed by
- (b) the first $M - i$ words of block $i + 1$.

N. B.

- (1) part (b) above does not apply to output block M ($M - i = 0$, and block $i + 1$ does not exist).
- (2) if the above-mentioned origins had been defined correctly, then the above definition of output blocks would yield the desired result.
- (3) In the cases where $M = 1$, the data on the output file are correct.
- (4) The last $(M - 1) * 13$ words of the last output block do not contain data.

The words which should have been used to assemble the output blocks are given in table 1; those which were used are given in table 2.

In the light of the above discussion the following procedure can be derived for matching the range data correctly to the erroneous dB data.

Having assembled the M blocks of length 386 in memory, define a set of "incorrect" block origins corresponding to those used in Watts' processing; since the M words are no longer present at the beginning of the data, this series is now $1 - M, 401 - M, \dots, 400 * (M - 1) + 1 - M$. Taking the length of these blocks as 387, a new set of M blocks, each of length 386, may be generated by selecting and

reassembling portions of the blocks as described in (a) above, and adding $13 * (M - 1)$ words of padding to the end of the last block.

The words used to assemble the blocks of modified range data are indicated in table 3.

A listing of program LUNACPY6 begins on page seven. Following the listing is an updated set of plots, designated SCI2BM, produced by LUNAPLT5 from file SCI2M.

Block	4 MHz.	8.1 MHz.	16 MHz.	32.1 MHz.
1	3- 388	5- 390	9- 394	14- 399
2	389- 774	391- 776	395- 780	400- 785
3		777-1162	781-1166	786-1171
4		1163-1548	1167-1552	1172-1557
5			1553-1938	1558-1943
6			1939-2324	1944-2329
7			2325-2710	2330-2715
8			2711-3096	2716-3101
9				3102-3487
10				3488-3873
11				3874-4259
12				4260-4645
13				4646-5031

Table 1 - Locations which should have been used to assemble blocks of dB data for file SCI2.

Block	4 MHz.	8.1 MHz.	16 MHz.	32.1 MHz.
1	3- 387 401	5- 387 401- 403	9- 387 401- 407	14- 387 401- 412
2	402- 774 (775- 787)	404- 787 801- 802	408- 787 801- 806	413- 787 801- 811
3		803-1187 1201	807-1187 1201-1205	812-1187 1201-1210
4		1202-1548 (1549-1587)	1206-1587 1601-1604	1211-1587 1601-1609
5			1605-1987 2001-2003	1610-1987 2001-2008
6			2004-2387 2401-2402	2009-2387 2401-2407
7			2403-2787 2801	2408-2787 2801-2806
8			2802-3096 (3097-3187)	2807-3187 3201-3205
9				3206-3587 3601-3604
10				3605-3987 4001-4003
11				4004-4387 4401-4402
12				4403-4787 4801
13				4802-5031 (5032-5187)

Table 2 - Locations which were used to assemble blocks of dB data for file SCI2. (Locations in parentheses contain meaningless information.)

Block	4 MHz.	8.1 MHz.	16 MHz.	32.1 MHz.
1	1- 385 399	1- 383 397- 399	1- 379 393- 399	1- 374 388- 399
2	400- 772 (773- 785)	400- 783 797- 798	400- 779 793- 798	400- 774 788- 798
3		799-1183 1197	799-1179 1193-1197	799-1174 1188-1197
4		1198-1544 (1545-1583)	1198-1579 1593-1596	1198-1574 1588-1596
5			1597-1979 1993-1995	1597-1574 1988-1995
6			1996-2379 2393-2394	1996-2374 2388-2394
7			2395-2779 2793	2395-2774 2788-2793
8			2794-3088 (3089-3179)	2794-3174 3188-3192
9				3193-3574 3588-3591
10				3592-3974 3988-3990
11				3991-4374 4388-4389
12				4390-4774 4788
13				4789-5018 (5019-5174)

Table 3 - Locations used to assemble blocks of range data for file SCI2M. (Locations in parentheses contain padding.)

```
*****  
*  
*                               LUNACPY6                               *  
*  
*****
```

```
C      PROGRAM TO GENERATE A MODIFIED VERSION OF FILE #2 , IN WHICH  
C      THE RANGE DATA CORRESPONDING TO MISSING DB INFORMATION HAVE  
C      BEEN DELETED.  
C
```

```
      INTEGER*4 MM(4) / 2, 4, 8, 13 /  
      REAL*4 DATA(6000)
```

```
C  
C      COPY ALL THE DATA WHICH REQUIRE NO MODIFICATION - I. E. THE  
C      LABEL RECORD THROUGH THE 2 MHZ. DATA.  
C
```

```
      DO 10 I = 1, 29  
         READ (1, 1000) (DATA(J), J = 1, 579)  
         WRITE(2, 1000) (DATA(J), J = 1, 579)  
10 CONTINUE
```

```
C  
C      LOOP OVER THE FOUR FREQUENCIES WHICH REQUIRE MODIFICATION.  
C
```

```
      DO 150 I = 1, 4  
         M = MM(I)  
         IORG = 1  
         IEND = 386
```

```
C  
C      READ THE M BLOCKS OF RANGE DATA INTO MEMORY.  
C
```

```
      DO 20 J = 1, M  
         READ (1, 2000) (DATA(K), K = IORG, IEND)  
         IORG = IORG + 386  
         IEND = IEND + 386  
20 CONTINUE
```

```
C  
C      ID IS INITIALIZED AS THE FIRST WORD OF THE FIRST GROUP  
C      OF 13 WORDS TO BE DELETED.  
C
```

```
      ID = 388 - M
```

```
C  
C      MM1 IS THE NUMBER OF GROUPS OF 13 WORDS TO BE DELETED.  
C
```

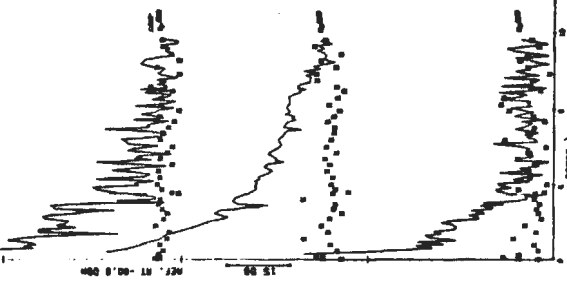
```
      MM1 = M - 1
```

```
C  
C      N IS THE NUMBER OF WORDS TO BE MOVED FROM THE BEGINNING OF  
C      BLOCK J + 1 TO THE END OF BLOCK J. (INITIALLY M - 1)  
C
```

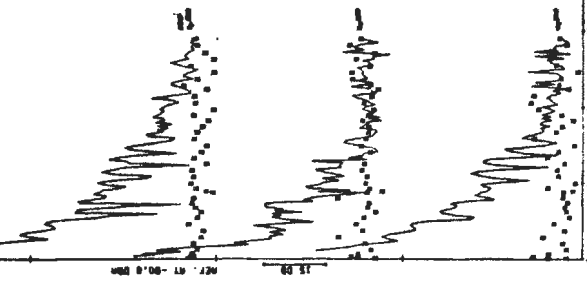
```
C
C
C      N = MM1
C
C      LOOP OVER THE SET OF 13-WORD GROUPS.
C
C      DO 50 J = 1, MM1
C
C          TRANSFER N WORDS FROM BLOCK J + 1 TO BLOCK J.
C
C          DO 40 K = 1, N
C              JD = ID + K - 1
C              JS = JD + 13
C              DATA(JD) = DATA(JS)
40      CONTINUE
C
C          FOR GROUP J + 1, N IS DECREMENTED BY 1.
C
C          N = N - 1
C
C          THE BEGINNING OF GROUP J + 1 IS 400 WORDS FROM THE
C          BEGINNING OF GROUP J.
C
C          LL = ID + 400
C
C          JD IS CURRENTLY THE INDEX OF THE 386TH WORD OF OUTPUT BLOCK
C          J; SET JS TO INDEX THE 1ST WORD, AND THEN WRITE THE BLOCK
C          ON THE OUTPUT FILE.
C
C          JS = JD - 385
C          WRITE(2, 2000) (DATA(K), K = JS, JD)
C          WRITE(6, 3000) (DATA(K), K = JS, JD)
60      CONTINUE
C
C          COMPLETE OUTPUT BLOCK M BY PLACING THE LAST CORRECT RANGE
C          VALUE (CONTAINED IN WORD N)
C
C          N = 386 * M
C
C          IN THE MN LOCATIONS BEGINNING AT LOCATION N + 1.
C
C          NN = 13 * MM1
C          DO 80 J = 1, NN
C              JD = J + J
C              DATA(JD) = DATA(N)
80      CONTINUE
C
C          THE FIRST WORD OF THE OUTPUT BLOCK IS COMPUTED AS ABOVE,
C          AND THE BLOCK IS WRITTEN ON THE OUTPUT FILE.
```

```
C
      JS = JD - 385
      WRITE(2, 2000) (DATA(K), K = JS, JD)
      WRITE(6, 3000) (DATA(K), K = JS, JD)
C
C     REDEFINE N AS THE LAST VALID WORD (WITHIN THE LAST BLOCK)
C     OF DB DATA
C
      N = 386 - NN
C
C     FOR EACH COMPONENT,
C
      DO 140 IC = 1, 6
C
C         READ M BLOCKS OF DB DATA.
C
          DO 120 J = 1, M
              READ (1, 2000) (DATA(K), K = 1, 386)
C
C             WRITE BLOCKS 1 THROUGH M - 1 ON THE OUTPUT FILE IMMEDIATELY.
C
              IF (J .NE. M) GO TO 110
C
C             FILL THE LAST NN WORDS OF BLOCK M WITH PADDING
C             BEFORE WRITING IT ON THE OUTPUT FILE.
C
                  DO 100 K = 1, NN
                      JD = M + K
                      DATA(JD) = -135.
100          CONTINUE
110          CONTINUE
              WRITE(2, 2000) (DATA(K), K = 1, 386)
120          CONTINUE
140          CONTINUE
150          CONTINUE
          RETURN
C
1000  FORMAT(200A4, 200I4, 179A4)
2000  FORMAT(200F6.1, 180F6.1)
3000  FORMAT('0 ' / ' ', 15F8.1 / 25('X, 15F8.1/))
      END
```

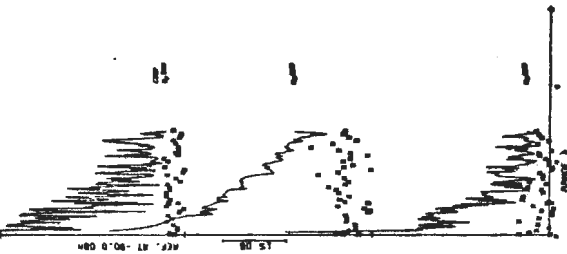

2.1 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY AREA, 1964



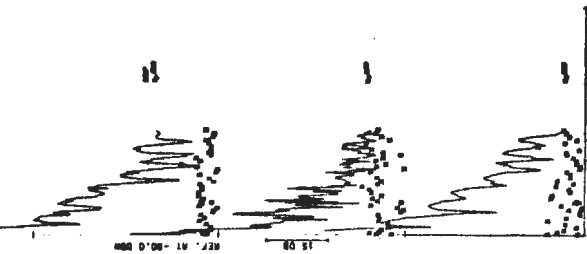
2.1 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY AREA, 1964



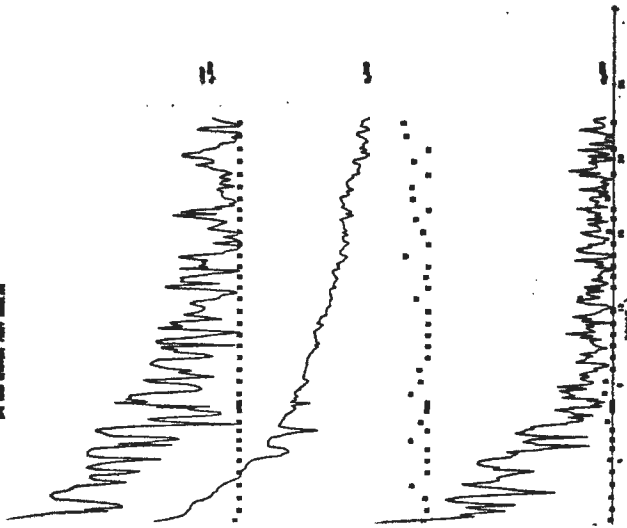
1.0 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY AREA, 1964



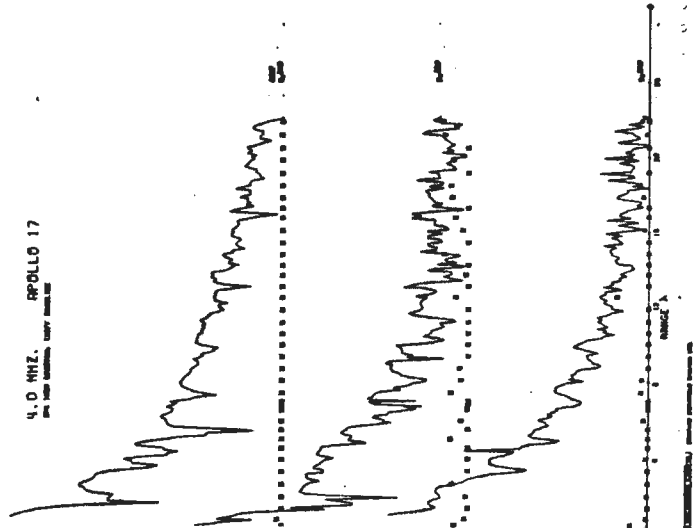
1.0 MHZ. APOLLO 17
ONE YEAR RECORD, TERRY AREA, 1964



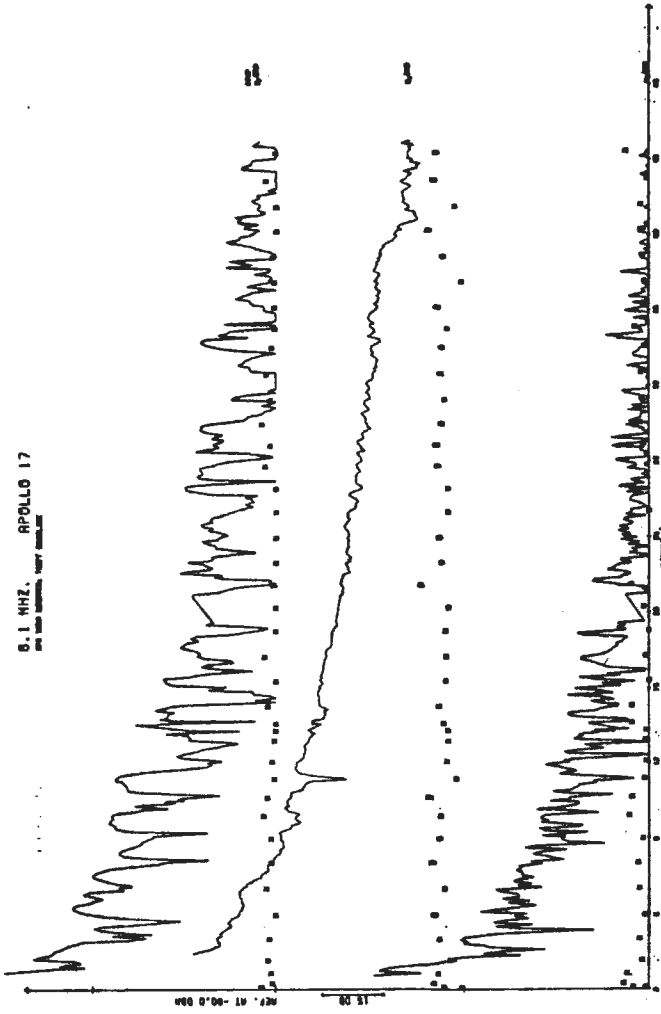
4.0 MHZ. APOLLO 17
PERIOD: 0.000 - 0.000



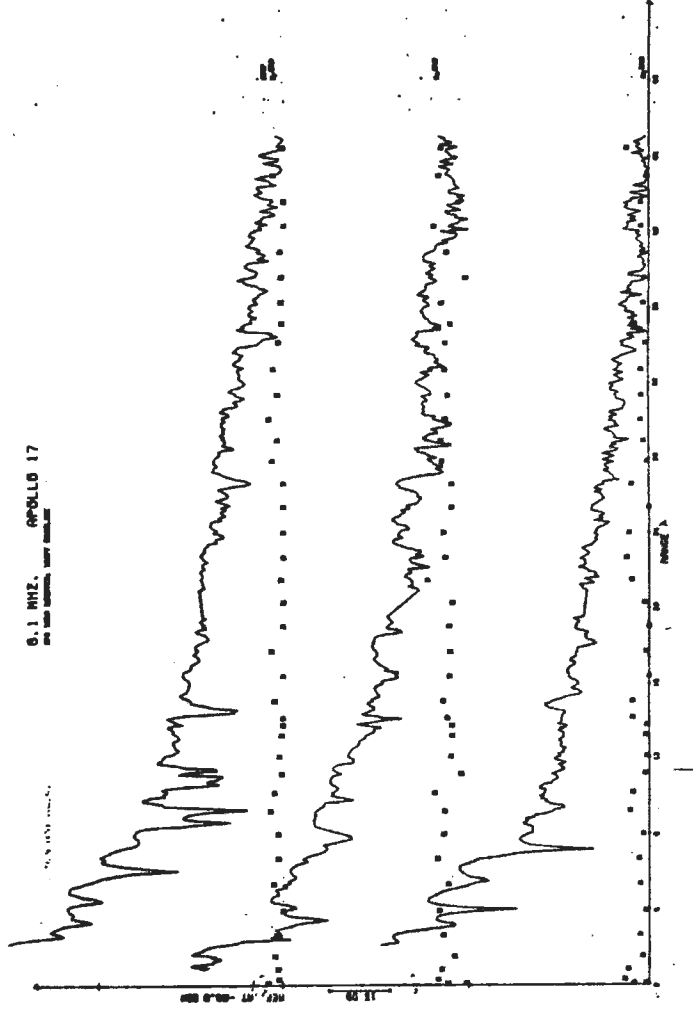
4.0 MHZ. APOLLO 17
PERIOD: 0.000 - 0.000



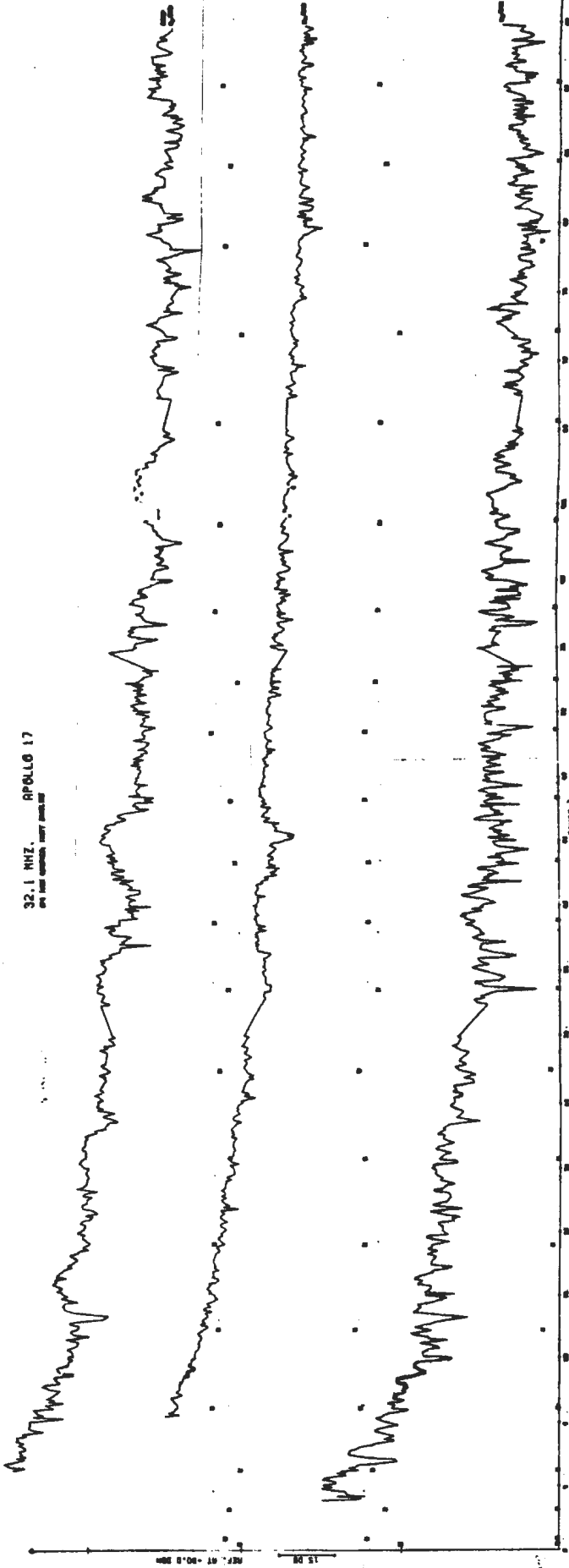
0.1 MHZ. APOLLO 17
PERIOD: 0.000 - 0.000



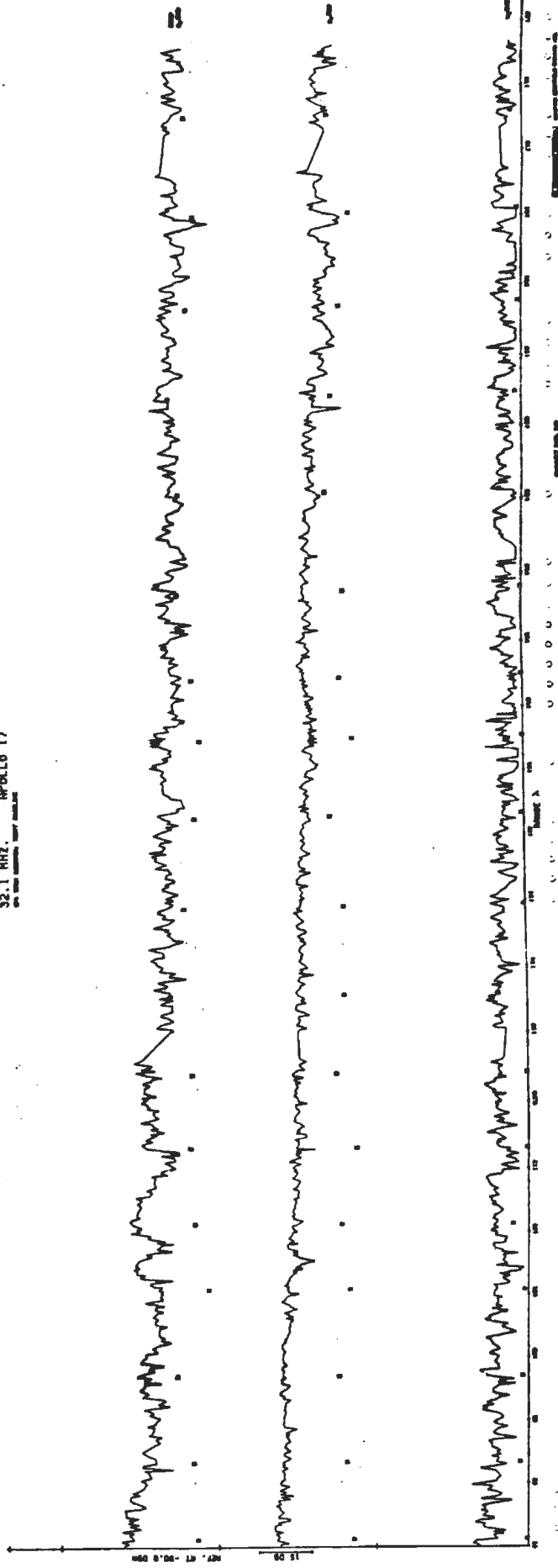
0.1 MHZ. APOLLO 17
PERIOD: 0.000 - 0.000



32.1 MHZ. APOLLO 17
ON THE SURFACE VERY SENSITIVE

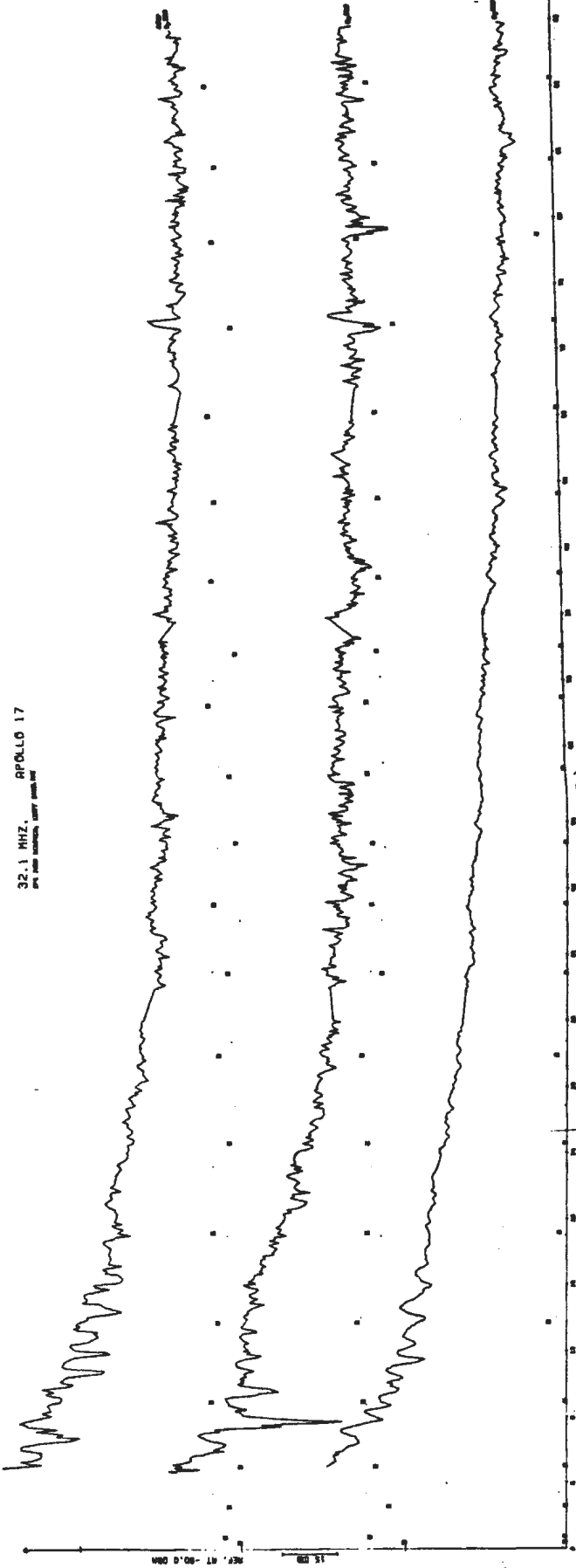


32.1 MHZ. APOLLO 17
ON THE SURFACE VERY SENSITIVE

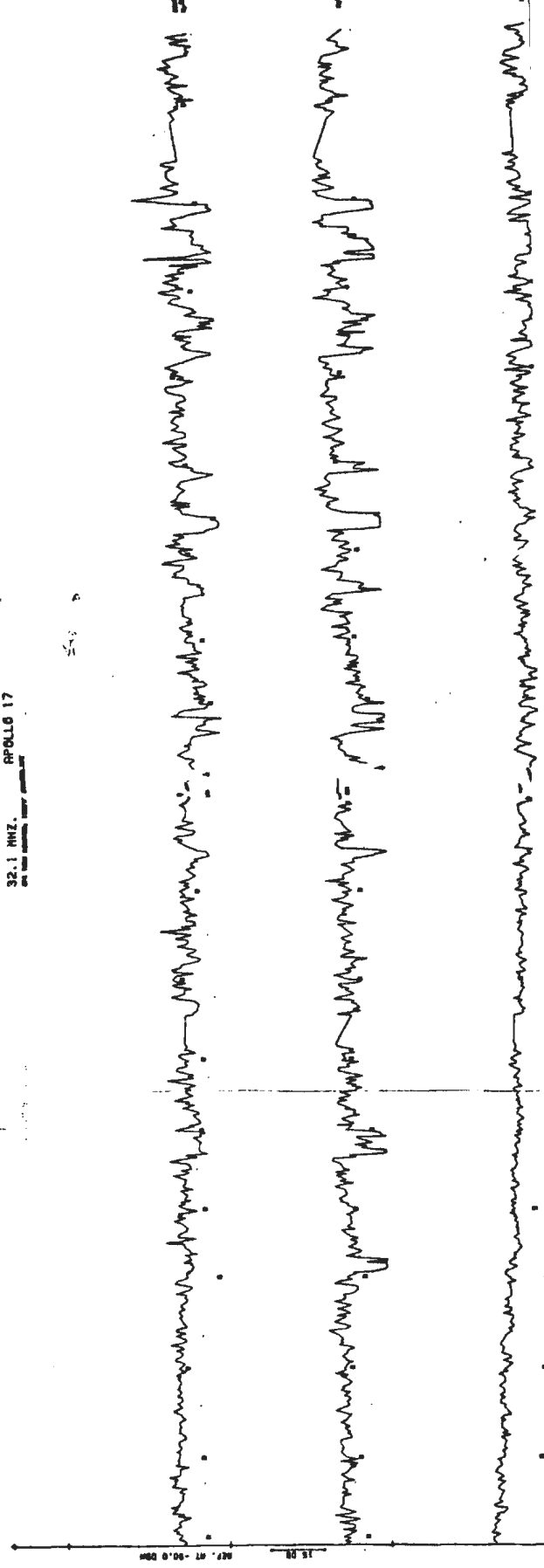


1100

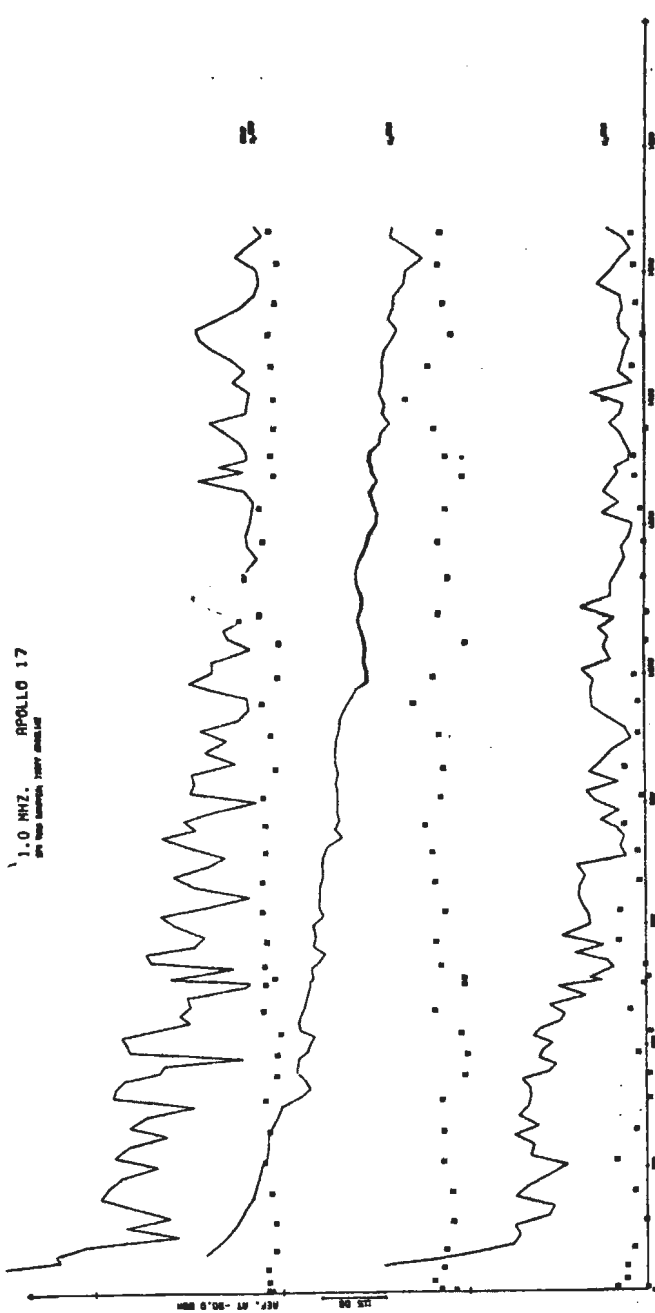
32.1 MHZ. APOLLO 17
ON THE SURFACE, LANTANA 100



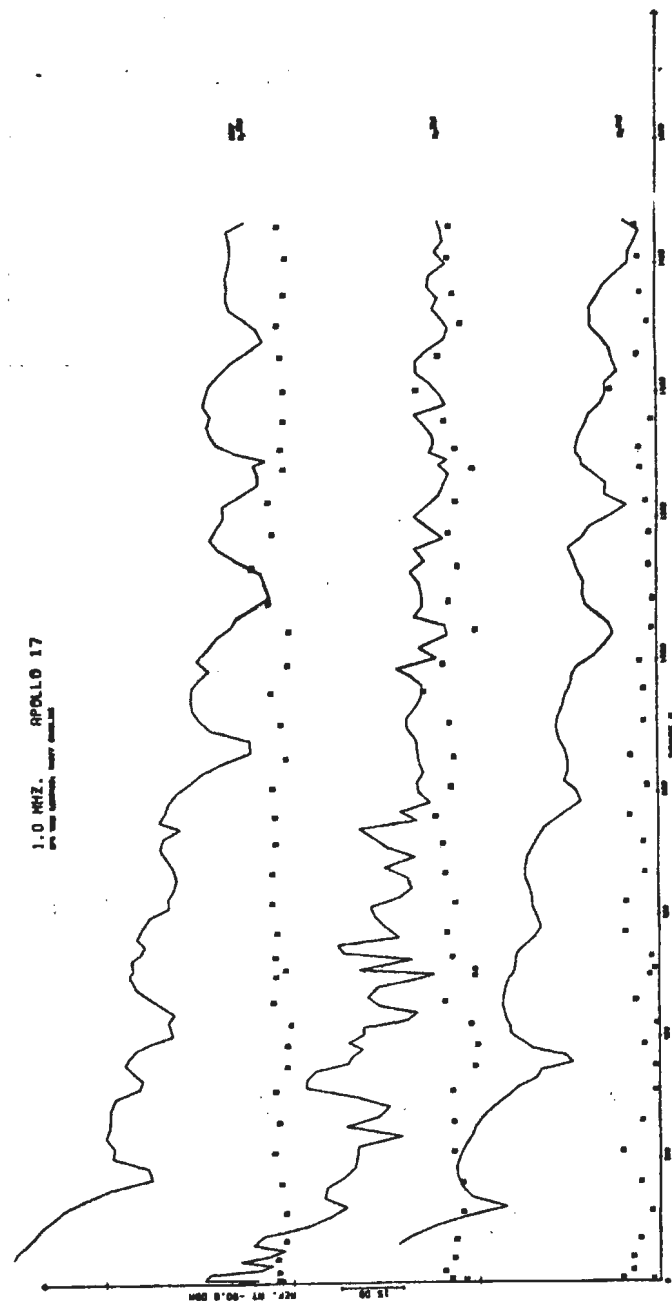
32.1 MHZ. APOLLO 17
ON THE SURFACE, LANTANA 100



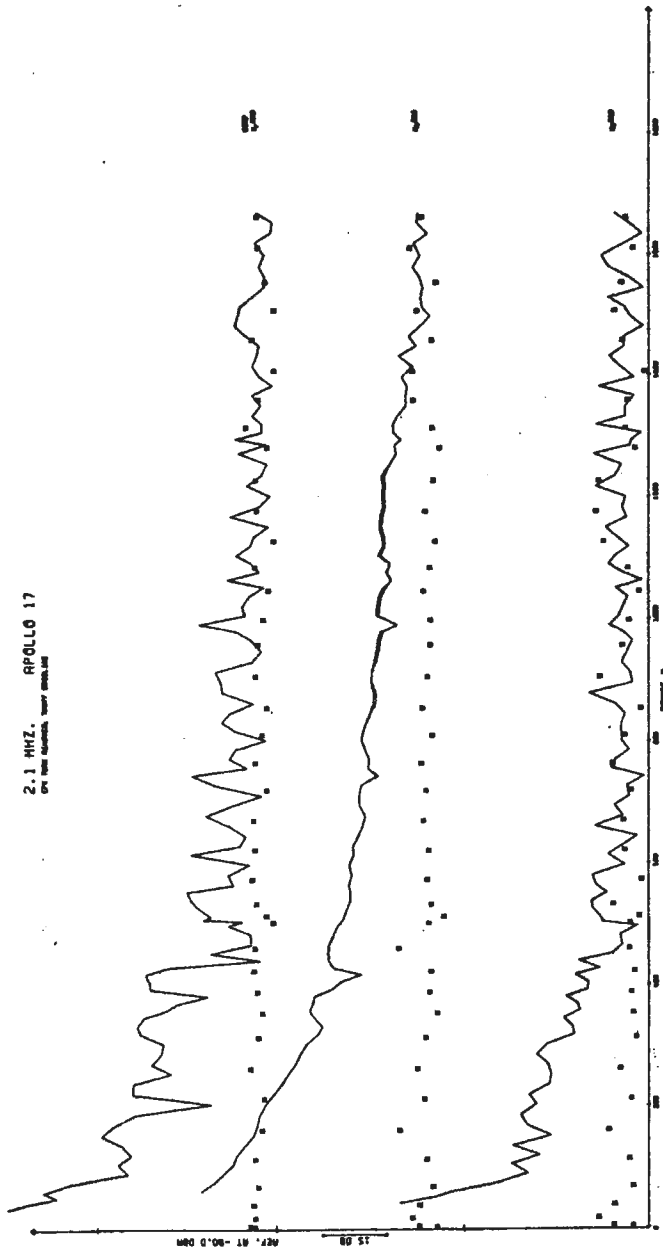
1.0 MHZ. APOLLO 17
ON THE SURFACE, 1969 JANUARY



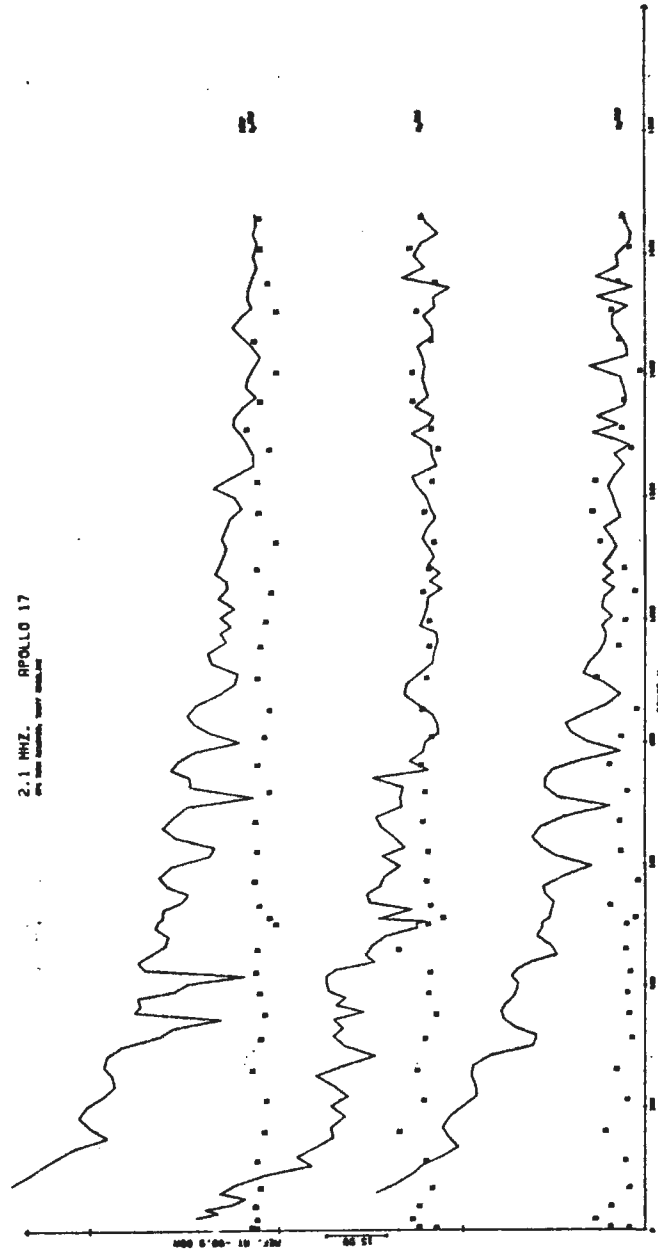
1.0 MHZ. APOLLO 17
ON THE SURFACE, 1969 JANUARY



2.1 MHZ. APOLLO 17
On the ground, near antenna

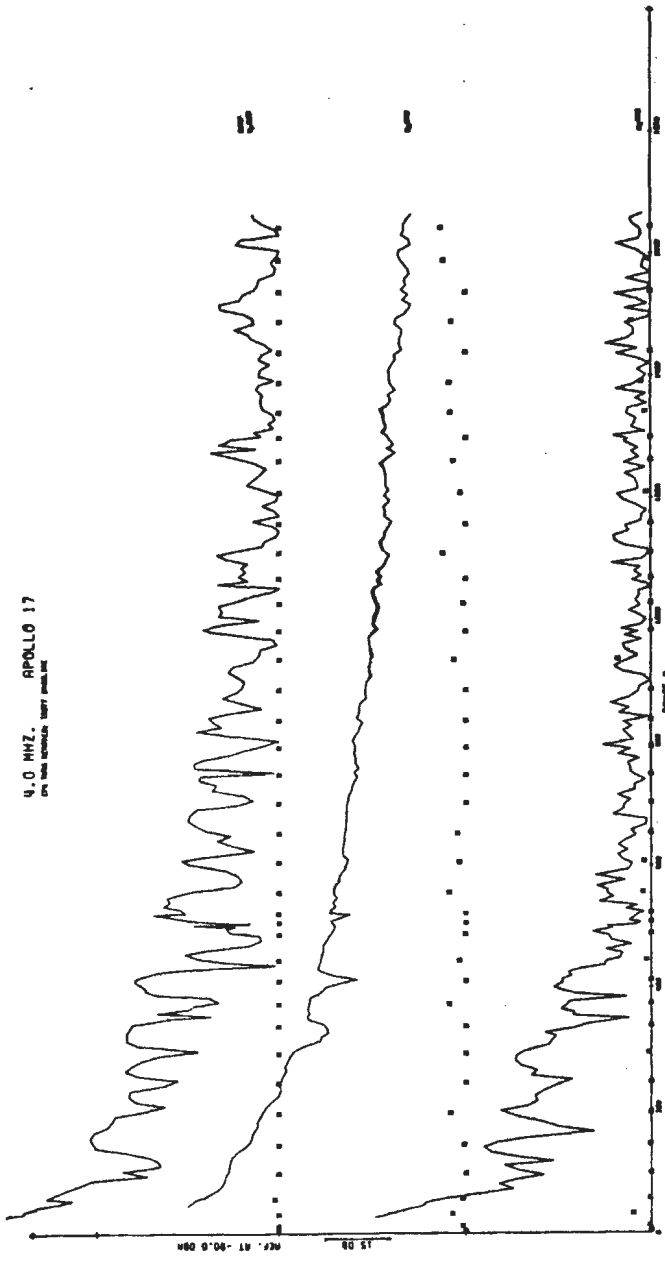


2.1 MHZ. APOLLO 17
On the ground, near antenna

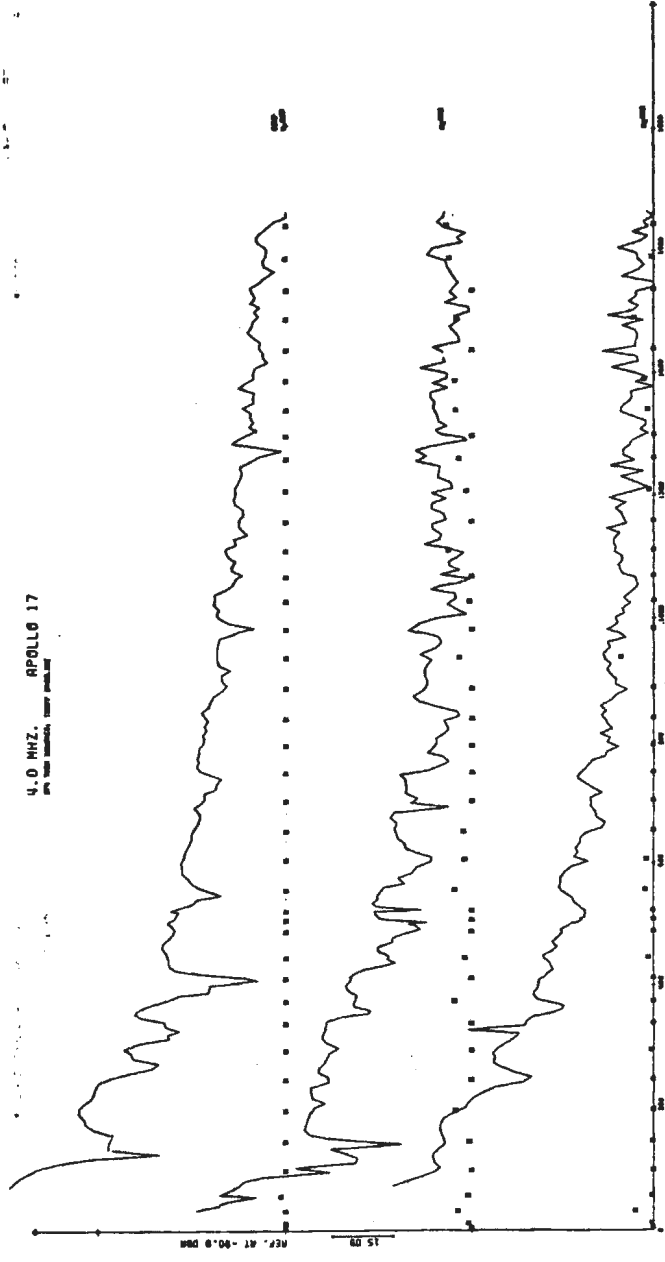


11-11

4.0 MHZ. APOLLO 17
ON THE SURFACE, TERRY PROBE 102



4.0 MHZ. APOLLO 17
ON THE SURFACE, TERRY PROBE 102

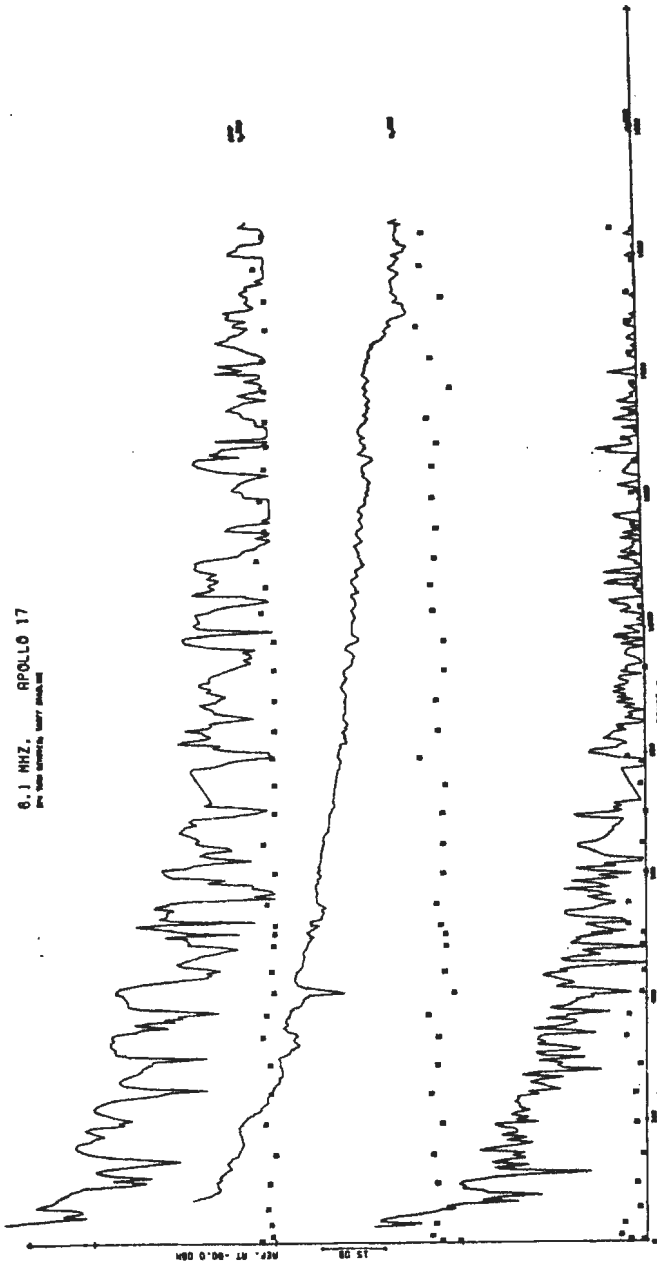


APOLLO 17 SURFACE TRANSMITTER

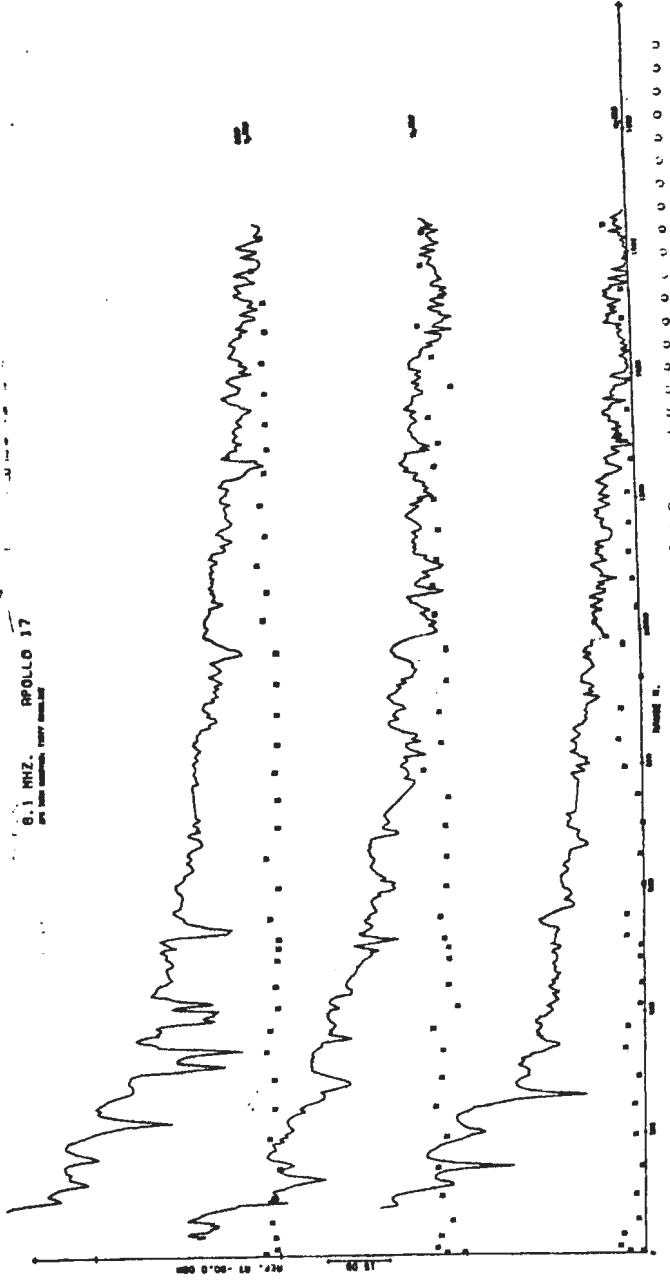
GROUND STATION

APOLLO 17 SURFACE TRANSMITTER

6.1 MHZ. RPOLLO 17
PER UNO SCALARE PER UNO SCALARE

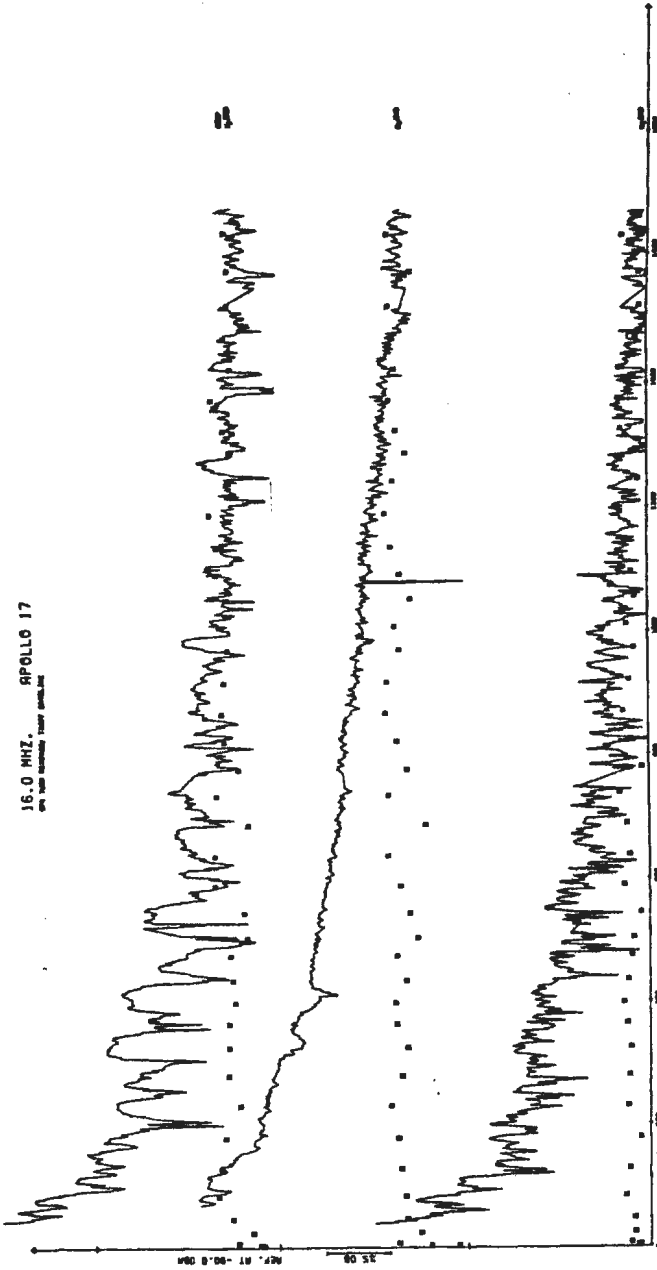


6.1 MHZ. RPOLLO 17
PER UNO SCALARE PER UNO SCALARE



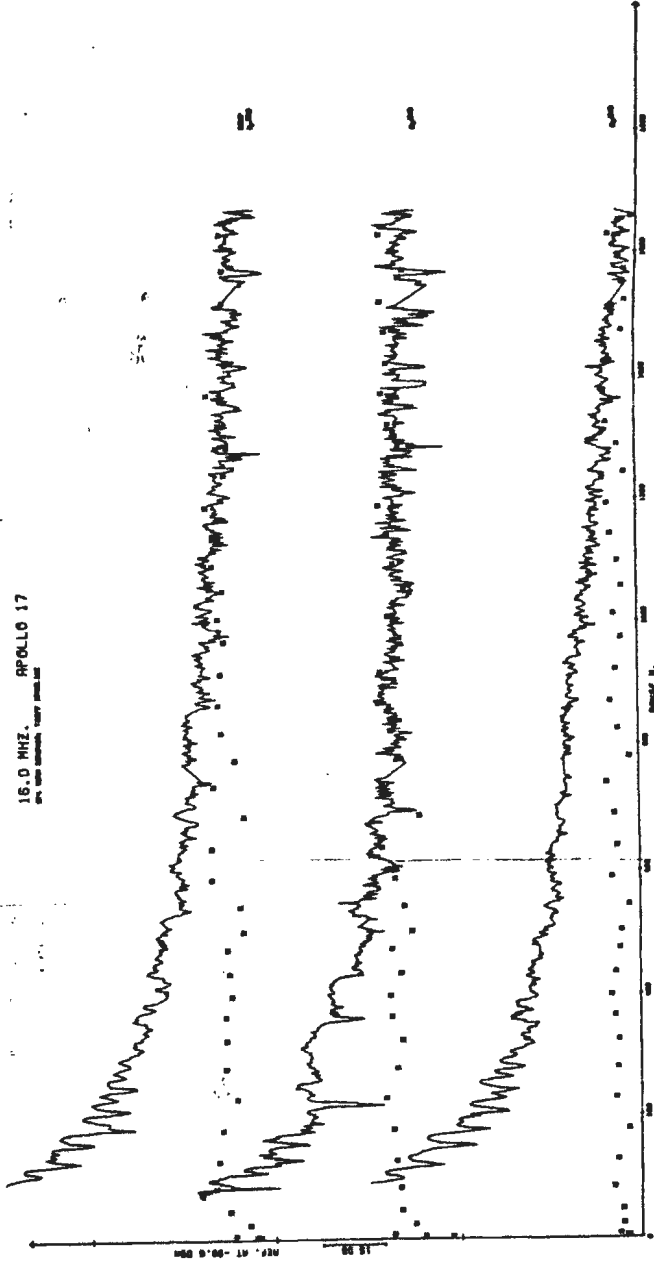
15.0 MHZ. SPOLLO 17

ON LINE RECORDING TAPED RECORD



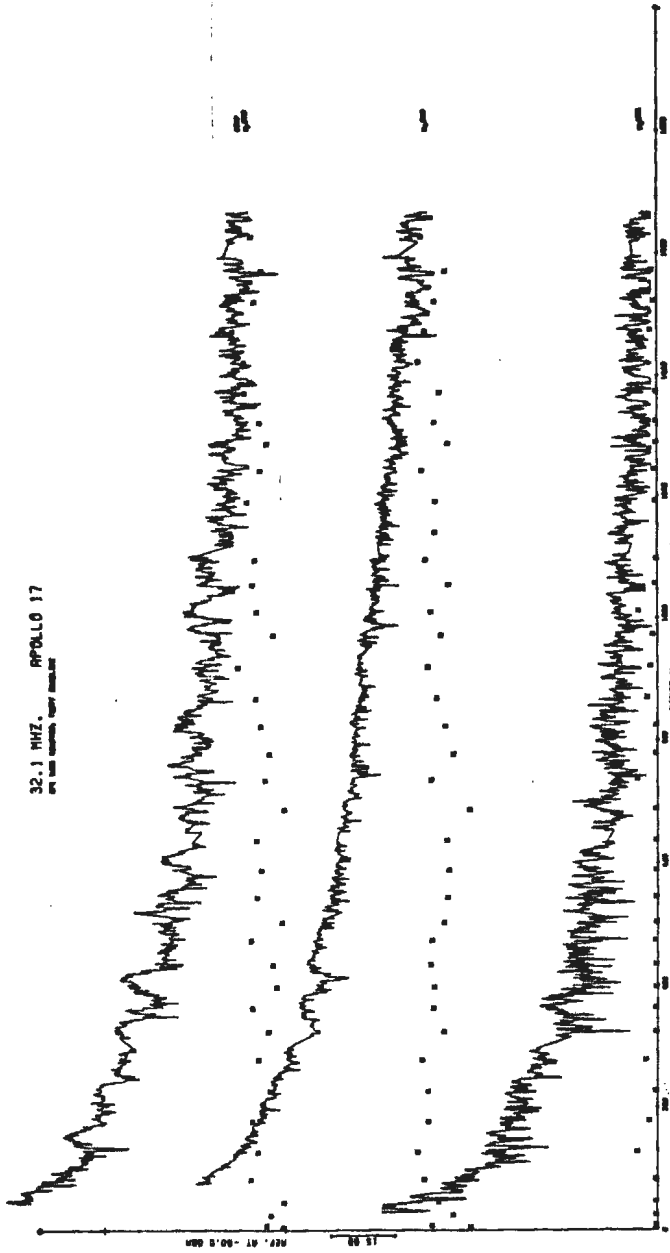
15.0 MHZ. SPOLLO 17

ON LINE RECORDING TAPED RECORD

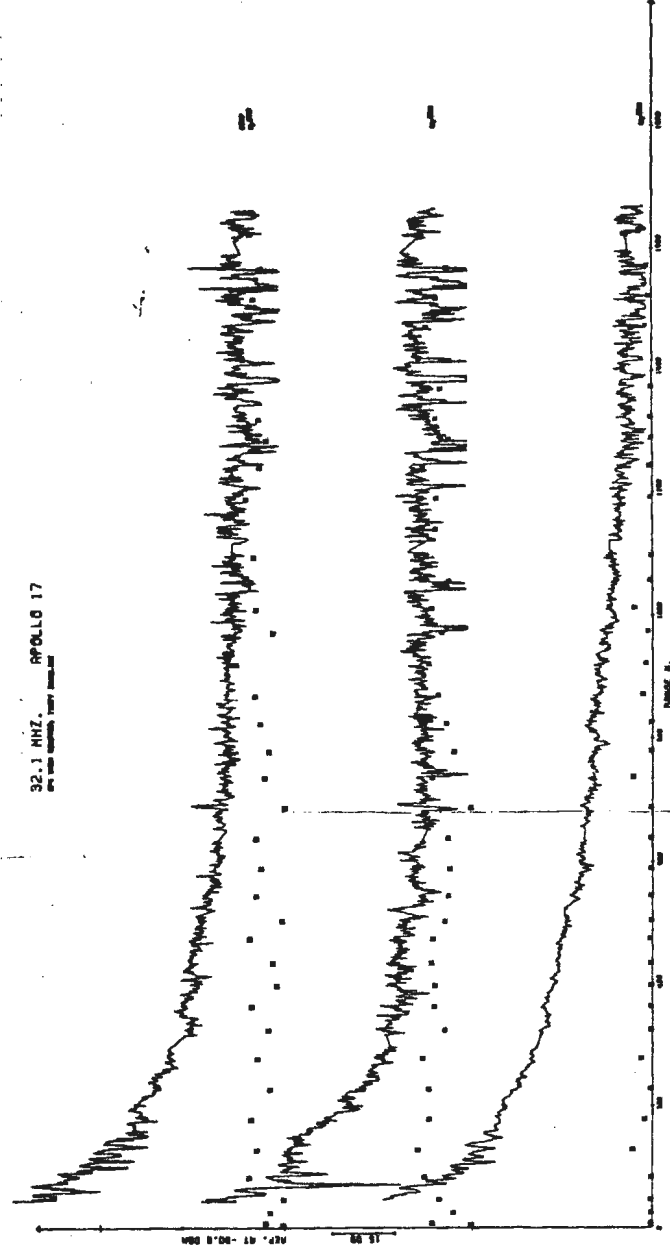


11 11 11

32.1 MHZ. APOLLO 17
ON THE SURFACE, TAPER RECORD



32.1 MHZ. APOLLO 17
ON THE SURFACE, TAPER RECORD



MEMORANDUM

July 22, 1974

TO: Distribution

FROM: J. C. Rylaarsdam

SUBJECT: Comparison of SFP Range Data and Data
from the VLBI Experiment

The VLBI data used for this study were obtained from tape number G2IMS (Goddard Space Flight Centre) as a set of x-y coordinate pairs; associated with the first pair in each group of five was a time (Greenwich Mean) expressed in hours, minutes, and seconds. These times were converted to seconds, and times for the four remaining pairs in each group were generated by adding values of one, two, three, and four seconds to the initial value. The x-y pairs were converted to distances.

The SFP data consisted of the sixteen megahertz range values from tape number SEP009. The time (GM) corresponding to the first datum was set at 1407.4 seconds; times for succeeding values were generated by repeated addition of 0.81 second, the time interval between samples.

For each VLBI range datum, a corresponding SFP range value was computed by linear interpolation, using the two arrays of time data; the difference between these ranges was calculated as the VLBI range minus the interpolated SFP range. The differences obtained are plotted in figure 1; in this plot, the data are grouped into ten-second intervals, and the maximum and minimum differences over each interval are displayed. The mean of these differences was found to be -23.94 metres (indicating a lag in the VLBI data), and the standard deviation of the differences was effectively zero, considering the limits of precision of the calculations.

A more direct visual comparison is provided by figure 2, in which both sets of data are plotted on a single set of axes.

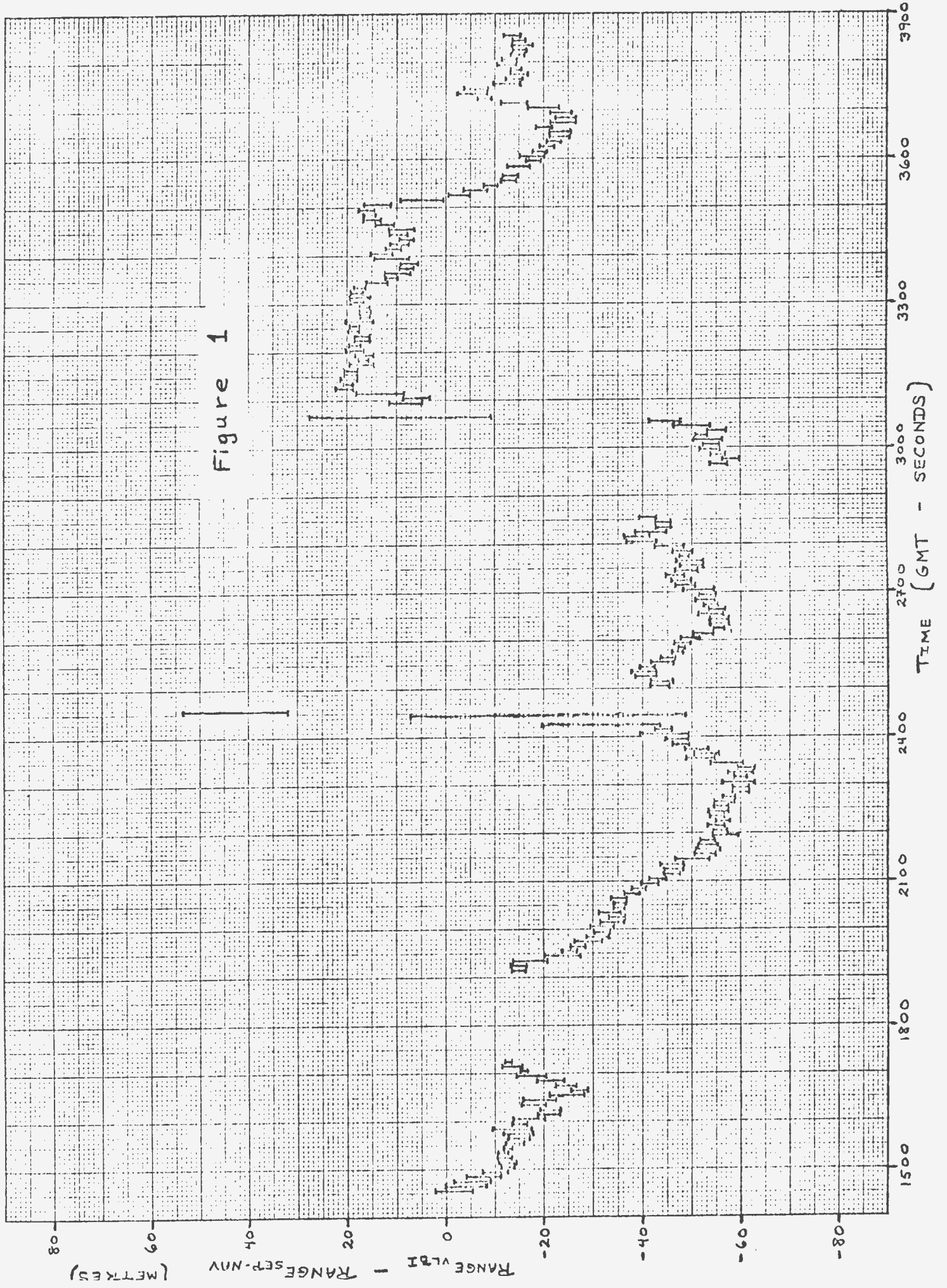


Figure 1

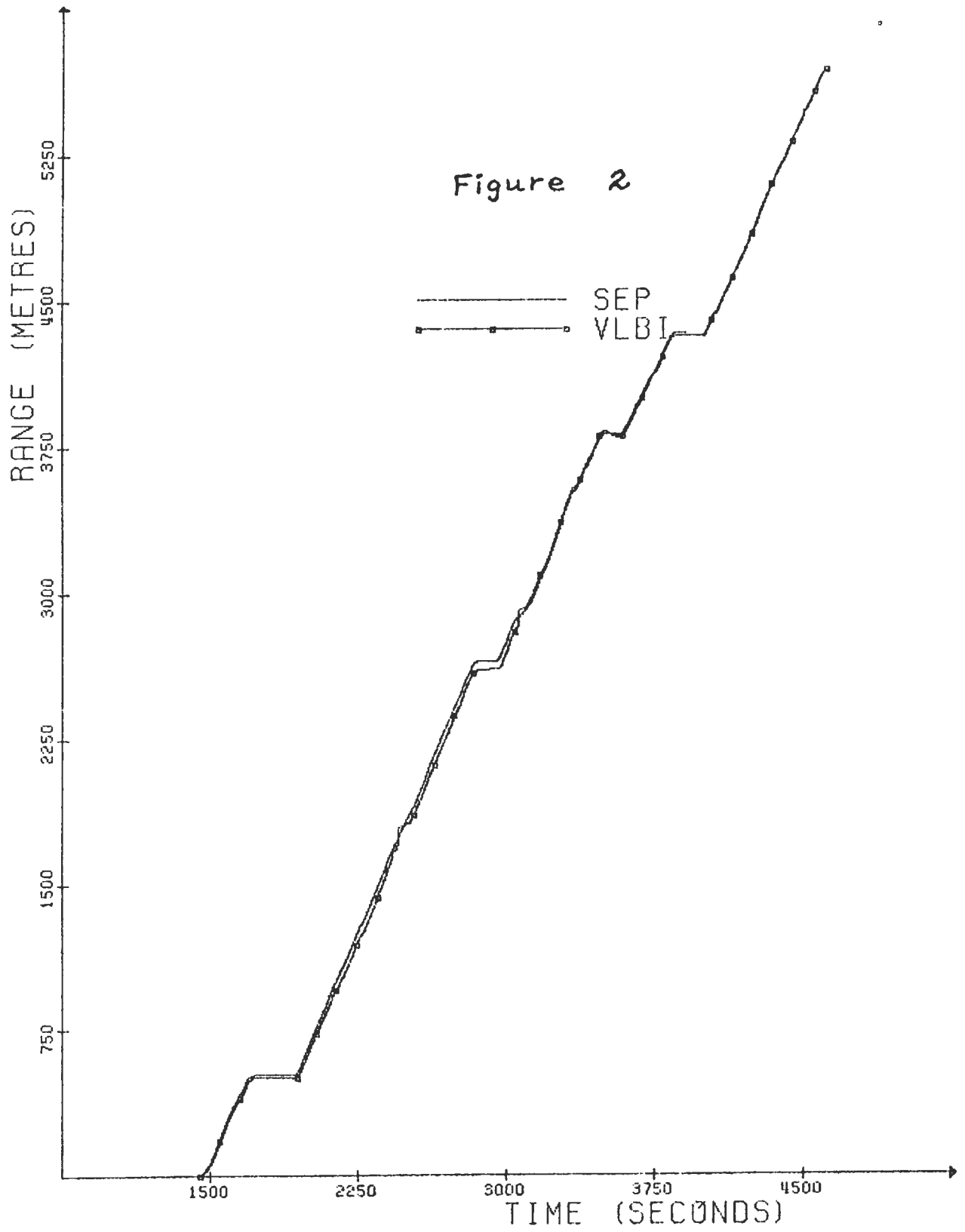


Figure 2

SEP
VLBI

July 22, 1974

MEMO TO: D.W. Strangway
FROM: James Rossiter
RE: SEP Antenna Patterns Reconstructed from EP-4 Turn

Introduction

During EVA II of Apollo 17, the Lunar Roving Vehicle (LRV) made a complete 360° turn around the deployment site of Seismic Explosion Package 4 (EP-4), about 525 m. from the SEP transmitter site. This turn provided an opportunity to estimate the directional characteristics of the three orthogonal SEP loop receiving antennas, as mounted on the LRV, over a dielectric earth.

Ideally, any signal received by the H_r (radial) antenna should smoothly interchange with the signal received by the H_ϕ (tangential) antenna as the LRV goes through each 90° of the turn. The H_z (vertical) signal should remain constant throughout the turn. If the turn were of zero radius, any deviations from the above could then be attributed to interference by the Rover and/or mount.

Data Reduction

Data were taken from Watt's lunar tape SEPD09, which included the error noted in his memo (July 2, 1974). Details of the organization of the data after their removal from tape are given by Rylaarsdam ("Apollo 17 SEP Data Processing", July 1974). The following steps were then taken in order to

get antenna patterplots.

(1) The values of all 36 components over the entire range of the turn (493 to 538 m. from the SEP site) were removed from Rylaarsdam's file SCI3 (which included no pre-processing of the turn by Watts). These values were stored in a new file called EP4, listed using program LUNACPY4, and plotted using program LUNAPLT4 (and routine GAPLOT). The points were spaced according to the time scale implicit in the data; an example is shown in Figure 1.

(2) Odometer counts (one count = 0.49 m. of wheel turn), received from both the right front and left rear wheels of the LRV, are available for each 1.02 seconds of the traverse (see memo by Redman, July 16/73). Ideally, given a high density of odometer pulses, and assuming no wheel slippage or sticking, LRV speed and rate of turn could be completely determined. However, the coarseness of the odometer pulses prevented this detailed reconstruction (see Figures 2 and 3). Antenna patterns plotted using the navigation data (see Rylaarsdam's report) were far less consistent from component to component than were those plotted assuming the LRV speed to be constant during the non-stationary portions of the turn.

Therefore a template with three pairs of bounds was set up to separate the points that were recorded while the LRV was actually turning from those recorded while the LRV was either

on its traverse leg or stopped. By using components that had a good deal of character, the times during which the LRV was stationary were easily distinguished on the set of plots like Figure 1. The end-points of the turn were more difficult to estimate, and consistency from component to component was the only criterion available.

Unfortunately the 16 and 32 MHz data could not be used to construct the template, since both of these frequencies contain a drop-out due to Watt's spooling error during the turn.

(3) The total angle through which the LRV turned was calculated in the following way:

Assume there is no net slippage or sticking of either wheel over the turn. Then, for each wheel,

$$c = n\pi r, \quad (1)$$

where c = the circumference of the turn made by the wheel
= total number of counts x 0.49 meters;
 $n\pi$ = number of radians of the turn; and,
 r = radius of the turn (m.).

Therefore,

$$n\pi = \frac{(c_o - c_i) \times 0.49}{r_o - r_i} \quad (2)$$

where $c_o - c_i$ = the difference in odometer counts between the two wheels over the turn (see Figure 3); and,
 $r_o - r_i$ = the distance between the two wheels
= 1.73 m. (Apollo 17 LRV Manual).

For the turn, $c_o - c_i = 21 \pm 2$,

therefore, $n\pi = 6.0 \pm 0.5$ radians.

Although this is evidently a fairly crude estimate, it indicates that the turn was close to 360° .

(4) The portions of file EP4 determined by step 2 to be actually in the turn proper were plotted as a function of angle using program ANTENNAO (and routine ANTPAT). A complete set of patterns is shown in Figure 4. The angles start along the negative x-axis, and increment uniformly clockwise over 2π radians.

Discussion

Basically the plots show the expected type of behaviour. The vertical components are fairly smooth (except those which have very low signals), with few lobes, while the H_r and H_ϕ components do interchange. It must be pointed out that the 16 and 32 MHz plots do not contain any angular correction for the missing points, and this will certainly create some amount of distortion in the patterns.

Several of the plots do not align well with the north-south and east-west axes - e.g. 4MHz H_ϕ endfire. This is possibly due to an incorrect choice of either the bounds or of the total angle.

The major obstacles in obtaining good patterns from this

analysis are as follows:

- (1) Very poor sampling for the lower frequencies (as few as 8 points for a complete turn at 1 and 2 MHz), giving virtually no resolution of any lobe structure.
- (2) Non-constant range of the LRV through the turn for the higher frequencies. The turn had a diameter of approximately 15 m - or about 1.5 wavelengths at 32 MHz. Therefore the signal received during the turn could have changed substantially quite independently of LRV rotation.
- (3) Lack of direct knowledge of a) the exact position of the turn in the data stream, b) the complete angle of rotation, or c) the speed of rotation. These could only be estimated, and compatibility from component to component used to improve the estimate. The problem was particularly severe because of the drop-outs at 16 and 32 MHz.
- (4) Unknown source signal. It is evidently not a plane wave, since the SEP transmitter was used. Reflections and scattering from the subsurface may well have had important influences on the type of pattern.

Conclusions

Considering the above problems, the amount of distortion of the patterns is within the error of the analysis. H_z appears non-directional at all frequencies; H_r and H_ϕ interchange smoothly through the turn. It is therefore not possible to attribute any large degree of interference to the

LRV or mount. This does not imply that such interferences did not exist - only that this analysis was not able to detect it.

It would probably be worth while in the future to analyse the data without the drop-outs at 16 and 32 MHz. These frequencies have both the highest resolution and are most likely to be susceptible to interference from the LRV or mount. A systematic attempt to use a number of different possible bounds and rotation angles may locate the turn in the data stream better. If so, such a study could be more definitive.

272

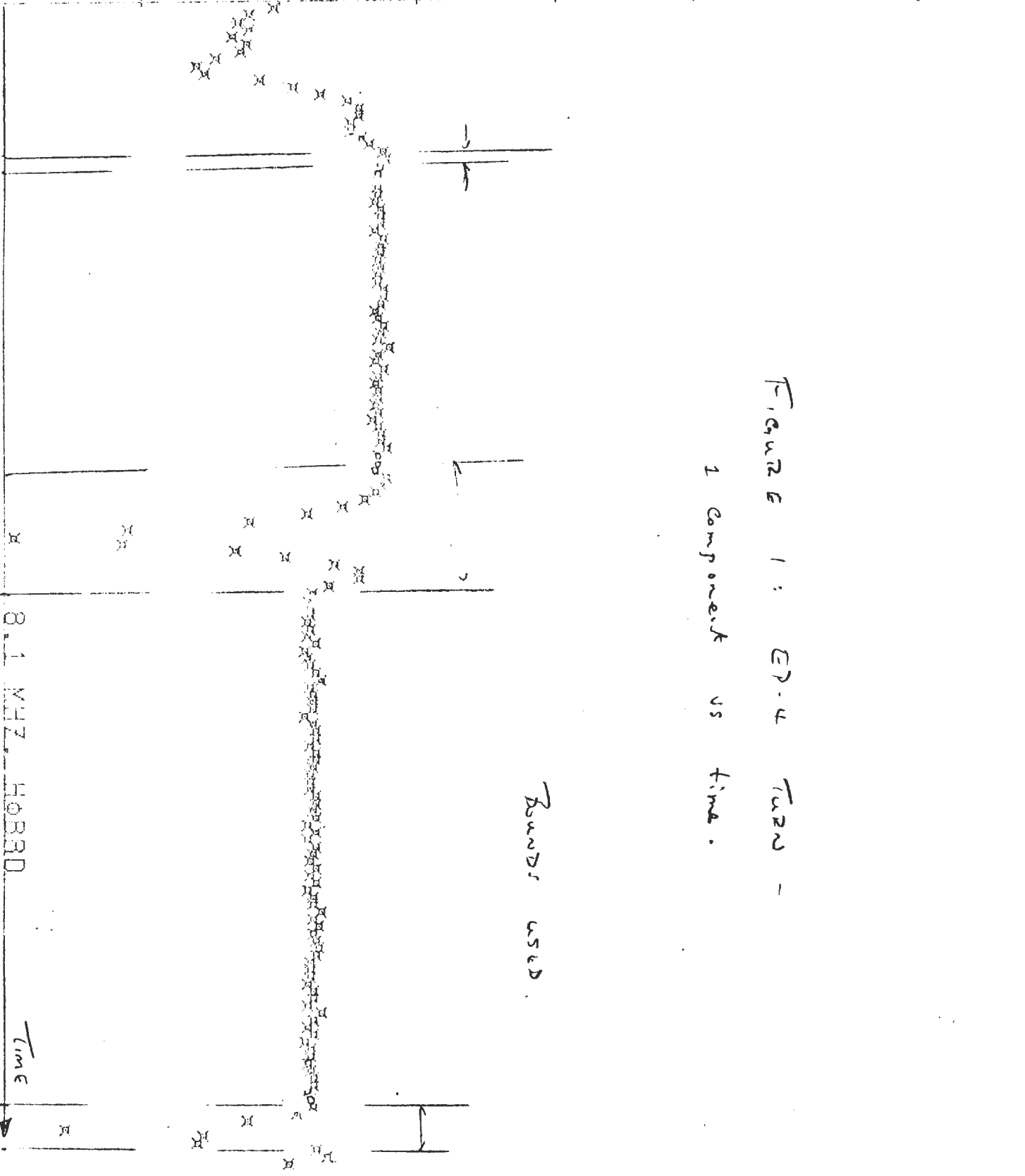
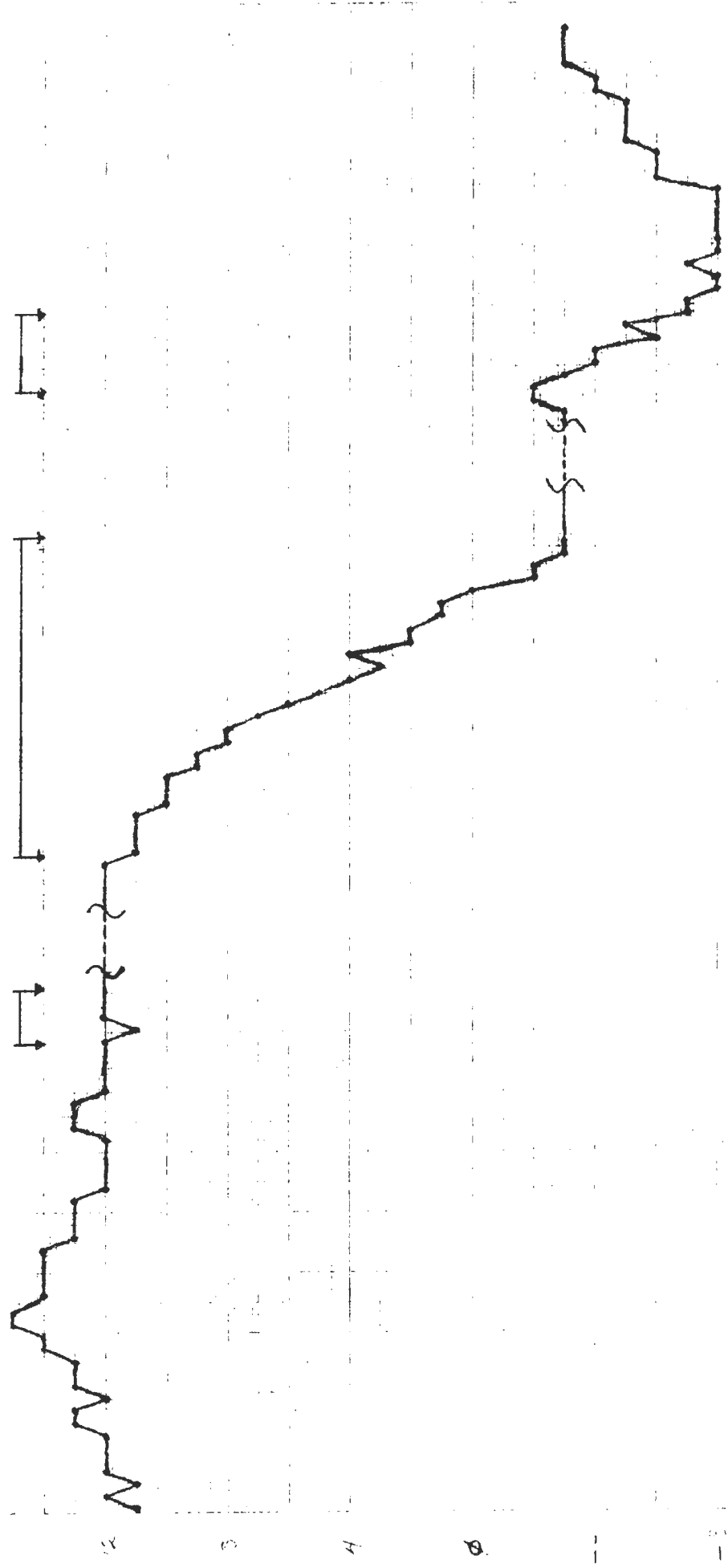


FIGURE 1: EP-4 TURN -
1 Component vs time.

Bands used.

7000
LRN WHEELS - EP-U TURN.

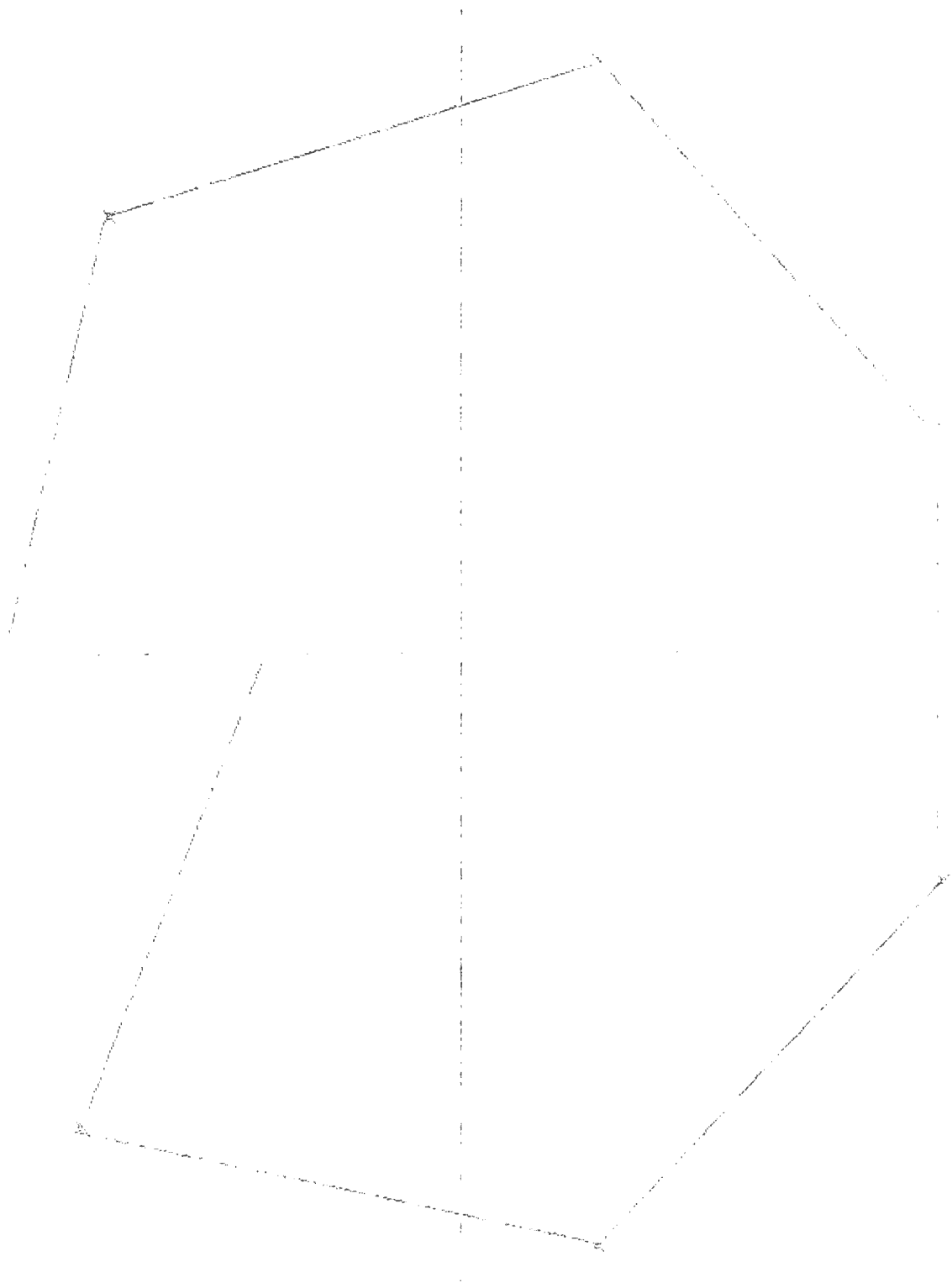
BOUNDS USED



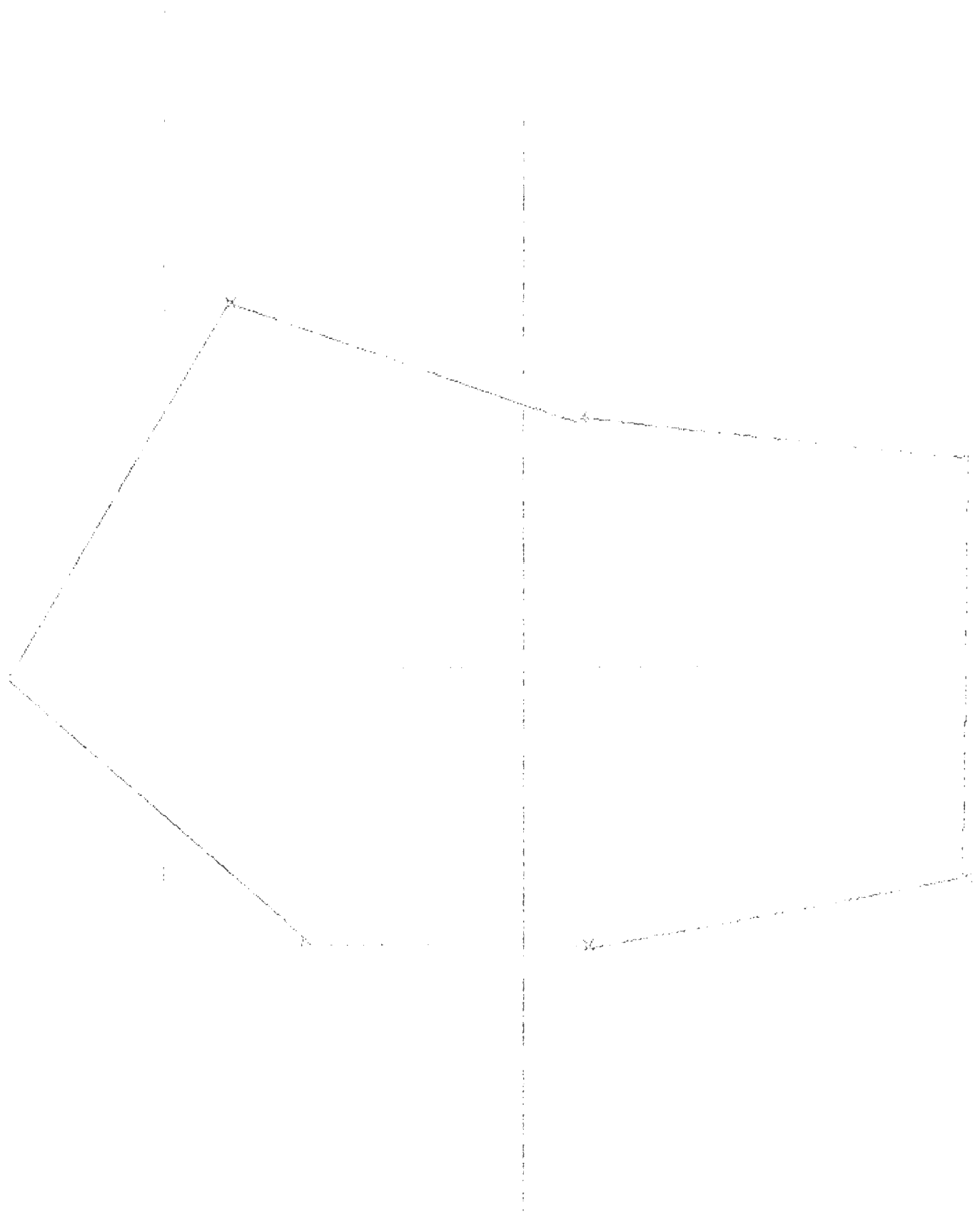
7000
LRN WHEELS - EP-U TURN.

Figure 4. SEP antenna radiation patterns from EP-4 turn for all components. Since choice of the exact position of the turn is somewhat arbitrary (see text), these patterns are only approximate.

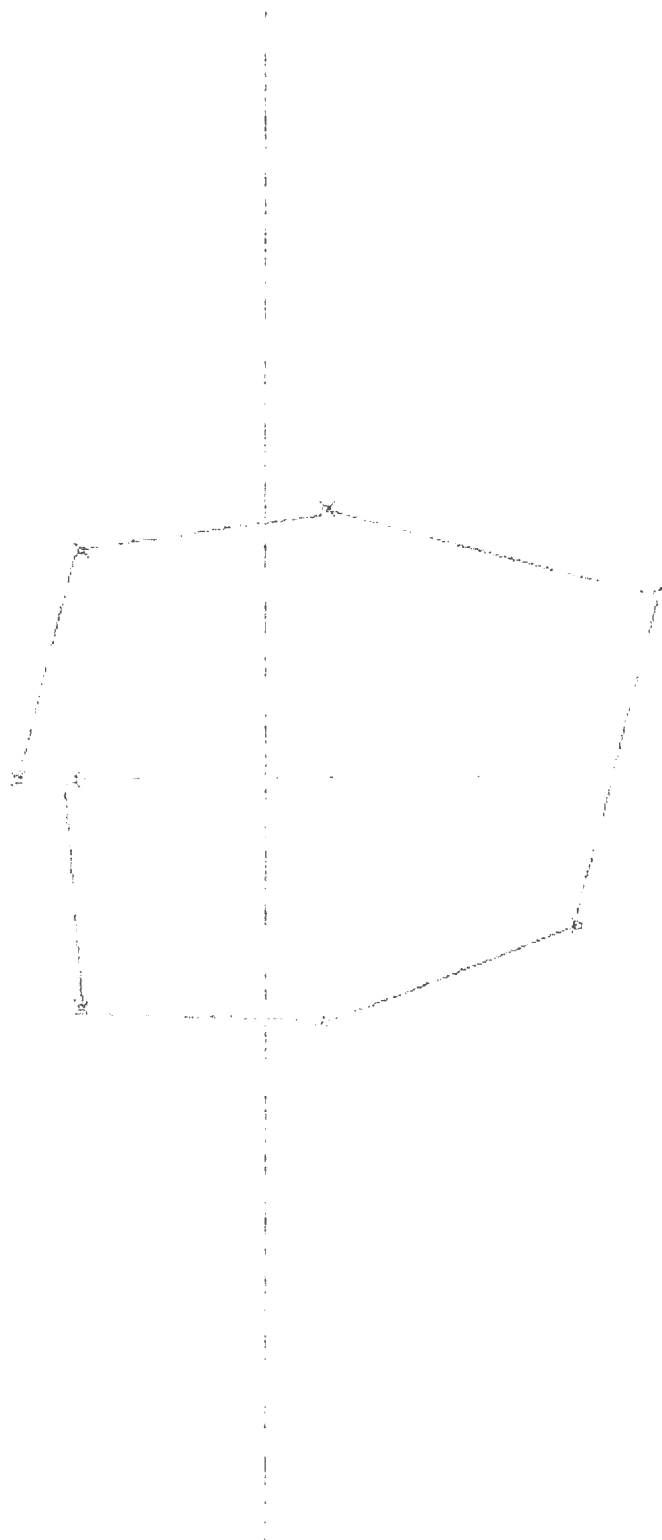
16 and 32 MHz each suffer a 13-point data dropout during the turn in these plots. This has not been corrected for in any way.



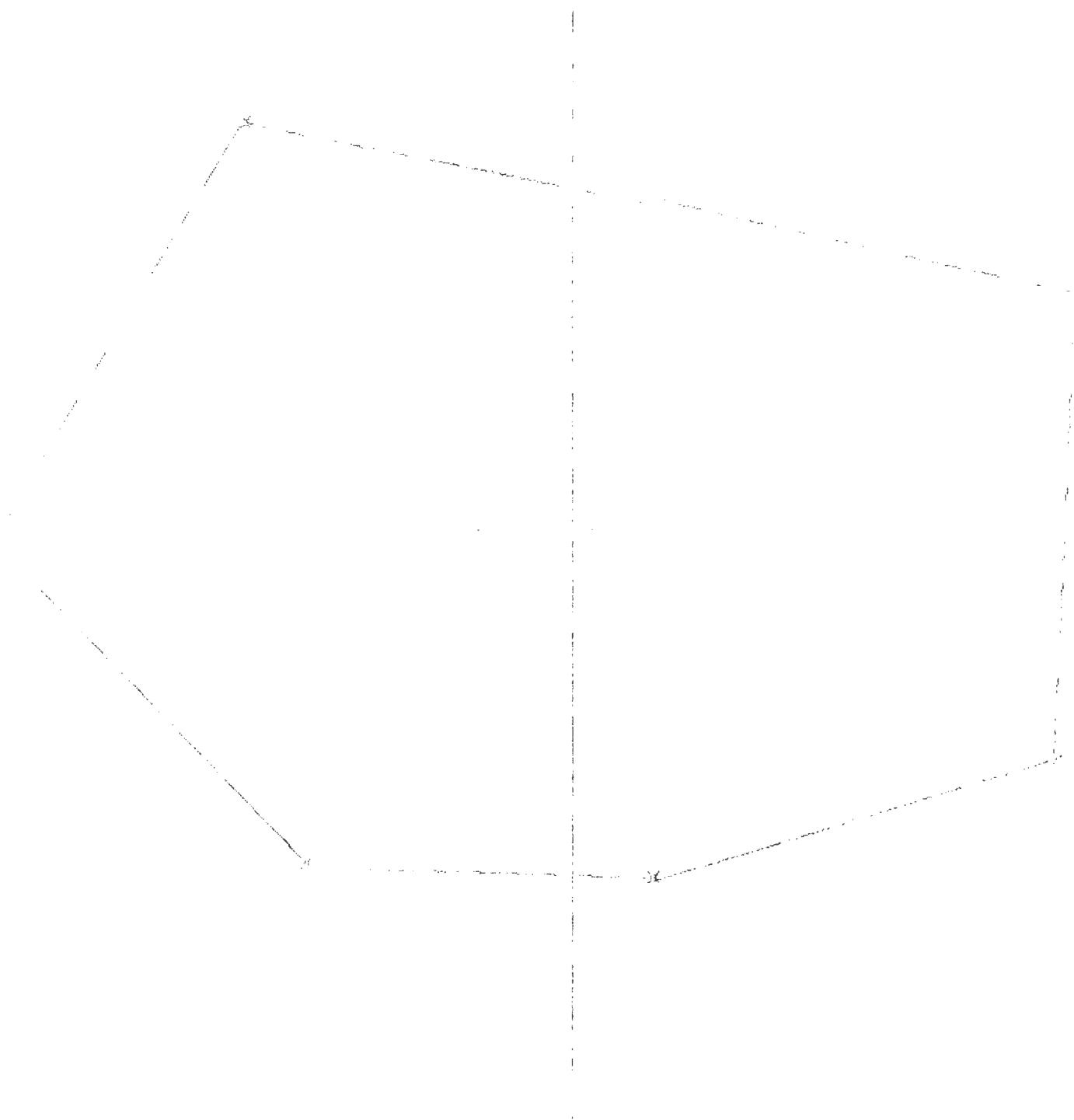
1.0 MB7. P17ND



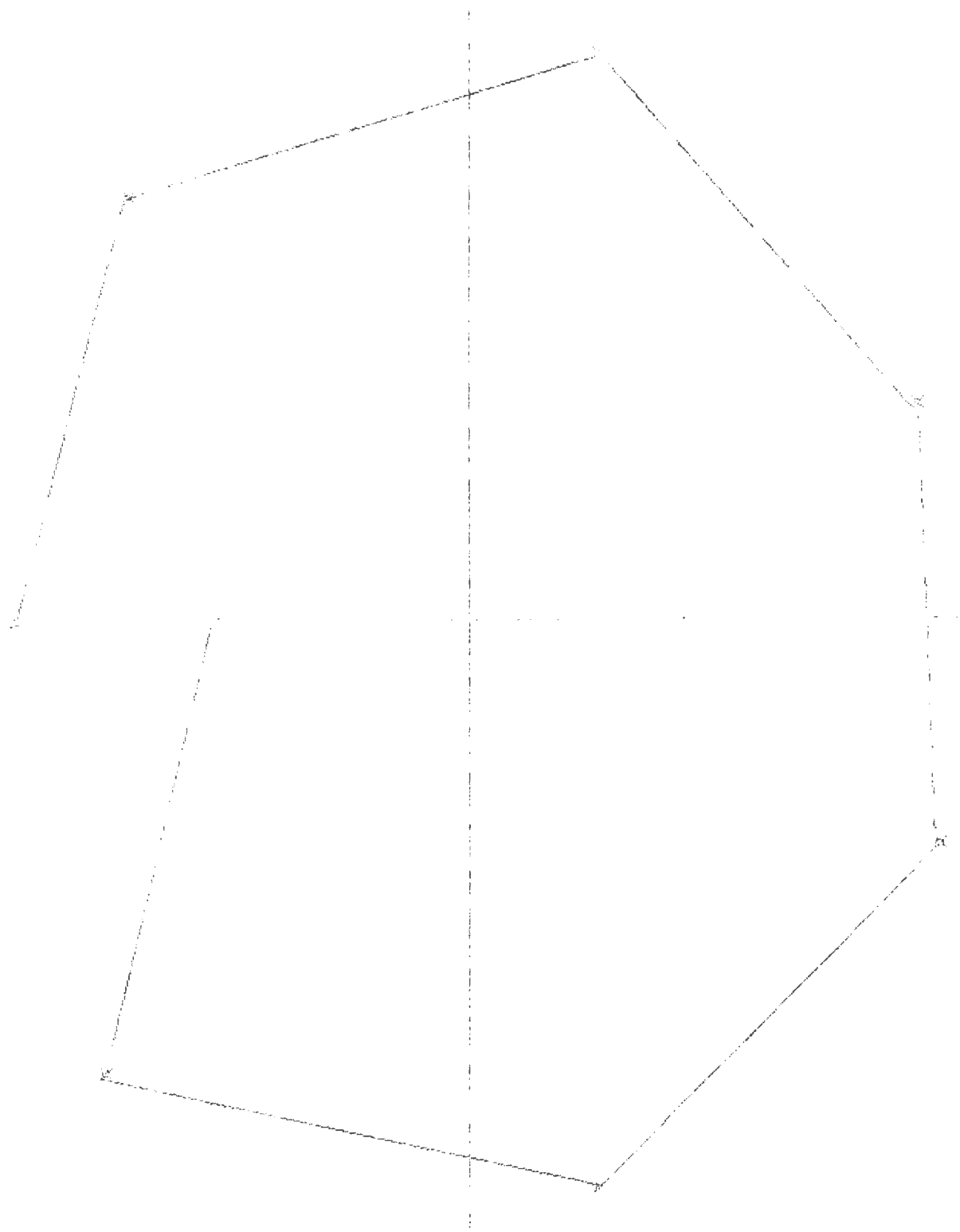
1.0 MHZ. HoENC



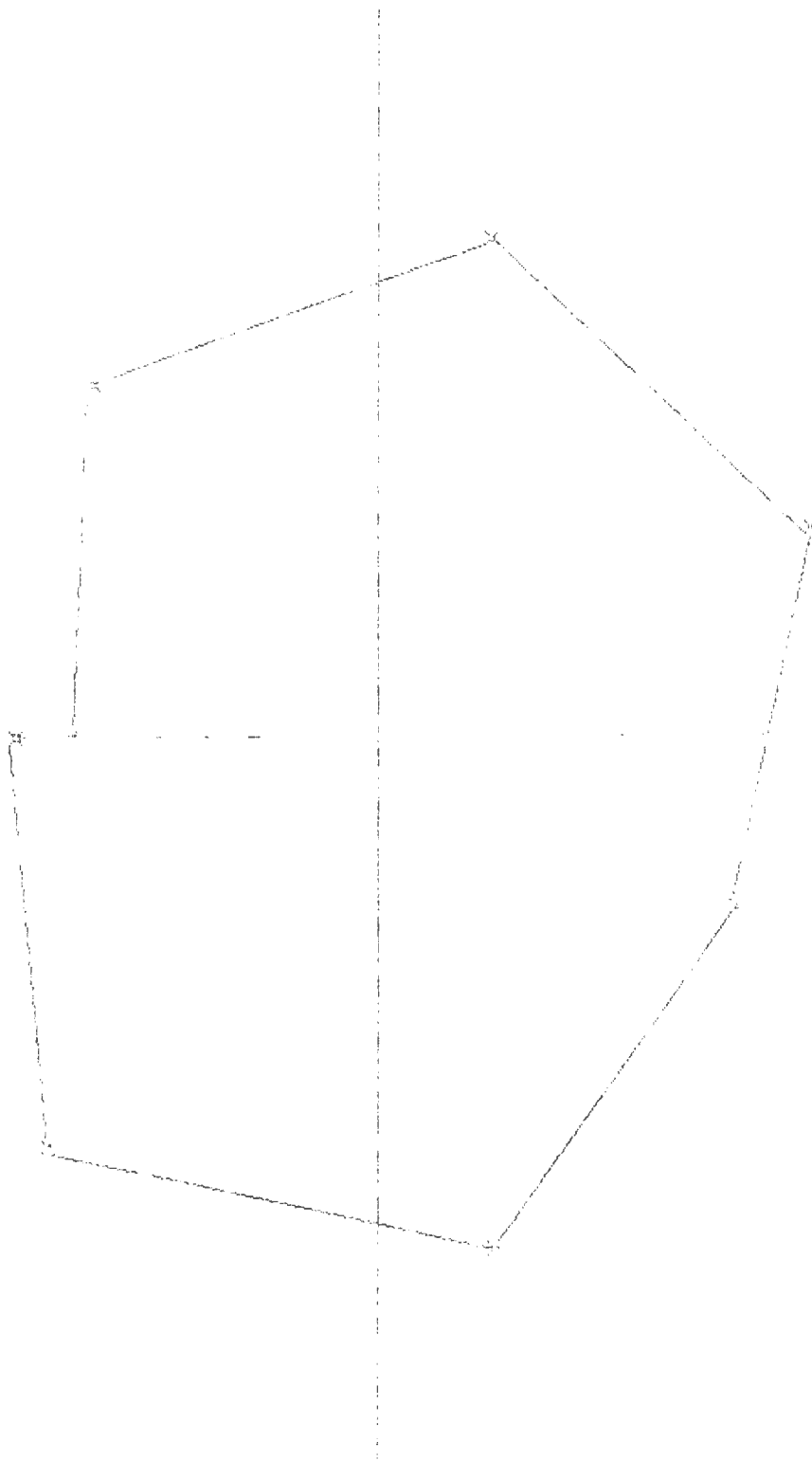
1.0 MHz. 4ZENC



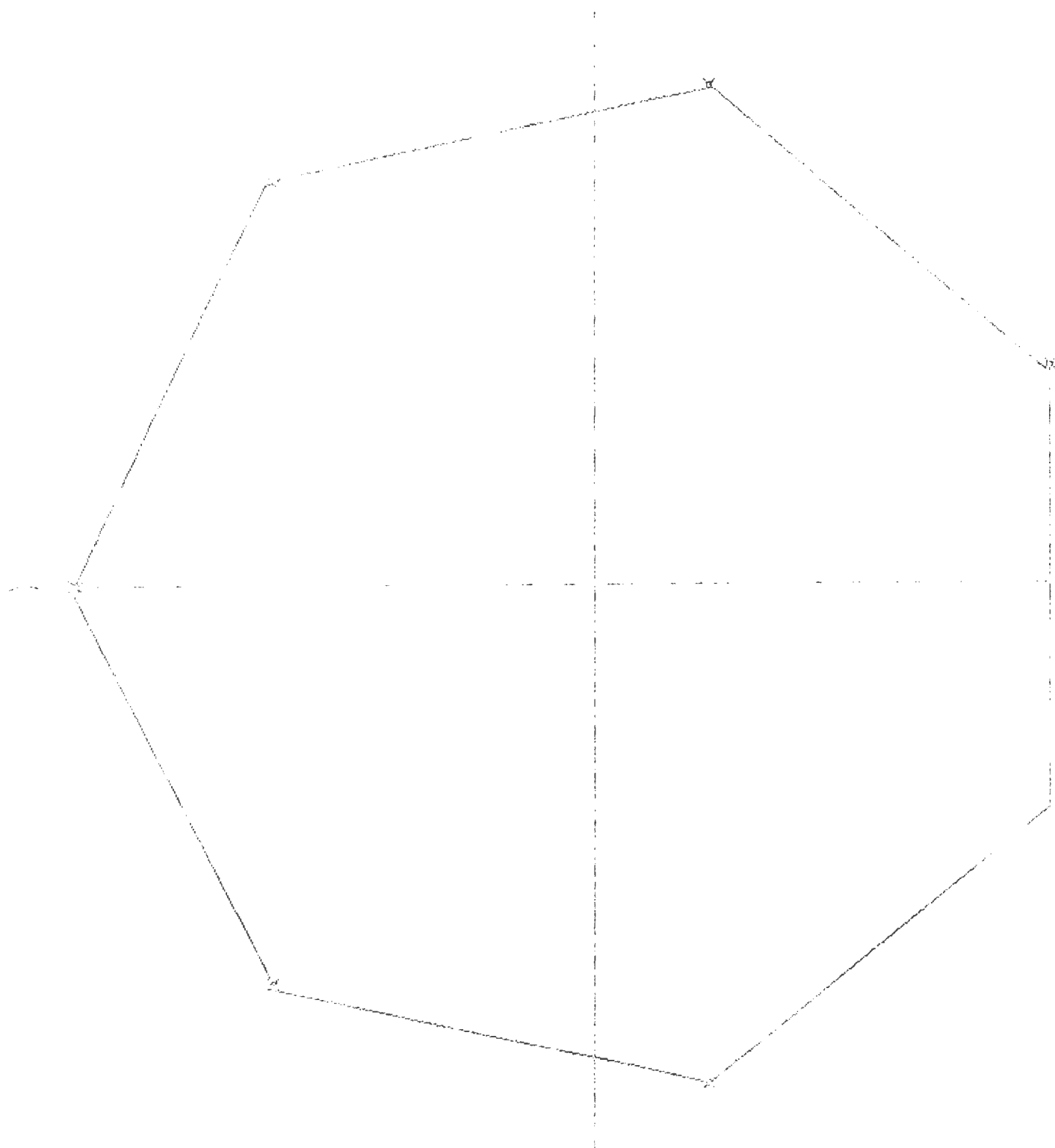
1.0 MHZ. HP3RD



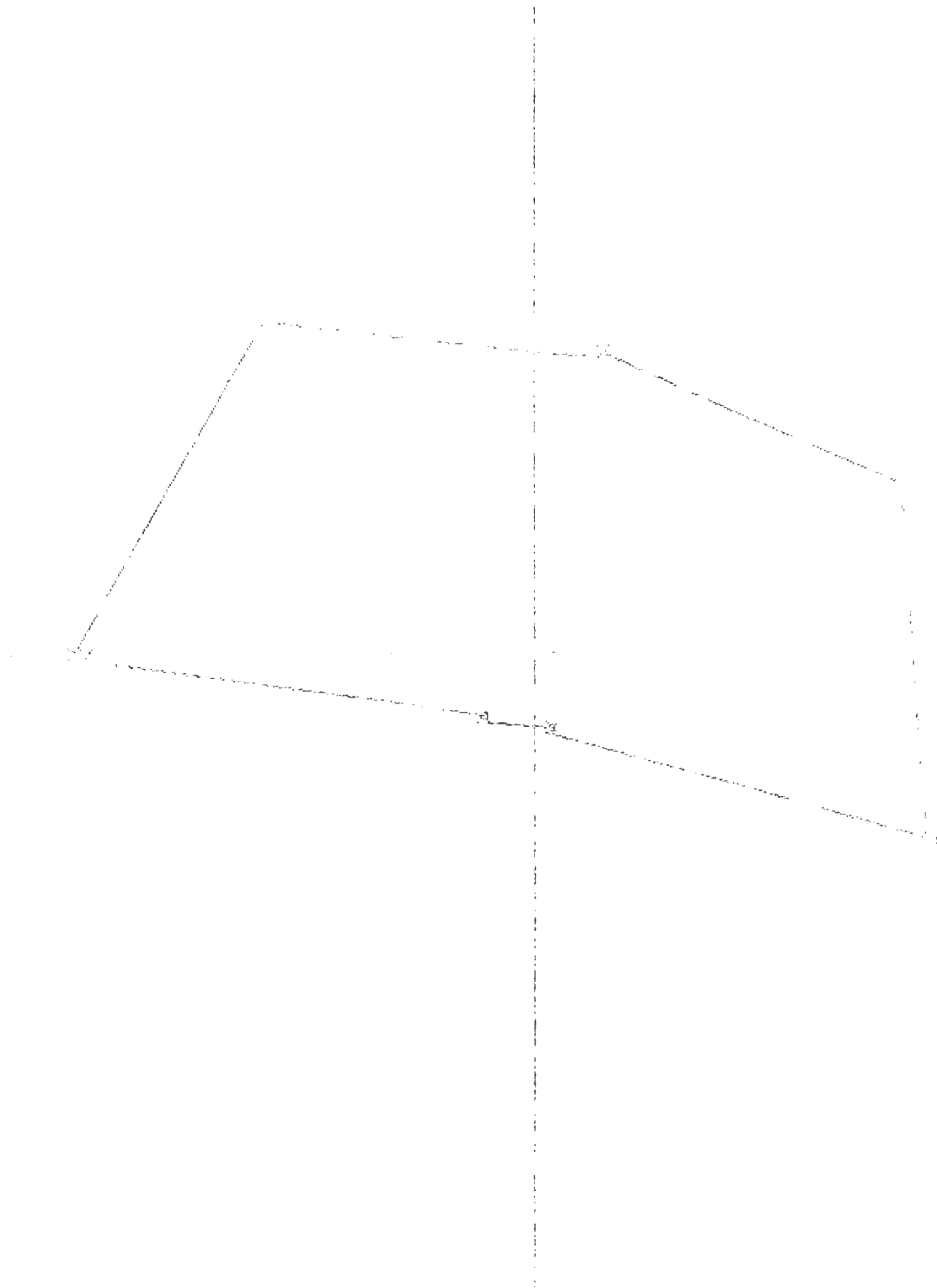
1.0 MHz. H₂O BRD



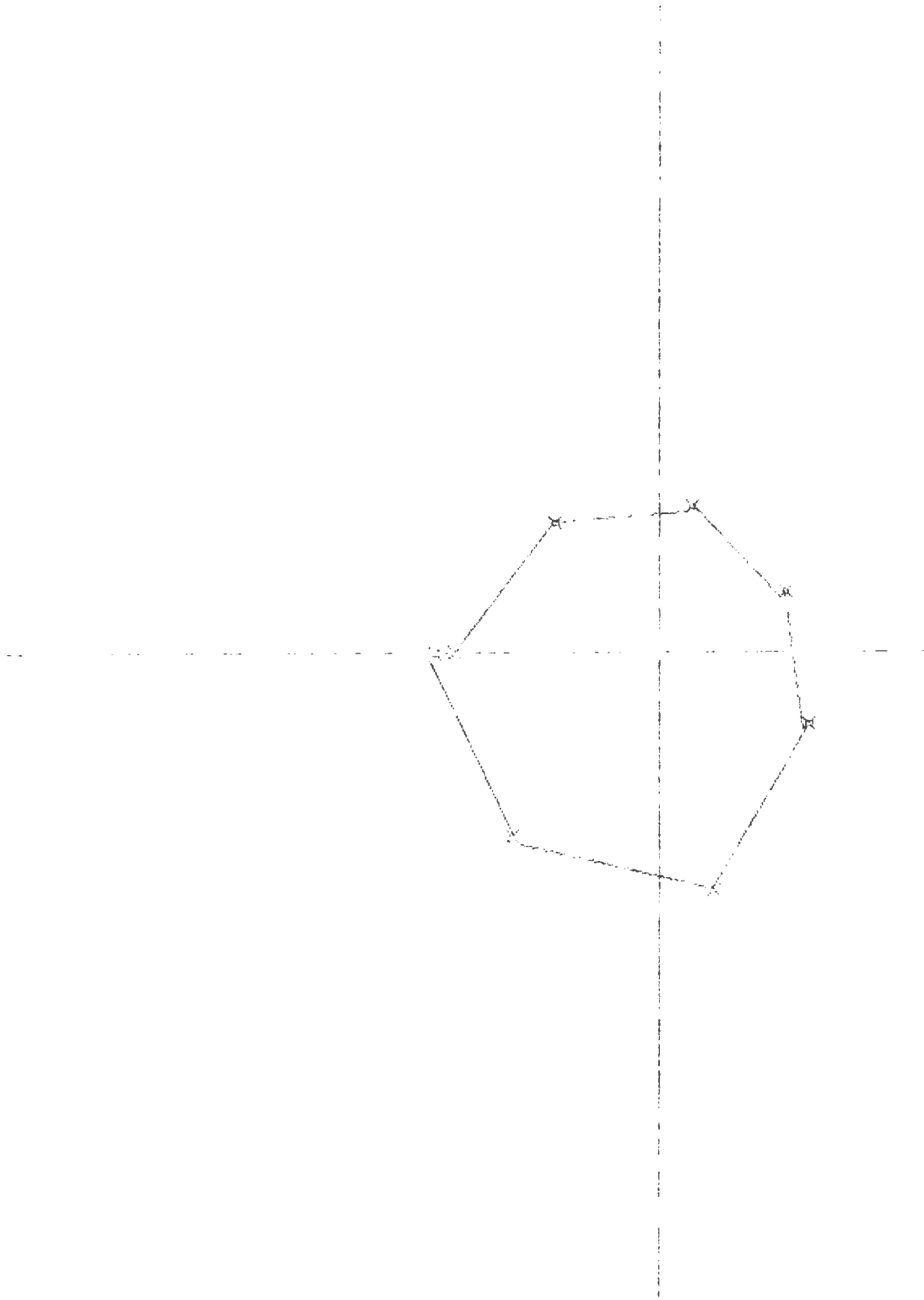
2.1 MHZ. H₂END



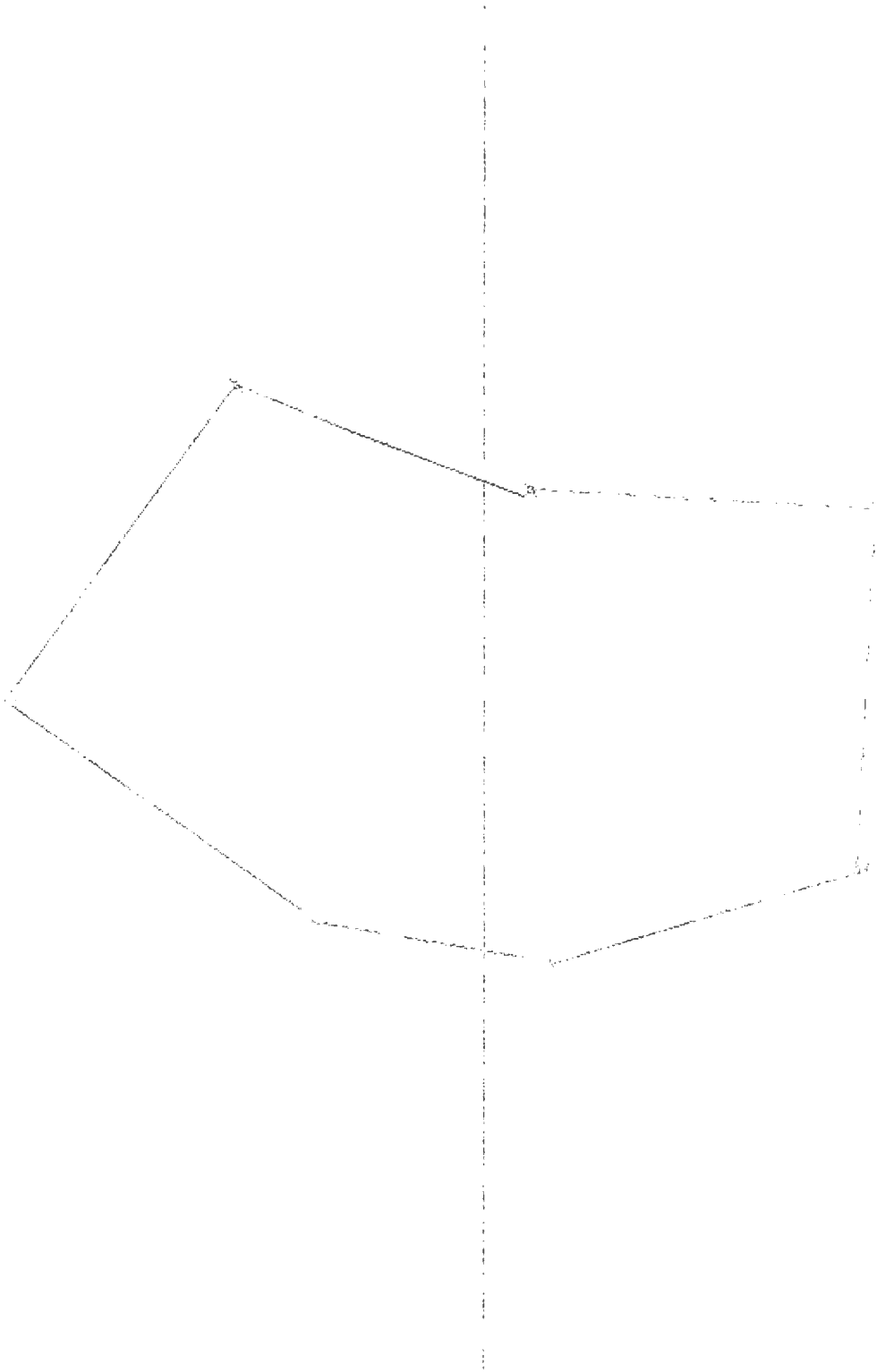
1.0 MHZ. HZBRD



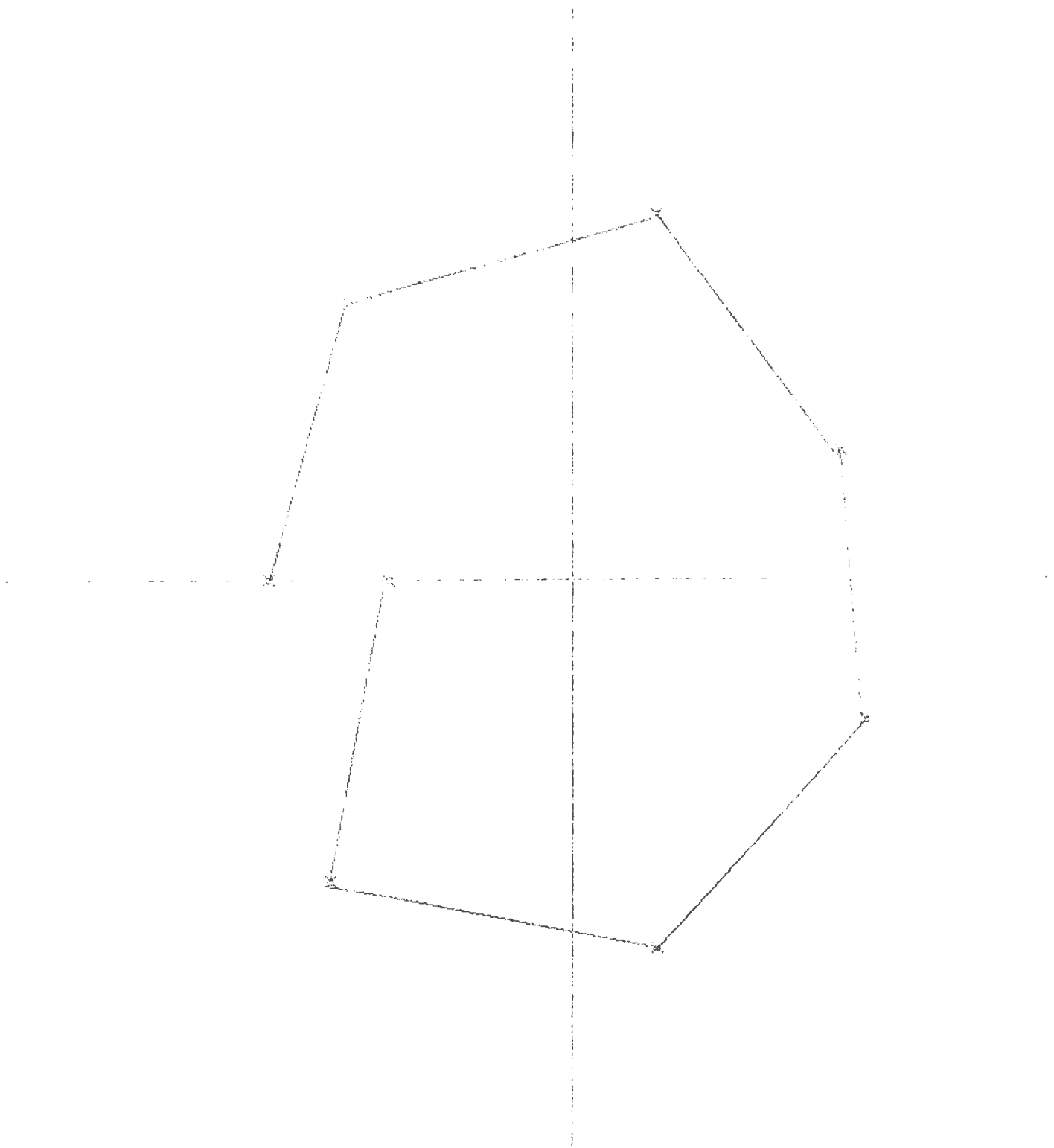
2.1 MHZ. HOENP



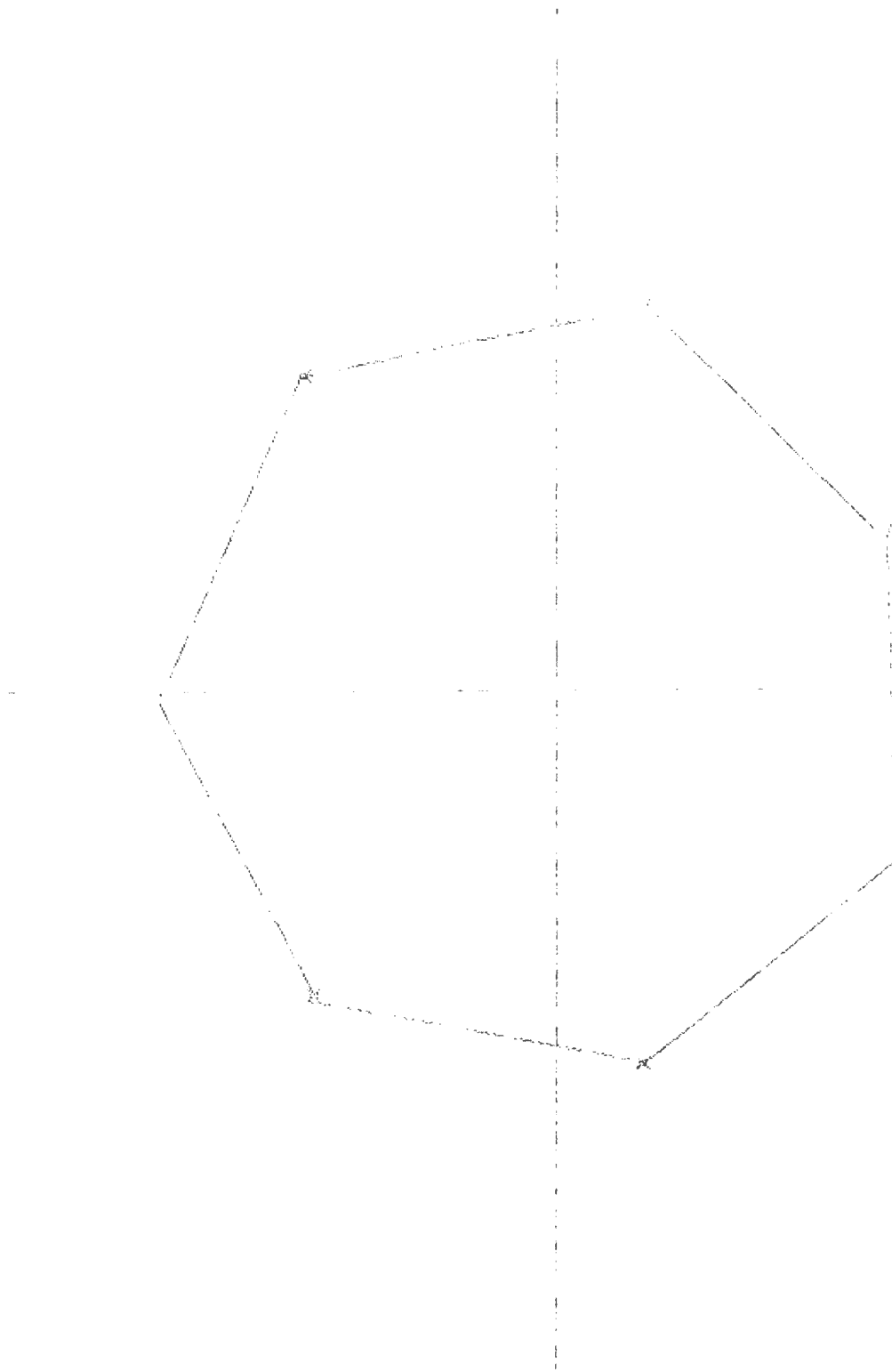
2.1 MHZ. HZENS



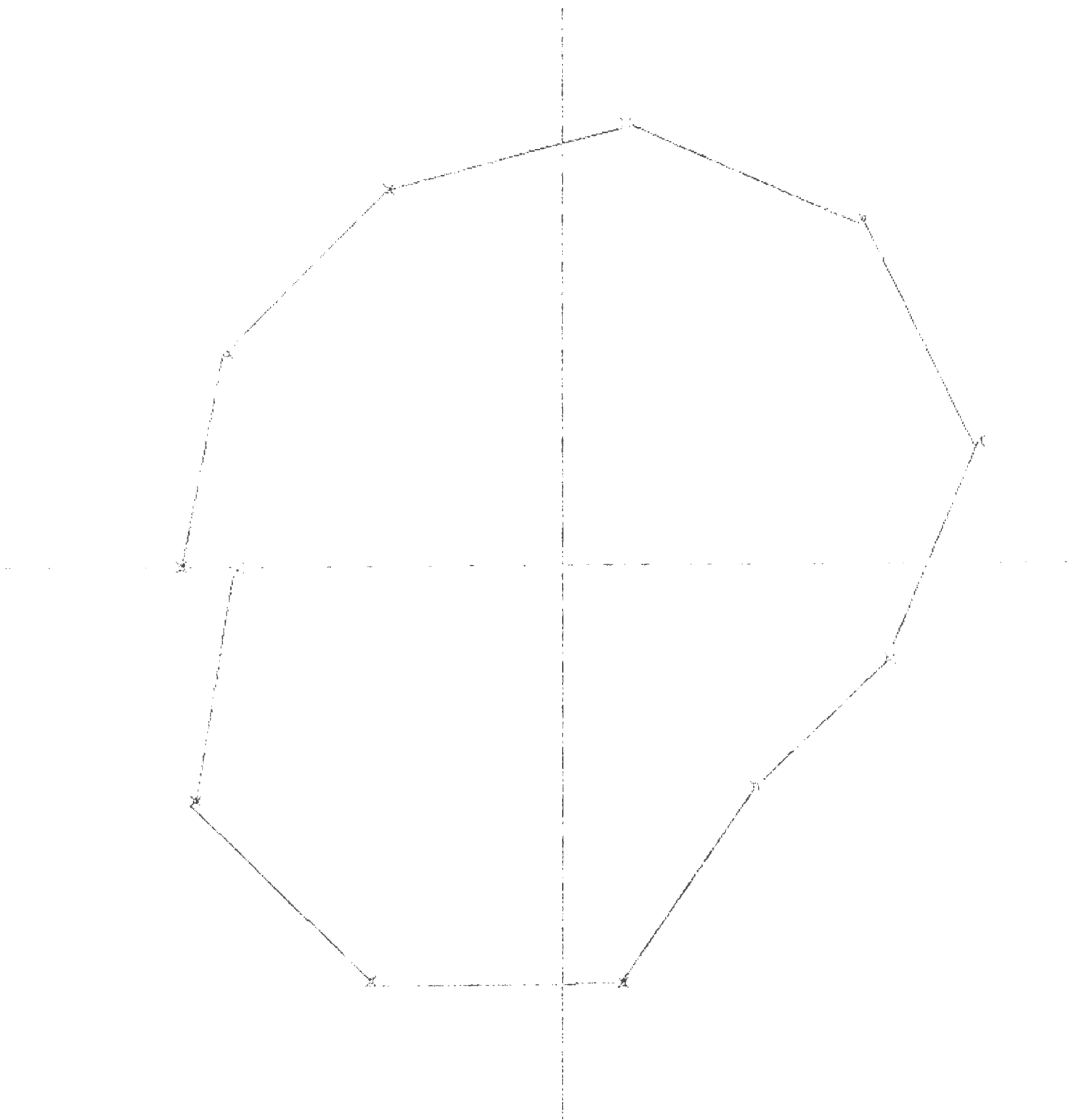
2.1 MHZ. FPDF



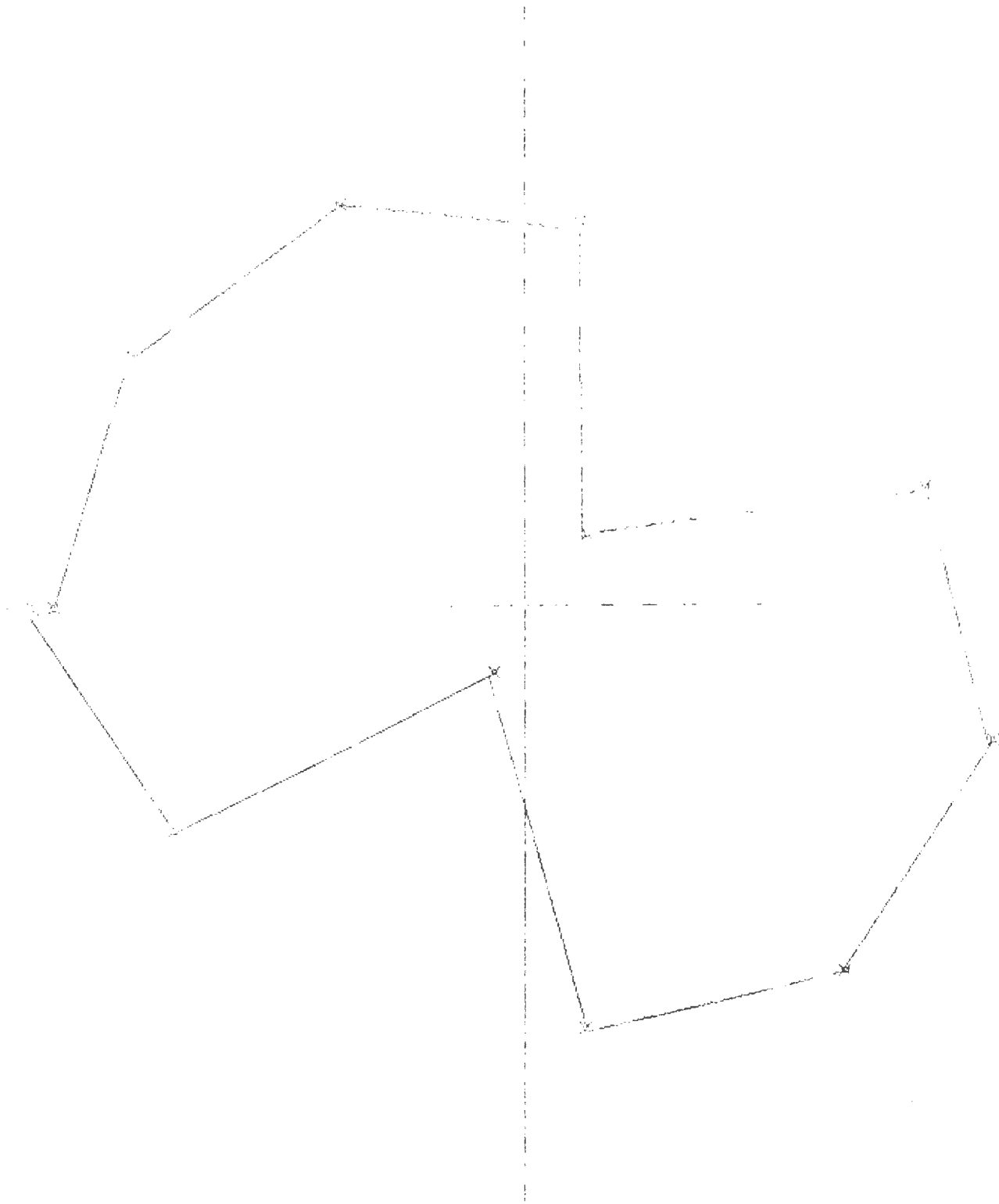
2.1 MHZ. HoBRD



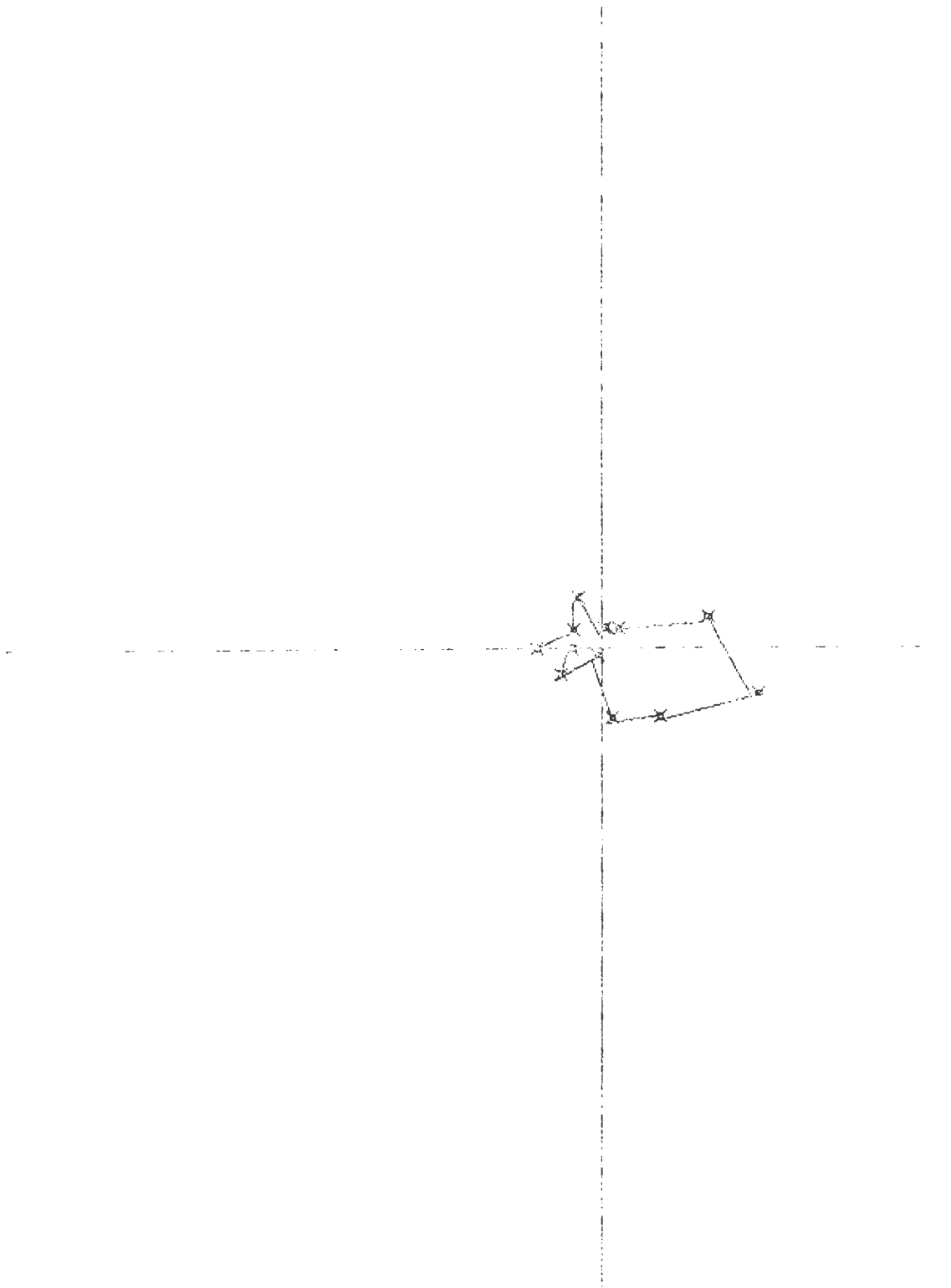
2.1 MHz. HZBRD



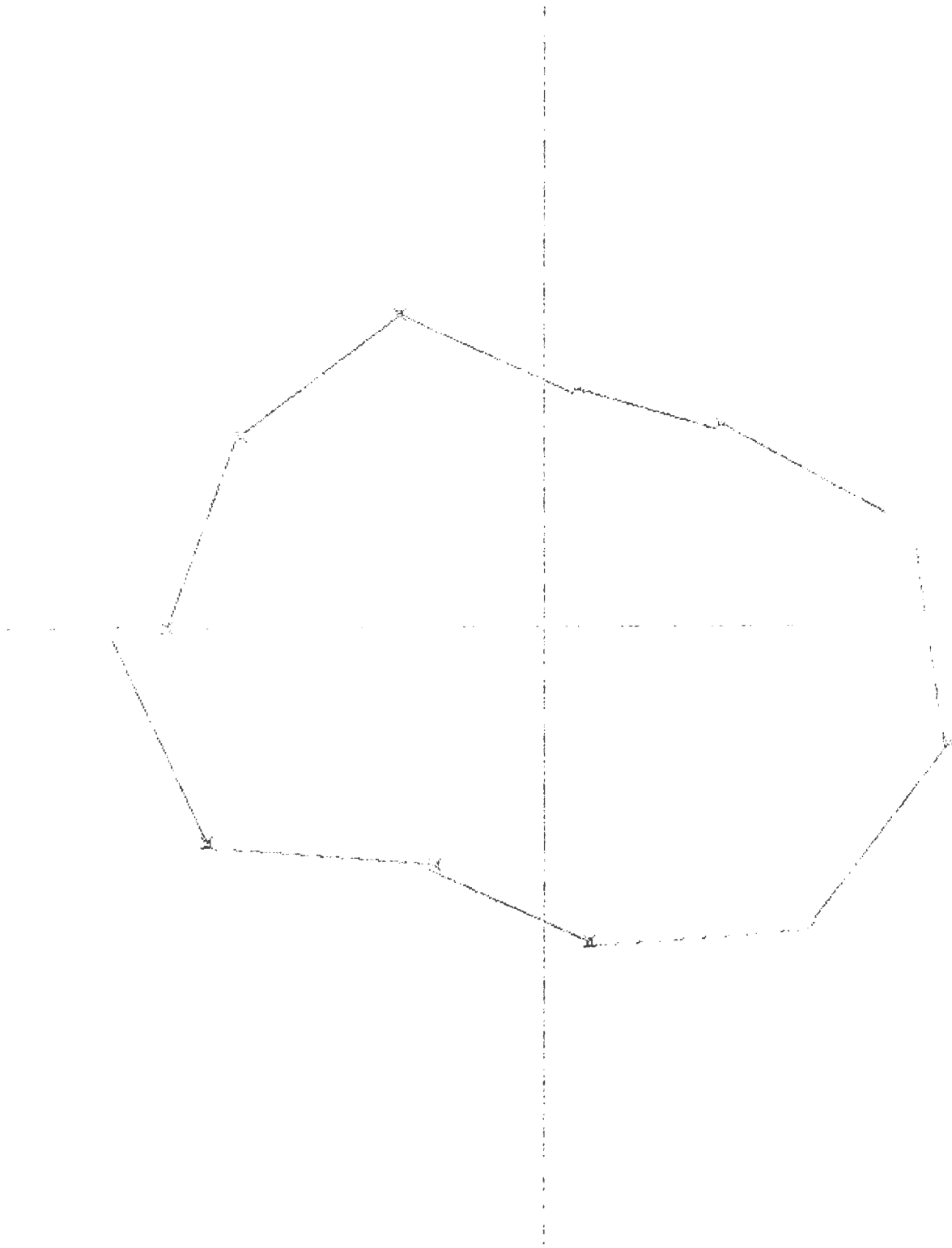
4.0 MHZ. H~~R~~END



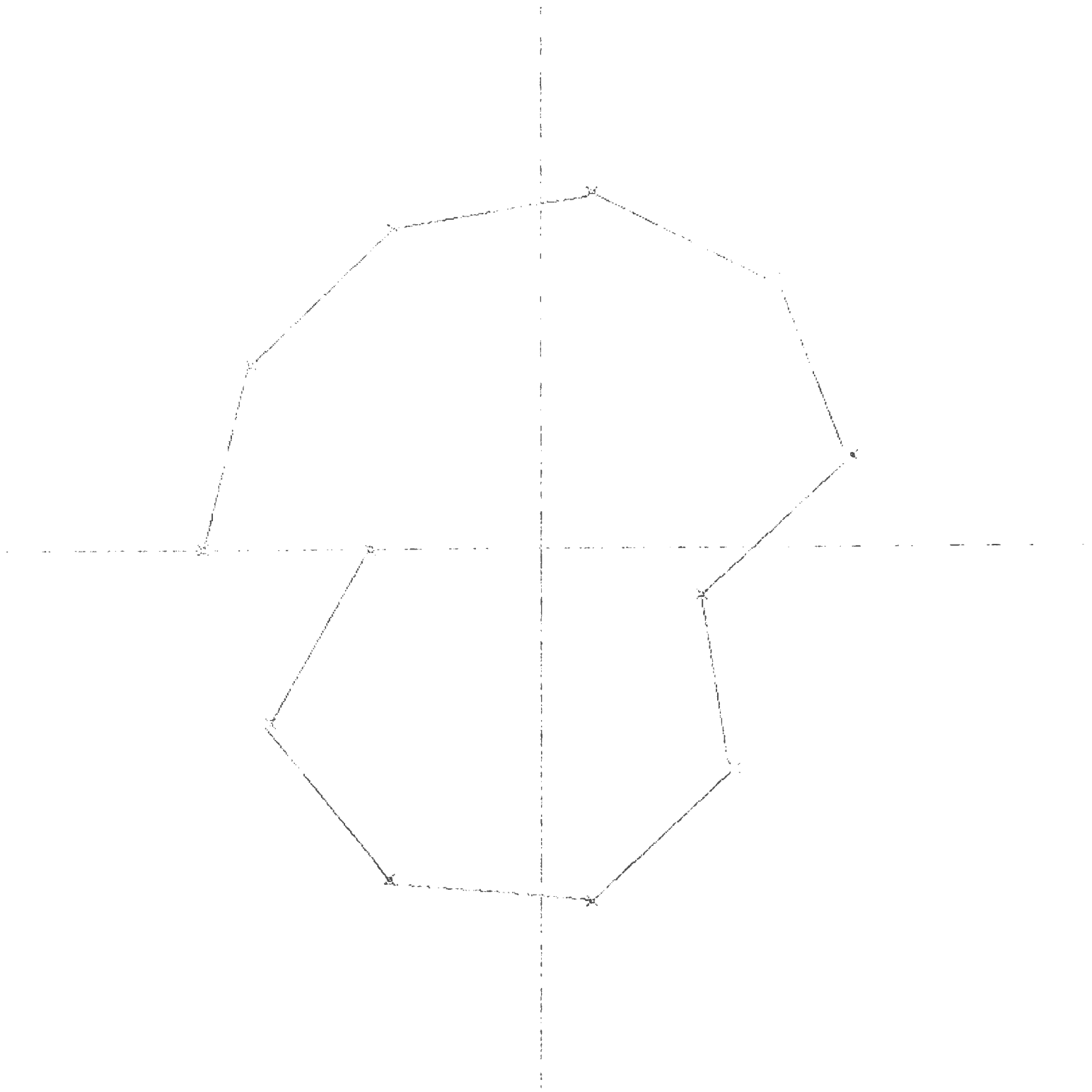
U.C. MHz. F0END



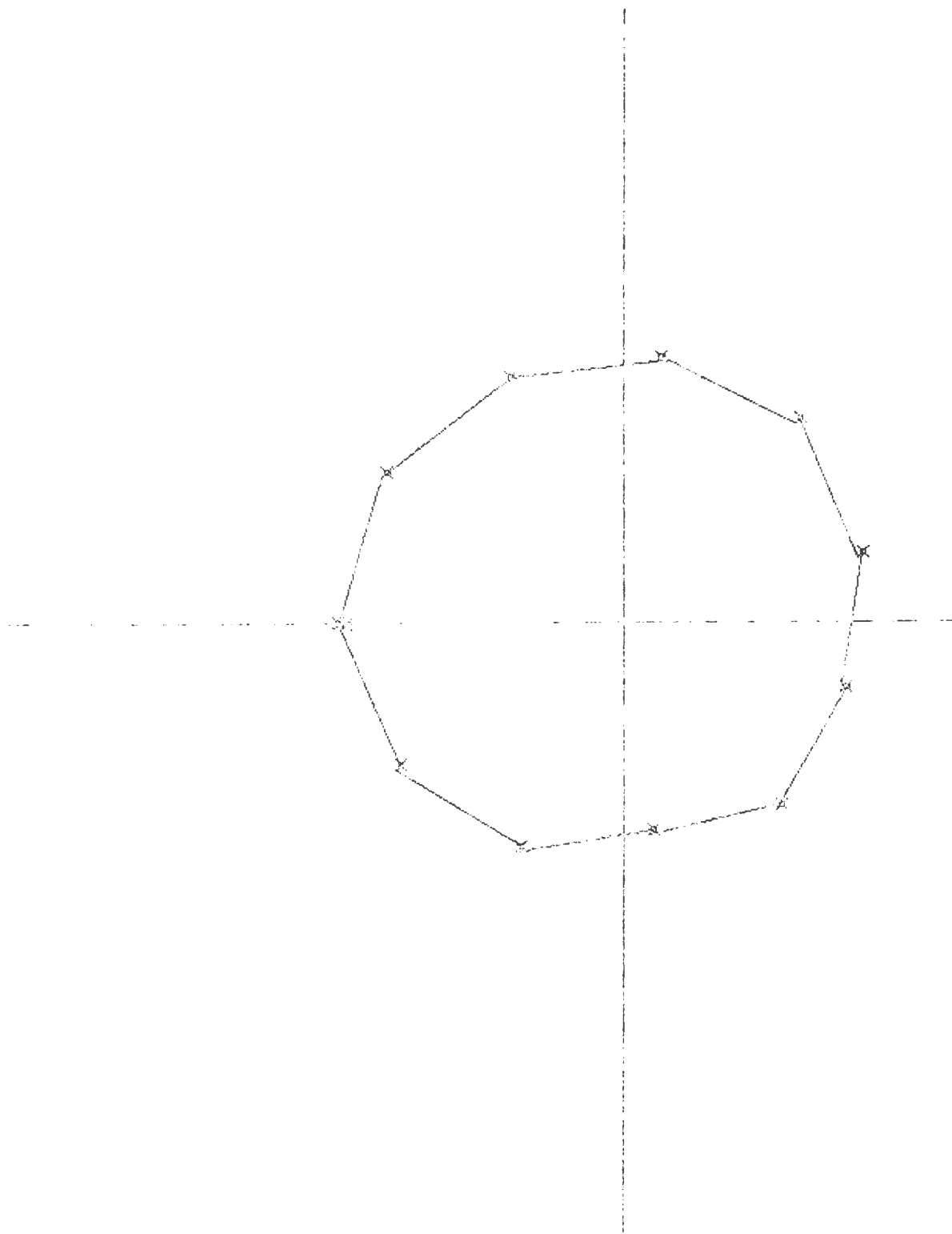
4.0 MHZ. HZEND



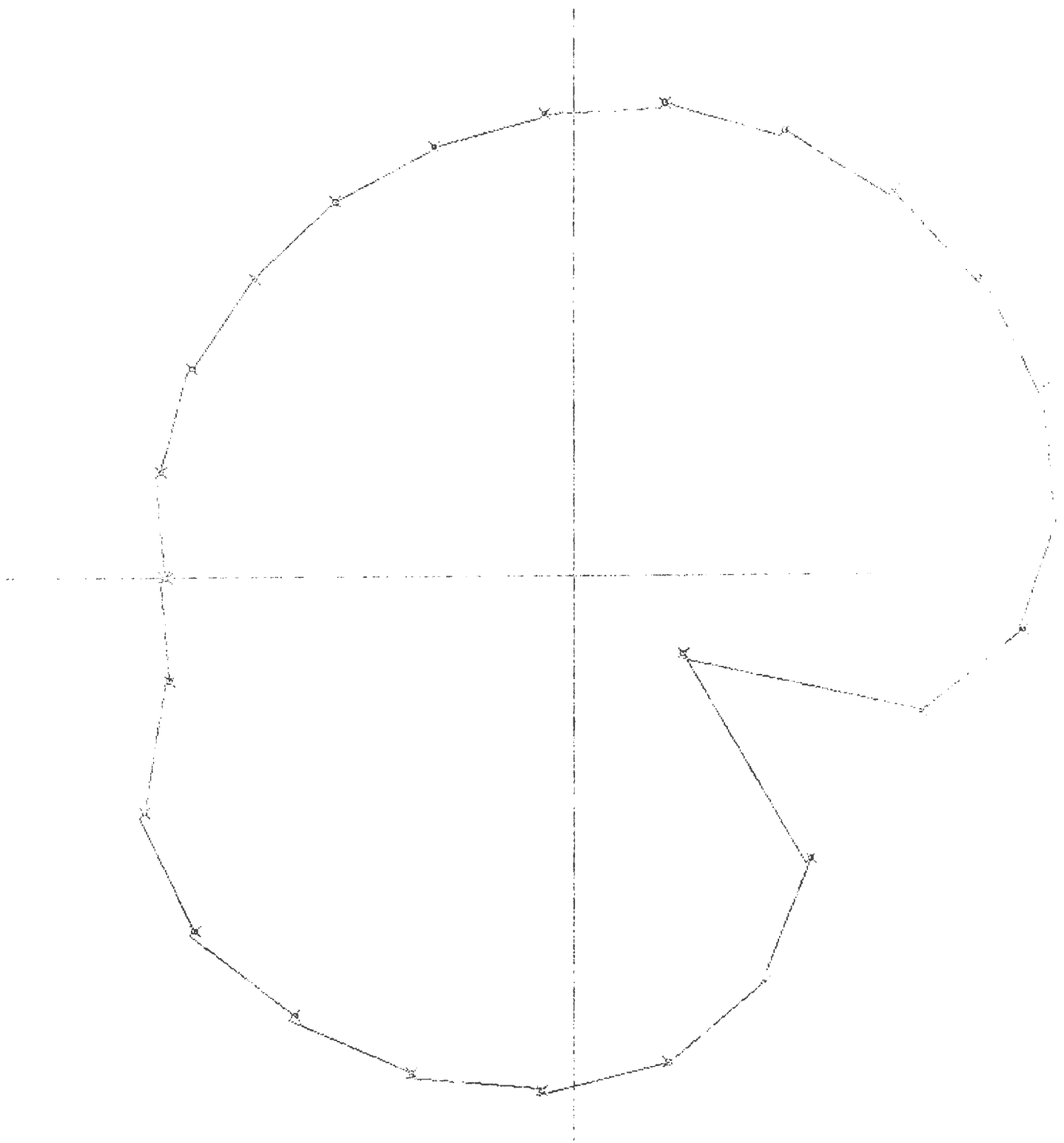
4.0 MHZ. H₂BR



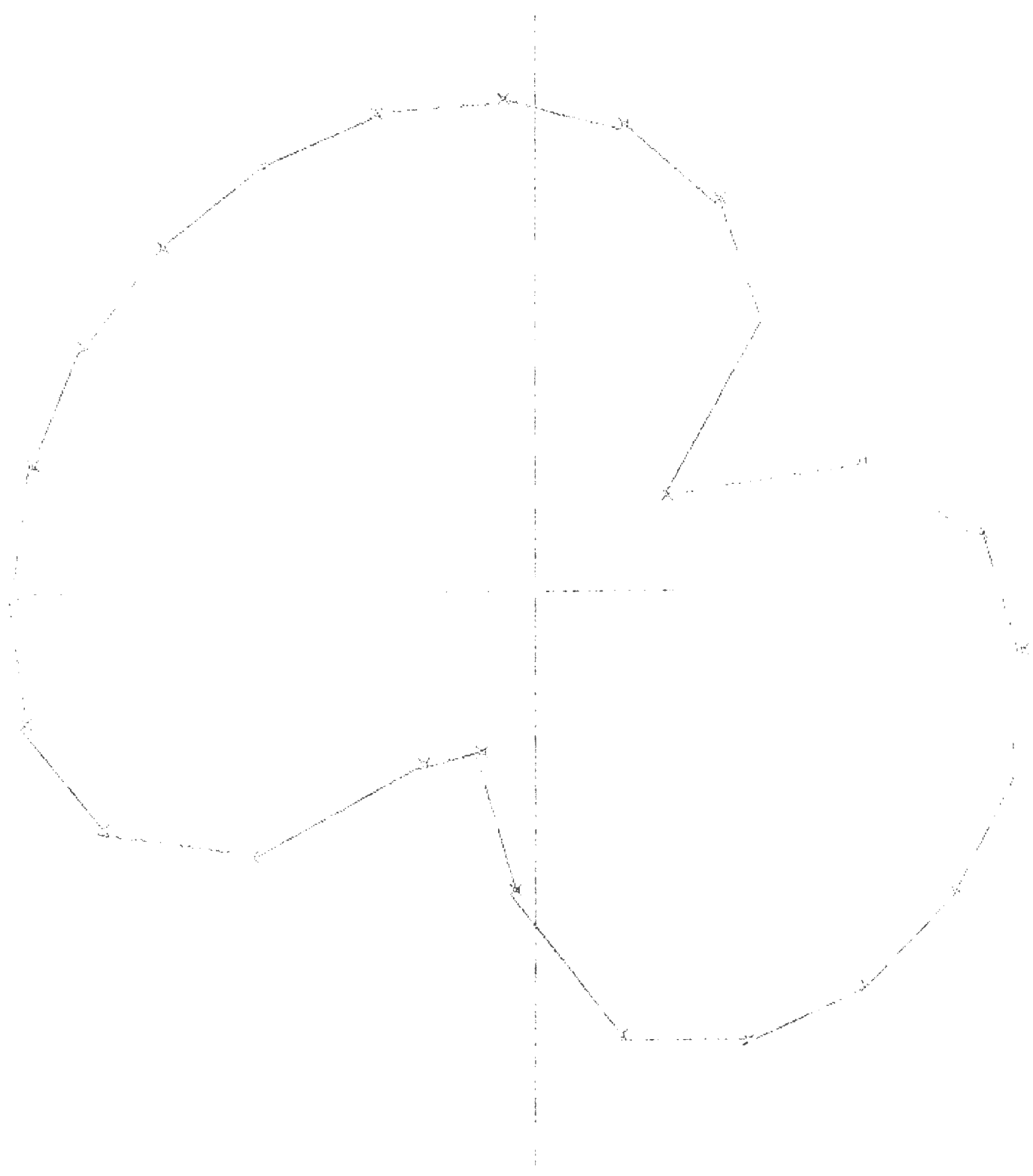
4.0 MFZ. HφBRC



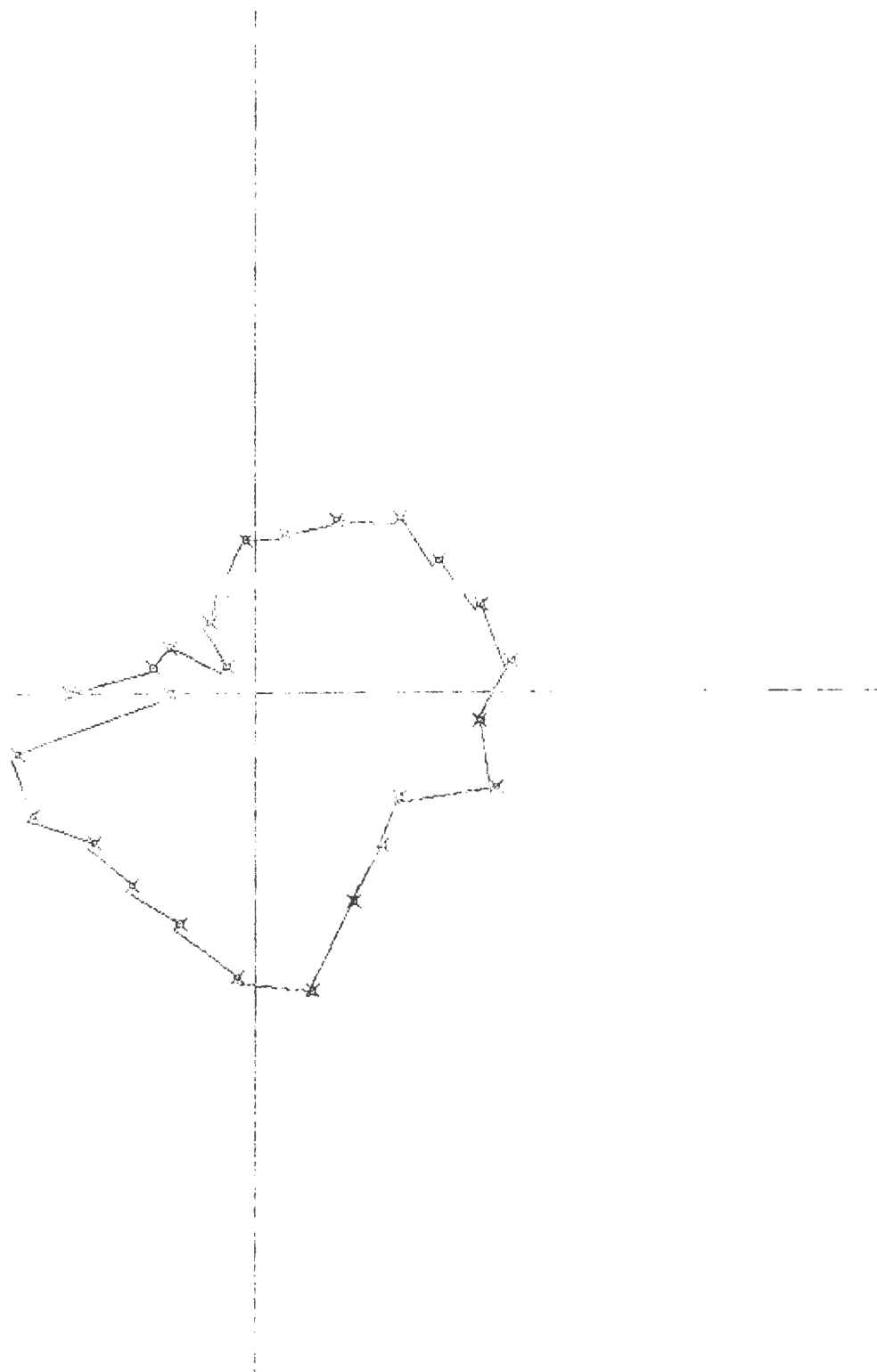
4.0 MHZ. HZBRD



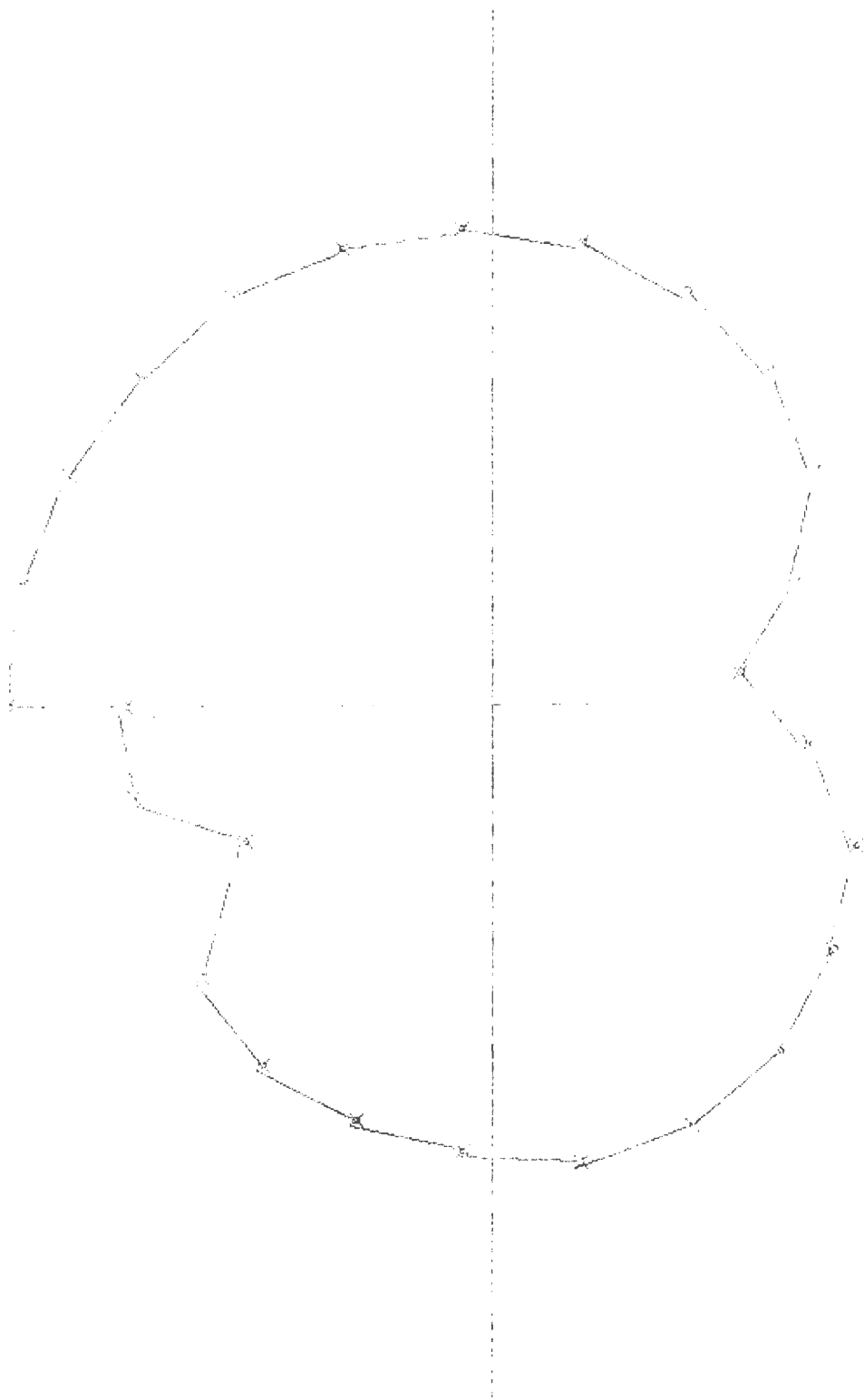
8.1 MHZ. H?END



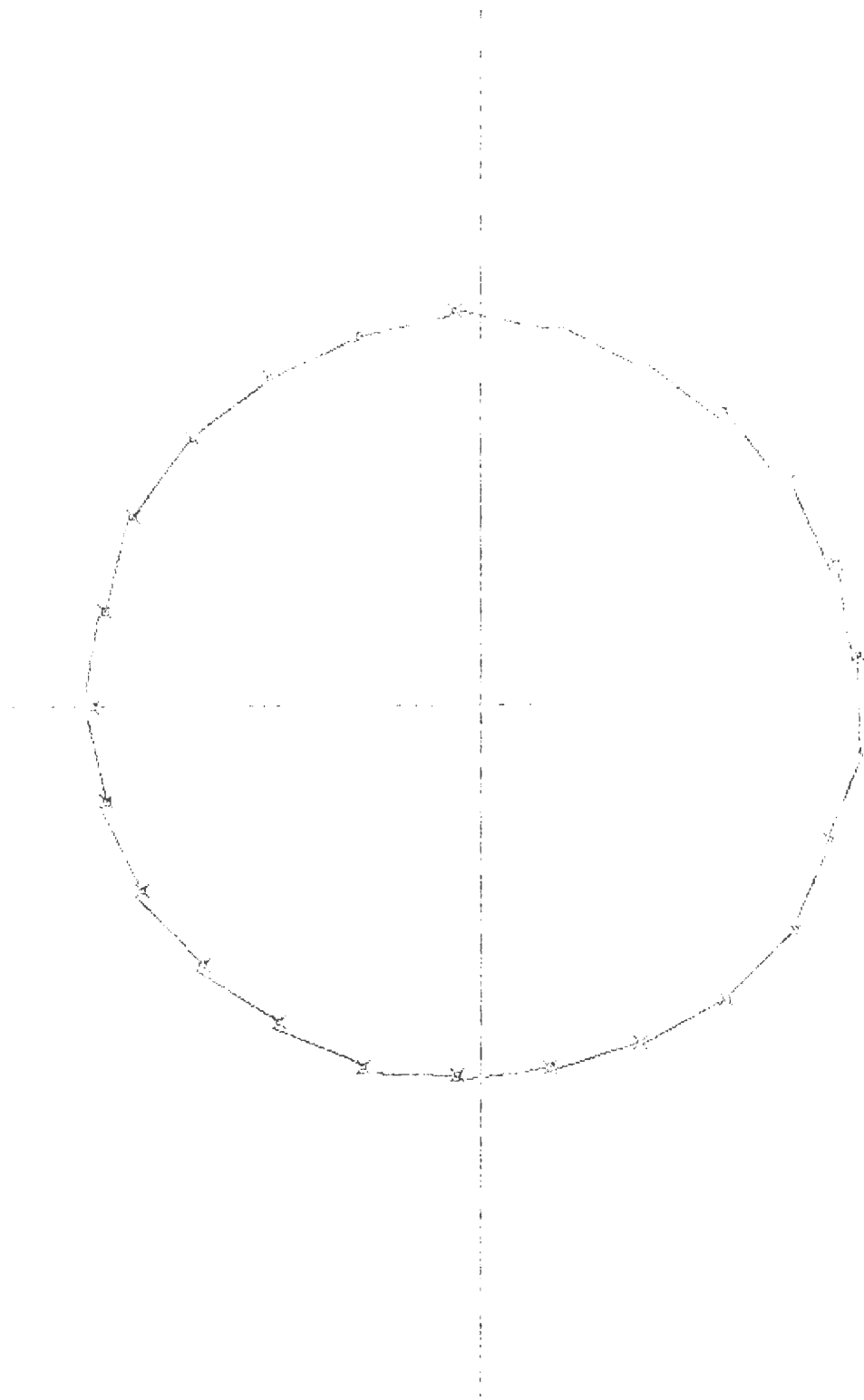
8.1 MHZ HoENC



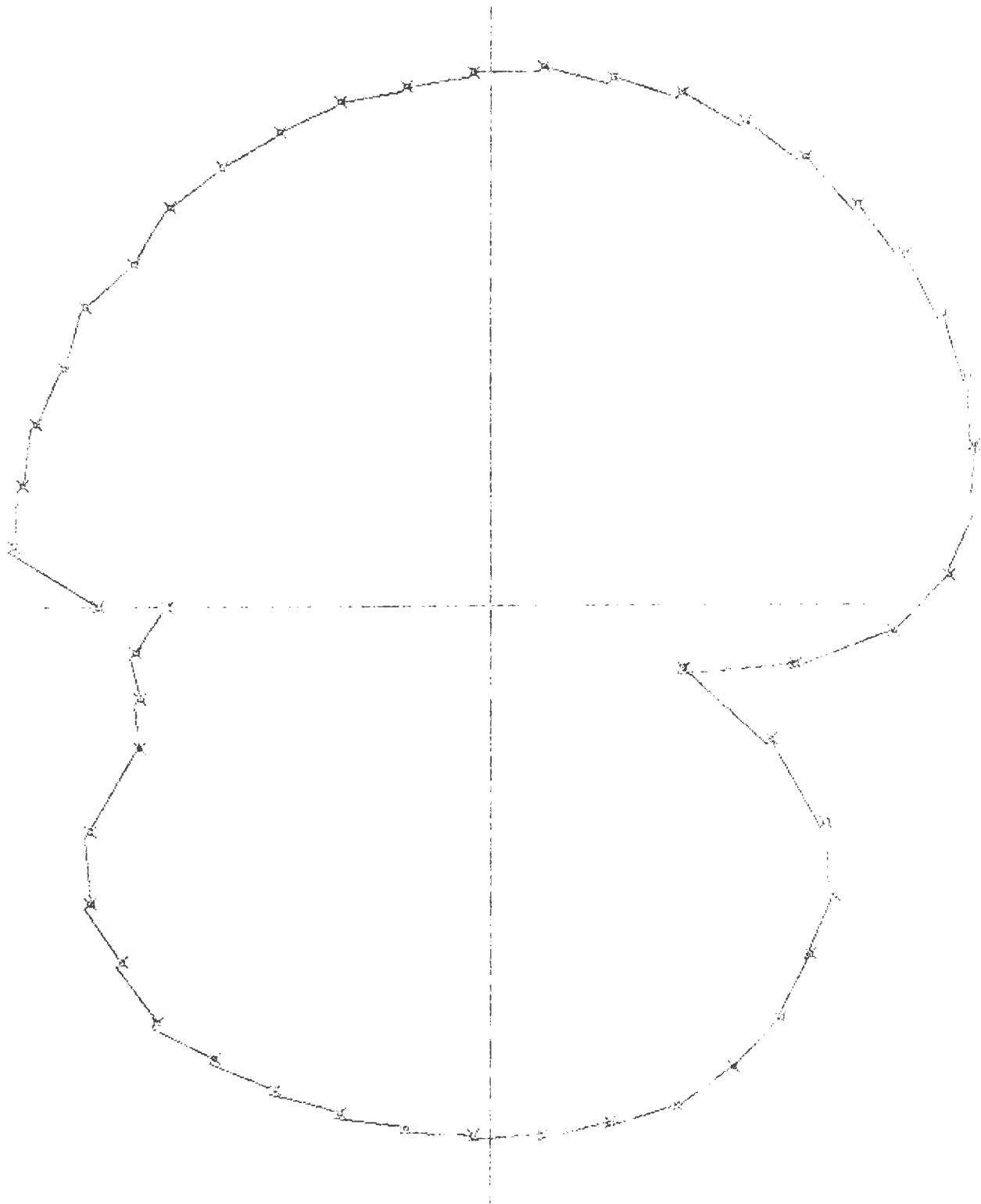
8.1 MHZ. HZENS



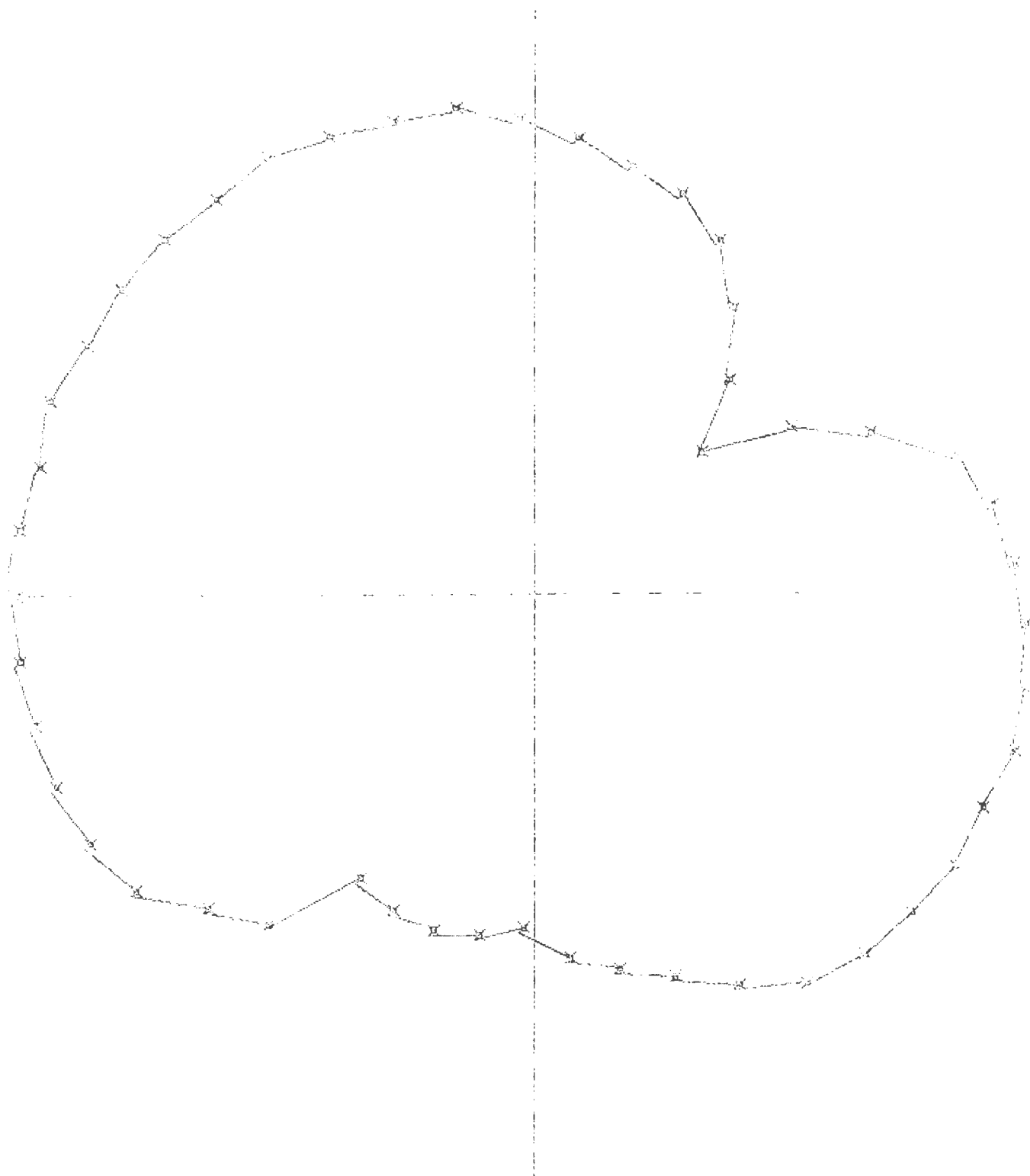
8.1 MHZ. HoBPC



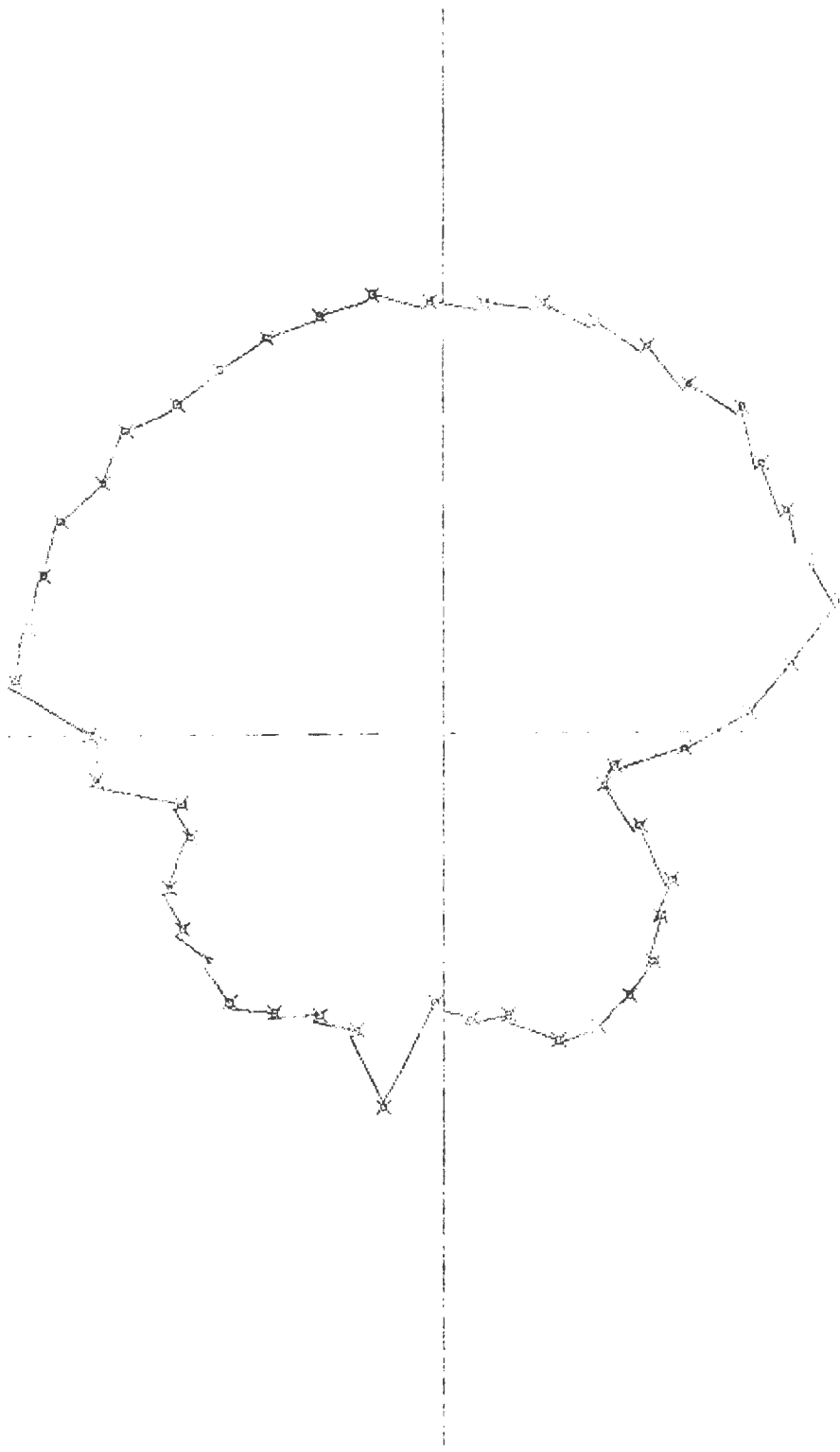
8.1 MHZ. HZBRC



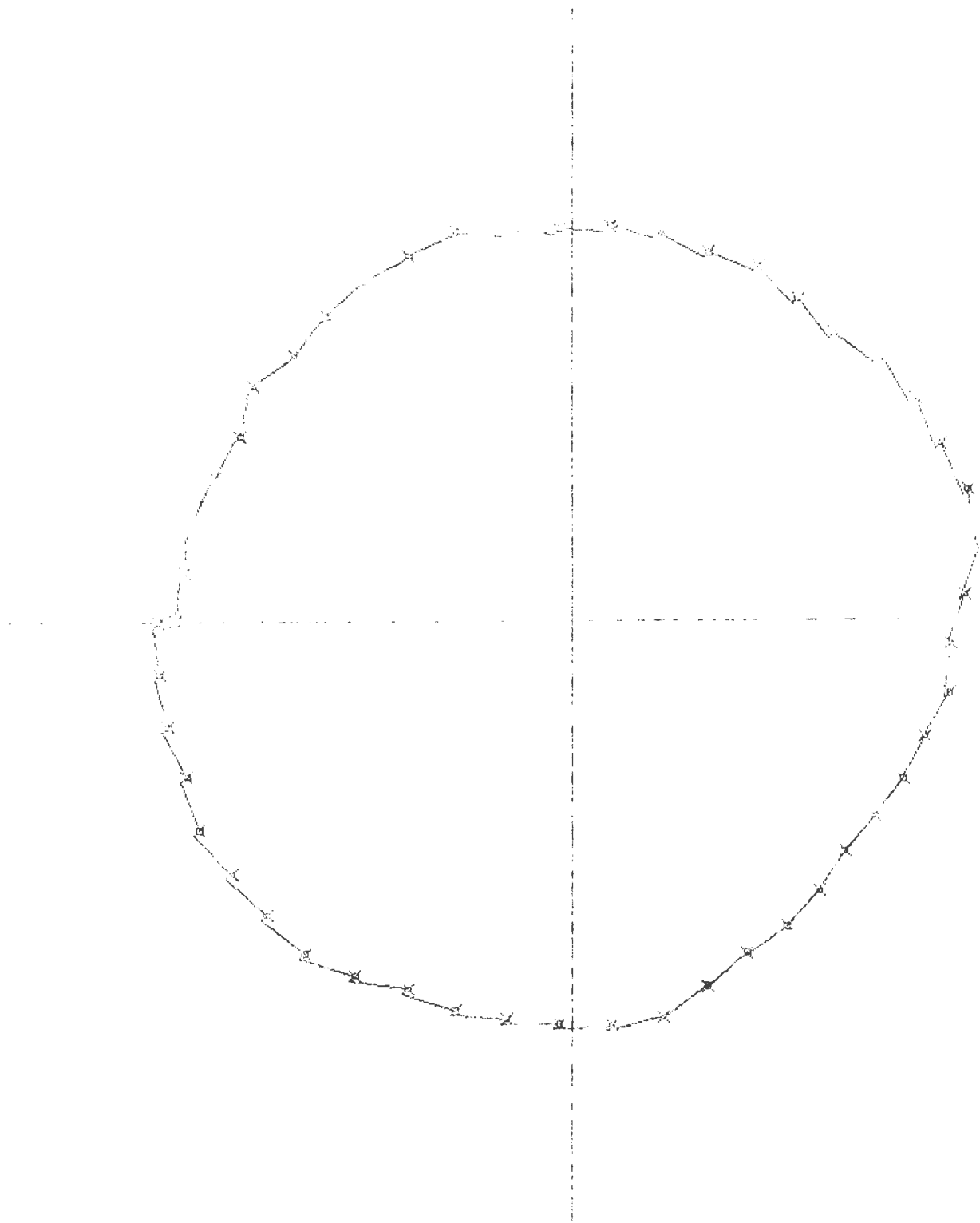
16.0 MHZ. HPENC



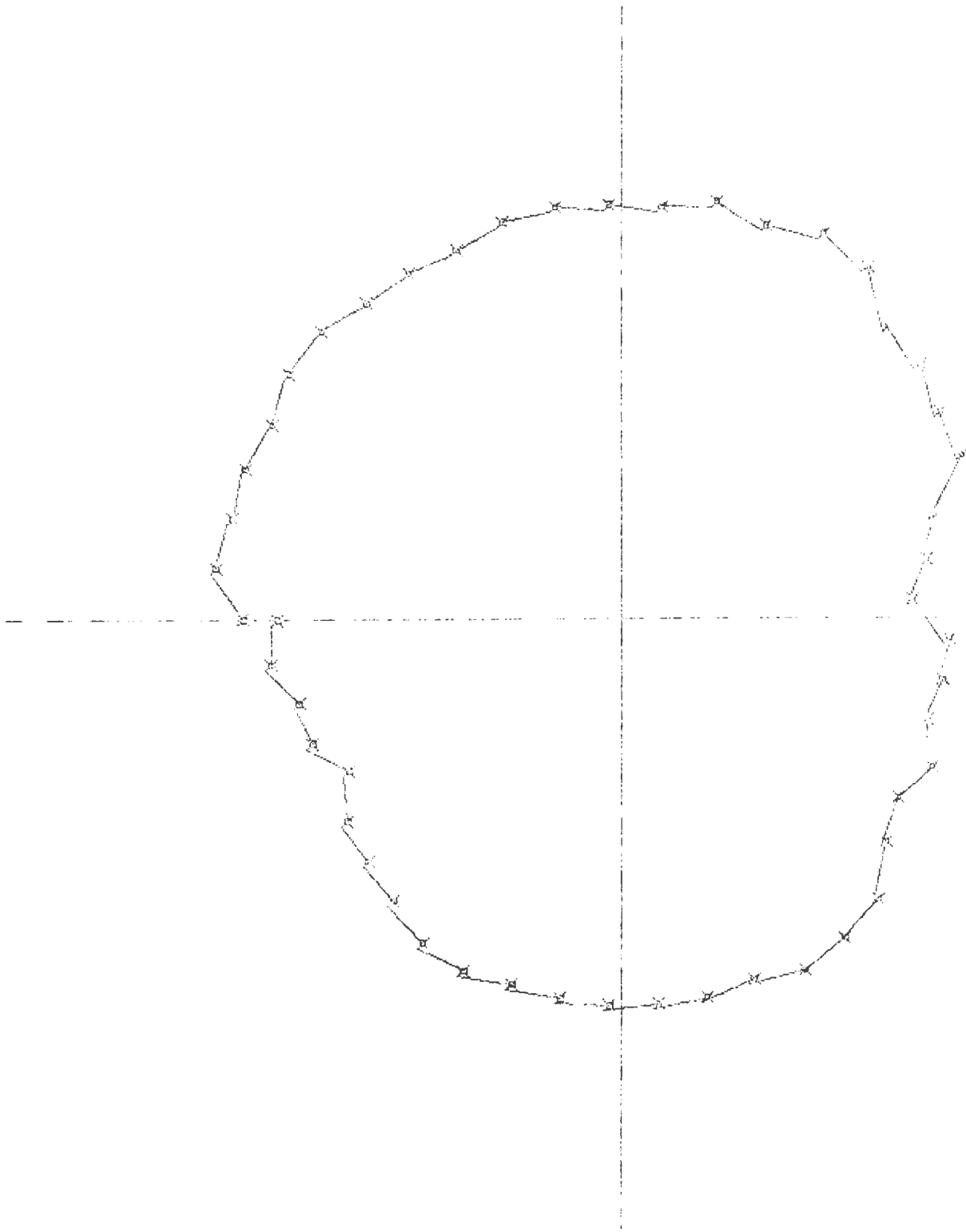
16.0 MHz. HφEND



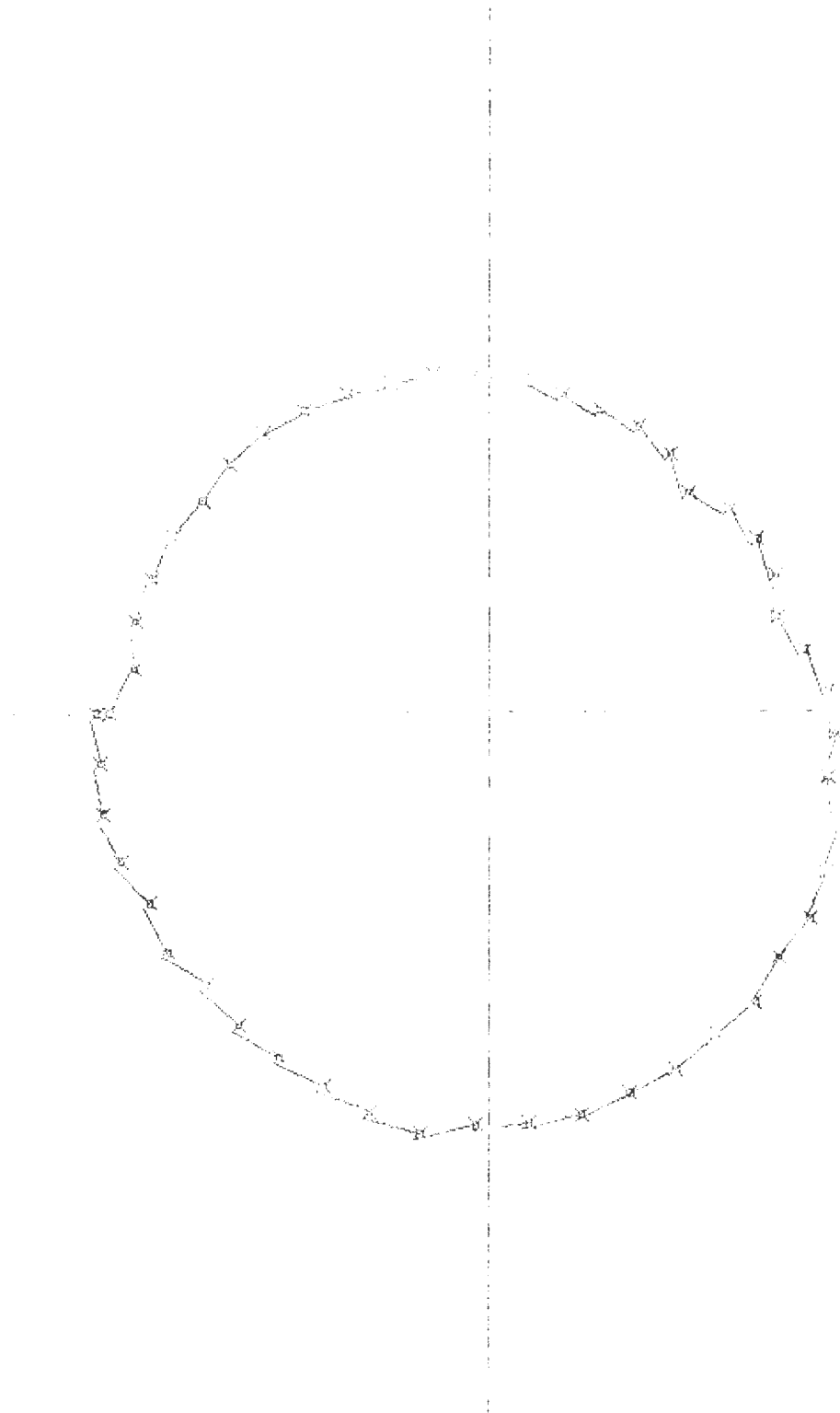
16.0 MHZ. HZEND



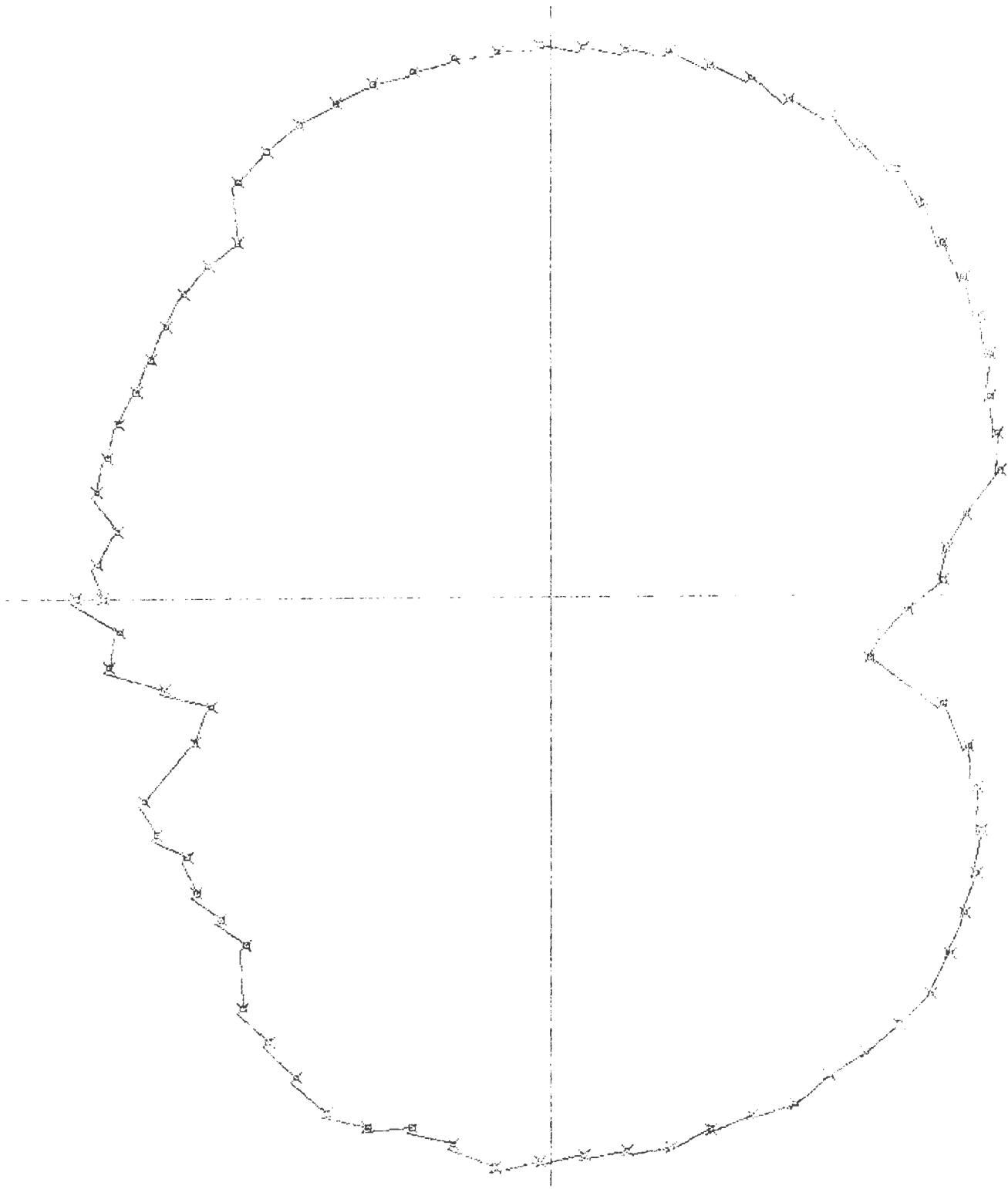
15.0 MHz. HPBRD



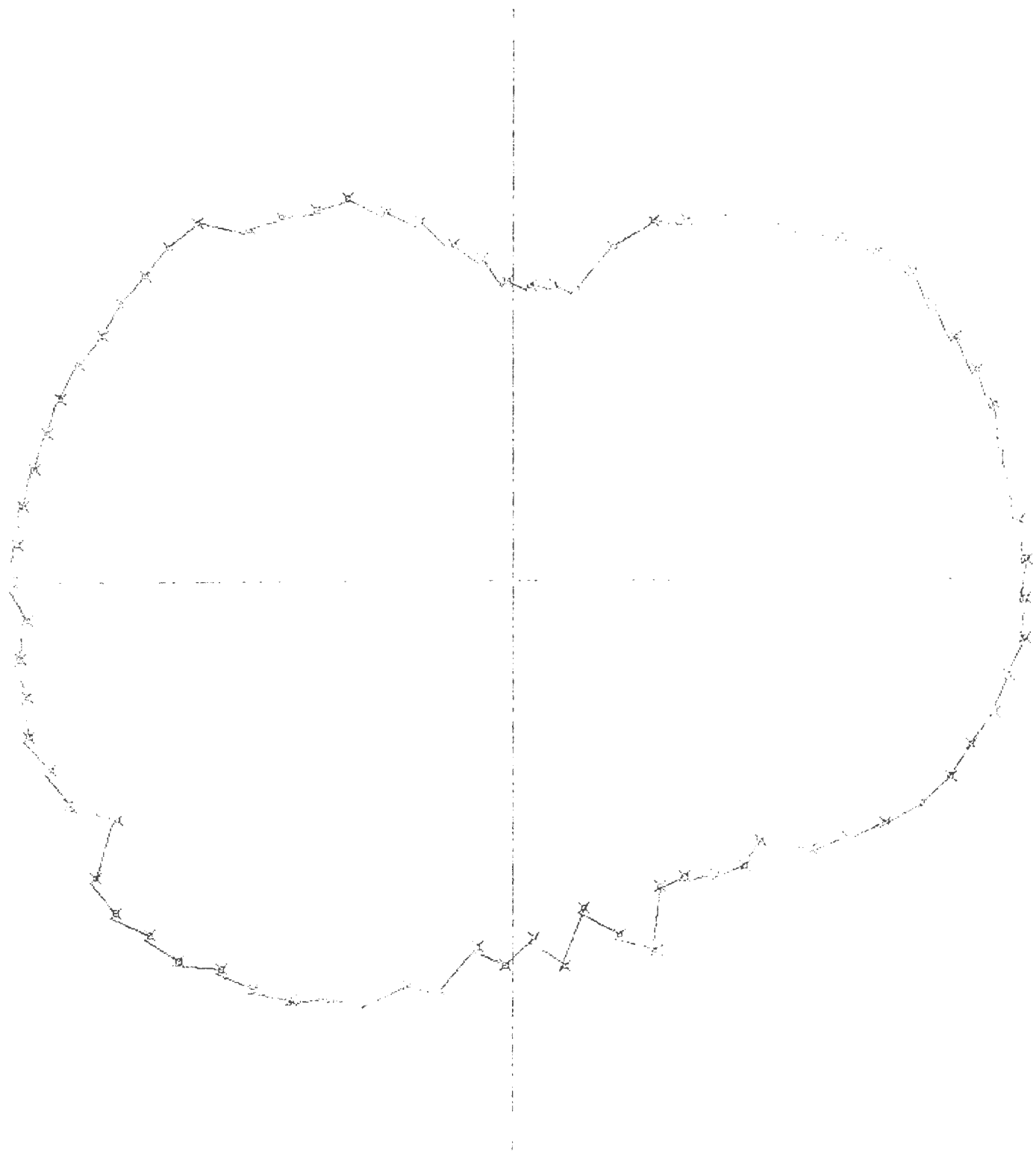
16.0 MHz. H ϕ 880



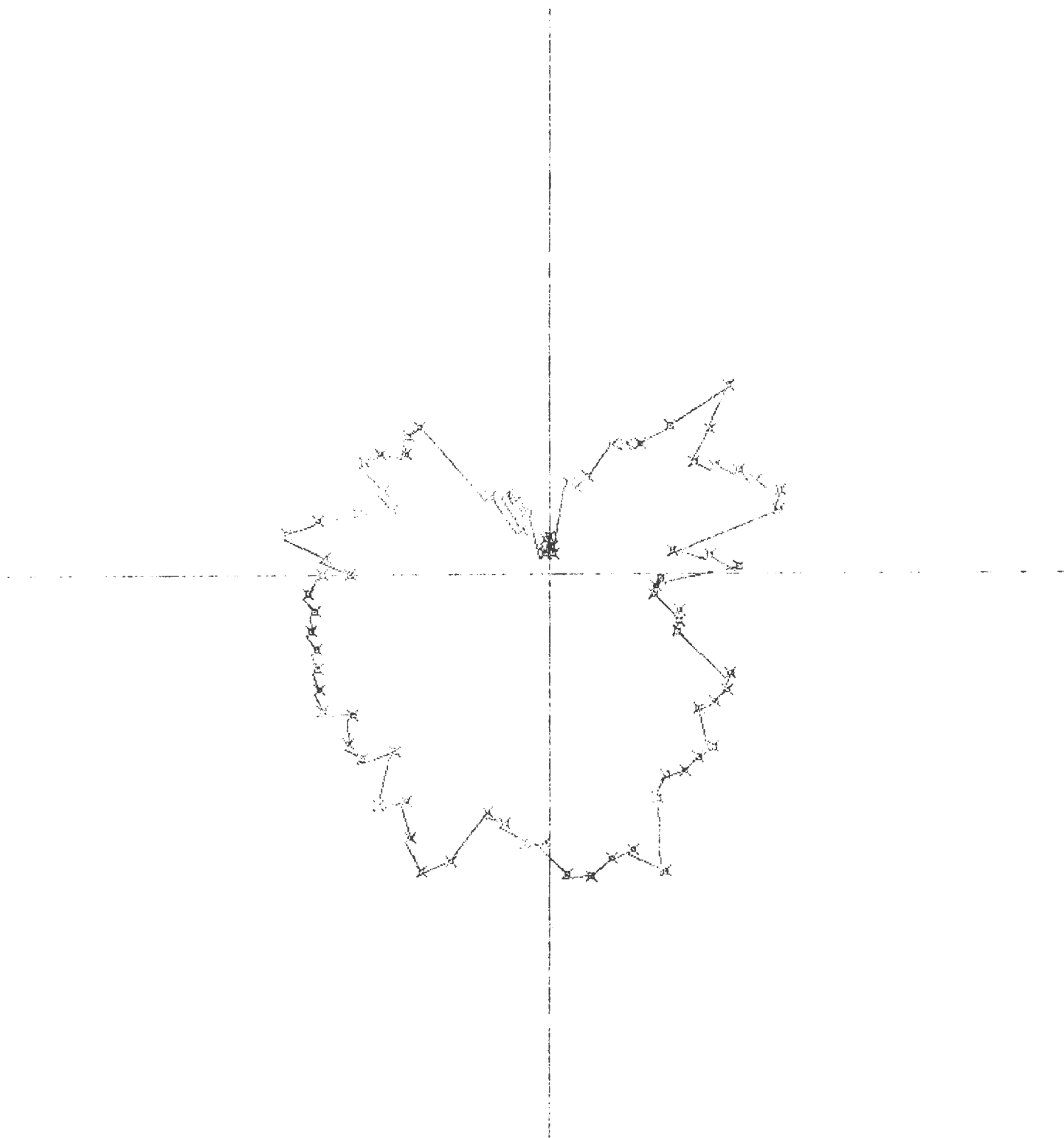
15.0 MHZ. HZBRD



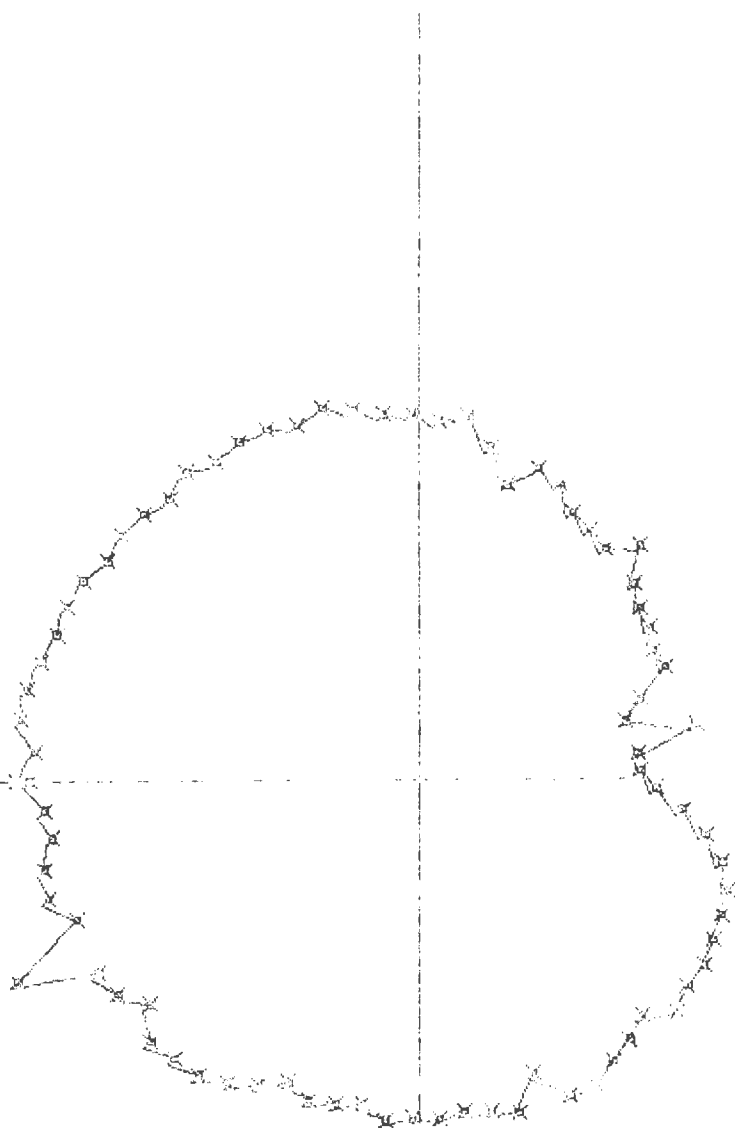
32.1 MHZ. HPEND



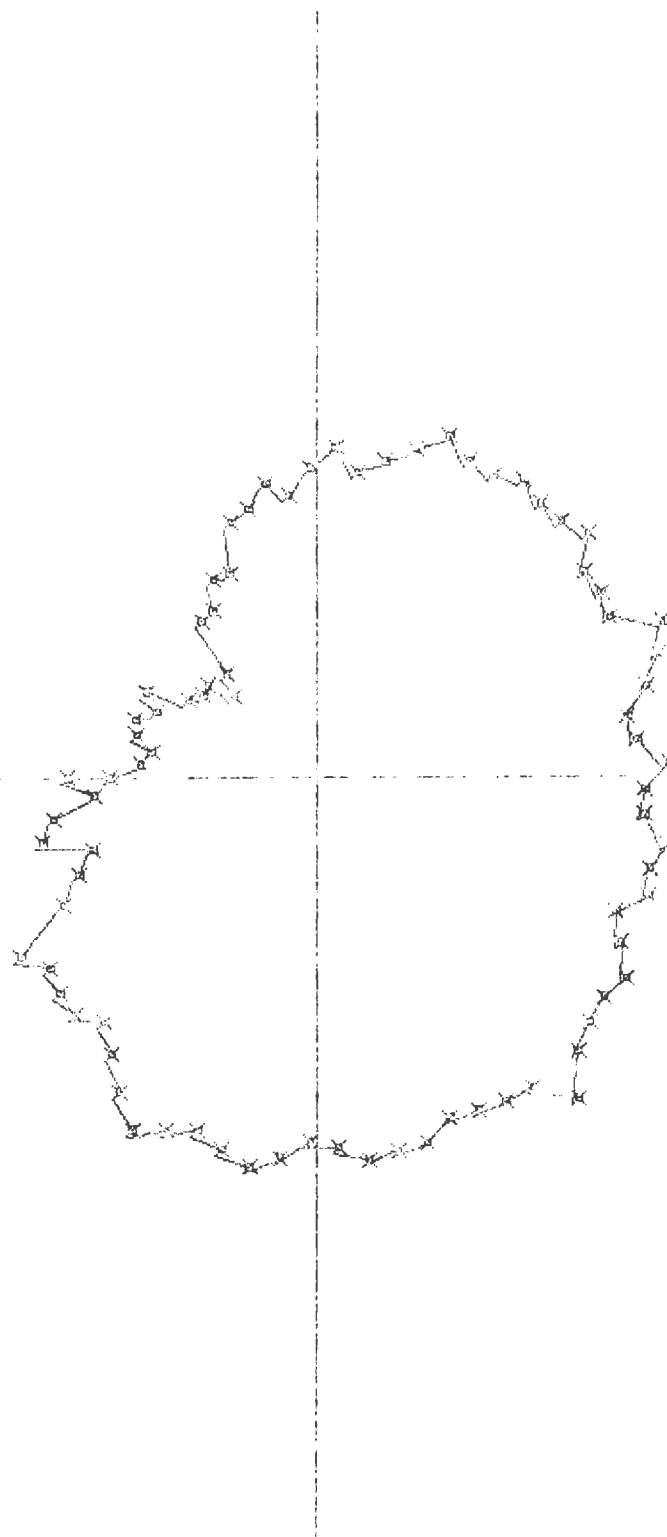
32.1 MHz. HOEND



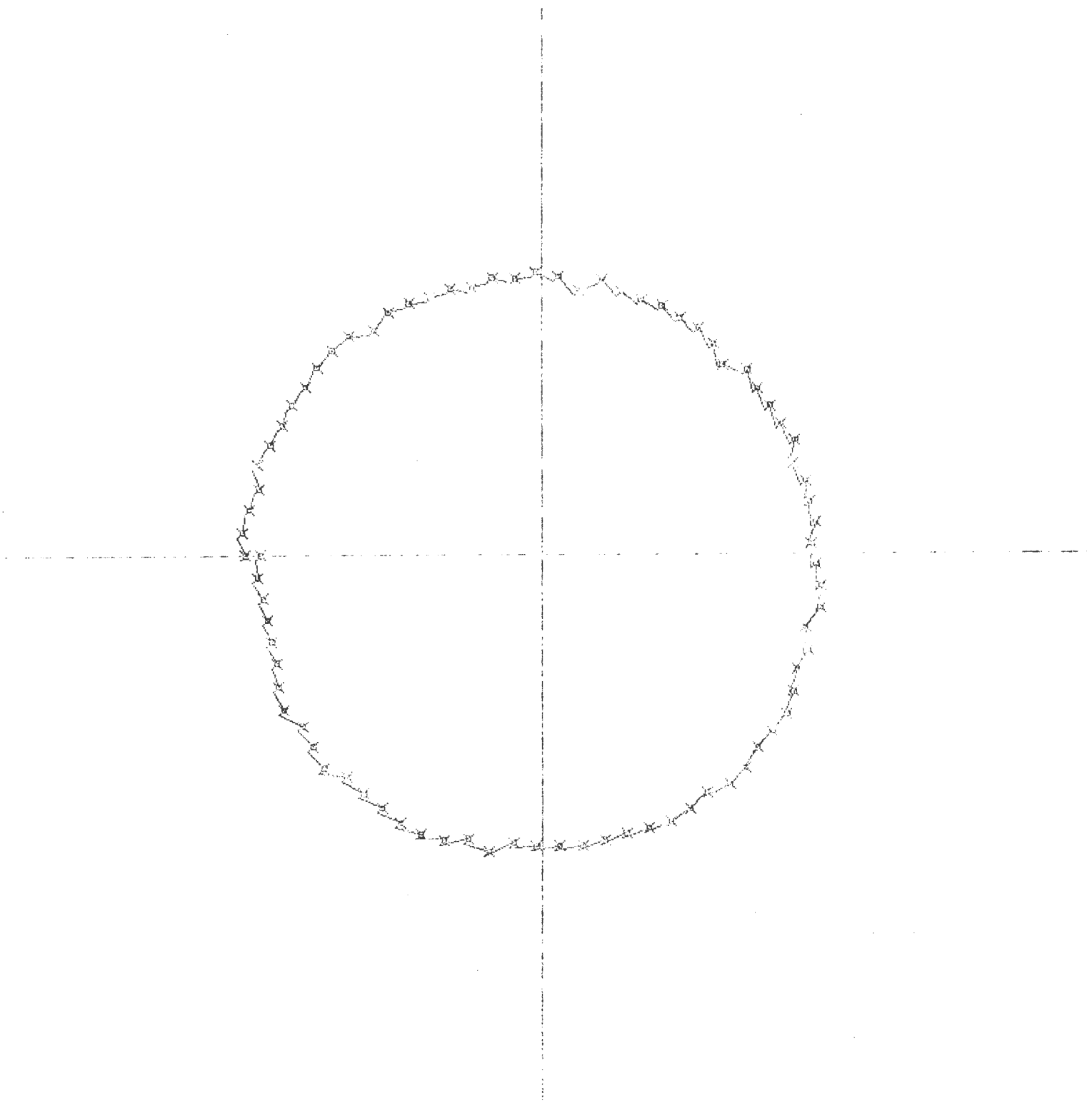
32.1 MHZ. HZEND



32.1 MHz. H[?]BRD



32.1 MHz. H ϕ BRD



32.1 MHZ. HZBRD

