NATHESON

NOAA Technical Memorandum ERL SEL-68

HIGH FREQUENCY RADAR SOFTWARE
REFERENCE MANUAL FOR PRODUCT TWO

Space Environment Laboratory
Boulder, Colorado
March 1984

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Research Laboratories

NOAA Technical Memorandum ERL SEL-68

HIGH FREQUENCY RADAR SOFTWARE
REFERENCE MANUAL FOR PRODUCT TWO

David C. Walden
James R. Winkelman
Lorne D. Matheson
Richard N. Grubb
David C. Walden, Editor

Space Environment Laboratory
Boulder, Colorado
March 1984

UNITED STATES DEPARTMENT OF COMMERCE

Malcolm Baldrige, Secretary

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

John V. Byrne, Administrator

Environmental Research Laboratories

Vernon E. Derr Director

NOTICE

Mention of a commercial company or product does not constitute an endorsement by NOAA Environmental Research Laboratories. Use for publicity or advertising purposes of information from this publication concerning proprietary products or the tests of such products is not authorized.

ACKNOWLEDGMENTS

Although many people have made significant contributions to the original design and continued development of the HF Radar system hardware and software, we wish here to recognize and thank those members of SEL and other organizations who have given material assistance in the planning, coding, and debugging of the HF Radar software, the subject of this document: Lynn Jacobson, Mike Jones, Dick Kressman, Ian McNulty, Gregg Merrill, Prentice Orswell, Ivor Page, Larry Schultz, Dyke Stiles, John Taylor, Steve Walter, and Keith Wright.

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

1. INTRODUCTION

Richard N. Grubb
David C. Walden

1.1. General

The Space Environment Laboratory of NOAA has developed a general purpose High Frequency (HF) Radar system capable of making most of the measurements of the ionosphere that can be made by coherent monostatic or bistatic radio wave sounding. This capability is provided by combining a very flexible, frequency-agile transmitter and receiver system with a digital control and signal processing system containing a general purpose 16-bit minicomputer for operator interaction and control.

This manual describes the software operating system written by the SEL staff to permit the Radar to be used for quite a wide range of standard measurements under simple operator control. The control language enables the user to exploit most of the capabilities of the instrument without having to program the system in detail. The manual is primarily directed at the user who needs to understand and use this capability. Secondarily, if used in conjunction with the computer manufacturer's system and language manuals, the SEL HF Radar Hardware manuals and the SEL source code listings, it should enable an experienced user to customize the software for special purposes or to produce new operating software.

1.2. The Hardware System

Details of the hardware system are given in Volumes 1-7 of the SEL HF Radar manual. The following very simplified description should enable the reader to understand the way the software controls the system.

The radar hardware contains a high power (10kW peak) pulse transmitter and a pair of receiver channels. Each receiver channel consists of an antenna multiplexer, filters, mixers and IF amplifiers, quadrature detectors and analog filters, and two analog-to-digital converters. A separate 64-channel analog-to-digital multiplexer monitors various functions (temperatures, voltages, forward and reflected pulse powers, etc.). Two microprocessor units (discussed below) are responsible for the operation and timing of the radar.

1.2. The Hardware System

The system computer that controls the overall operation is an Interdata 7/16 with the High Speed ALU option or an 8/16 minicomputer. Both have floating point hardware. Their memory consists of 64K directly-addressable bytes (8 bits/byte). The computer has 16 general purpose registers, 15 of which can be used for indexing. Its instruction set is similar to that of an IBM 360. The memory cycle time is one microsecond for the 7/16 and 750 nanoseconds for the 8/16.

The timing and control of the transmitter-receiver unit is performed by a microprocessor system called the Timing Sequence Generator (TSG). The TSG controls the transmission of a sequence of pulses, initiates data acquisition, and maintains timing for the entire instrument. The TSG performs these functions by executing programs in its memory that are selected by the system computer. The timing and sequencing control functions reside in the TSG because it can maintain microsecond accuracy in parallel with and independently of the system computer.

The preliminary data processing is done in a second microprocessor called the Front End Processor (FEP). The FEP receives data from the Receiving system and operates on them, in general passing on a subset of the data to the system computer. The FEP, like the TSG, is also a microcomputer system that runs in parallel with the system computer. The system computer loads programs into the FEP and the TSG selects the FEP program and fills the FEP input memory. The transfer of data from the FEP into the Interdata computer is done via a Direct Memory Access (DMA) device called EKO (radar ECHo data Output). The computer operating system controls the TSG and FEP as peripheral devices attached to the DIO bus. The DIO bus is a medium speed, locally produced bus used for general input and output of 8-bit bytes to other parts of the system under Interdata computer control. A maximum of 63 instruments can be addressed on the DIO bus.

Several standard peripherals serve the radar: a system console with a graphics Cathode Ray Tube (CRT) display and a full ASCII keyboard; a dual-platter, moving-head disc with a total capacity of 10 megabytes; a Model 33 teletype for system logging, diagnostics, and system boot-up; and a refresh CRT, or system display, used to display data under computer control.

1.3. The Operating System

The overall operation of the computer software is under the control of an Operating System (OS) called Multi-Tasking 2 (MT2), which was purchased along with the computer from the Interdata Corporation. The OS allows program modules to be coded in the form of foreground or background tasks. There can be up to 125 foreground tasks and one background task operating under the control of the OS.

The OS consists of three major modules called Executive, File Manager, and Command Processor. Parts of all modules can be overlayed; that is, portions of them can reside on the disc until they are needed. When the OS needs the disc-resident portions of the file Manager or the Command Processor, it reads them into memory for execution. A non-overlayed version of MT2 occupies about 2/3 of the system computer memory. However, a selectively overlayed version of MT2 requires only about 1/3 of available memory; unfortunately, it has slower operator response than the non-overlayed system. A maximally overlayed system, capable of sounding at the usual rate, can be fit into 1/4 of the available memory.

There are many options available to the user to define the specific version of MT2 that is desired. The options are specified at system generation time, and they normally cannot be changed at program execution time. Depending upon the amount of documentation required, system generation time will vary from about 30 minutes to about 5 hours of computer console time. The OS is described in the Interdata manual, OS/16 MT2 Program Logic Manual. The user wishing to generate a new system should consult another Interdata manual, OS/16 MT2 System Planning and Configuration Guide. Alterations to the OS made by SEL are described in Chapter 11 of this manual.

1.4. Software Products

This manual describes the second generation of operating software for the HF Radar, usually referred to as Product Two. Product Two is based on the experience gained with the original Product One software and adds many new capabilities as well as formally incorporating most of the various enhancements which were produced for Product One. There are marked functional differences between the two systems and the Product One software manual should be consulted if that system is used.

1.4. Software Products

Product One provided two basic modes of operation: a logarithmic frequency scan or Ionogram mode and a multiple fixed frequency sounding or Kinesonde mode. In both these modes the software detects specific echoes which meet echo recognition criteria and characterizes these as though they are "mirror-perfect" reflections.

Product Two has generalized these modes into Sweep-class and Table-class and has added a variation of T-class called W-class. The sweep modes permit complex patterns to be generated by combining multiple logarithmic and linear frequency sweeps so that the frequency and time resolution can be traded and optimized for particular applications. In addition, Product Two integrates the capability, previously available only in special modes of Product One, to manipulate and record the received data without echo recognition. This permits more complete offline analysis of, for instance, wave motions in the D region. Product Two also makes available a more complete suite of online display options and a powerful method of saving and recalling operating modes as well as a new set of more powerful Front End Processor programs for signal averaging, echo recognition, and pulse decoding. In many modes this now permits the system to run at 100 pulses per second for improved time resolution. It is also now possible to control the DIO configuration from the console.

1.5. Task Interrelations

Figure 1.1 (courtesy Beth Walden) shows the interrelationships between the software tasks in Product Two and the physical system. This is identical to that employed in Product One, except that some of the tasks occupy different memory partitions. Each physical device attached to the Interdata computer system is attached to one task. In the few cases where more than one software task uses the same device, coordination is necessary.

1.5.1. SOUNDER

The SOUNDER task controls the transmitter-receiver instruments (frequency synthesizer, RF and IF attenuators, bandpass filters, bandwidth settings) and the TSG-FEP programs. Upon receipt of certain hardware interrupts, SOUNDER sends an address to PICKER pointing to data in the computer memory available to PICKER. SOUNDER accepts scientist-operator commands from COUNSEL to the radar system. SOUNDER also provides access to the CALL system described in Chapter 10 and can load various configurations of the PICKER task, as required by different sounding modes.

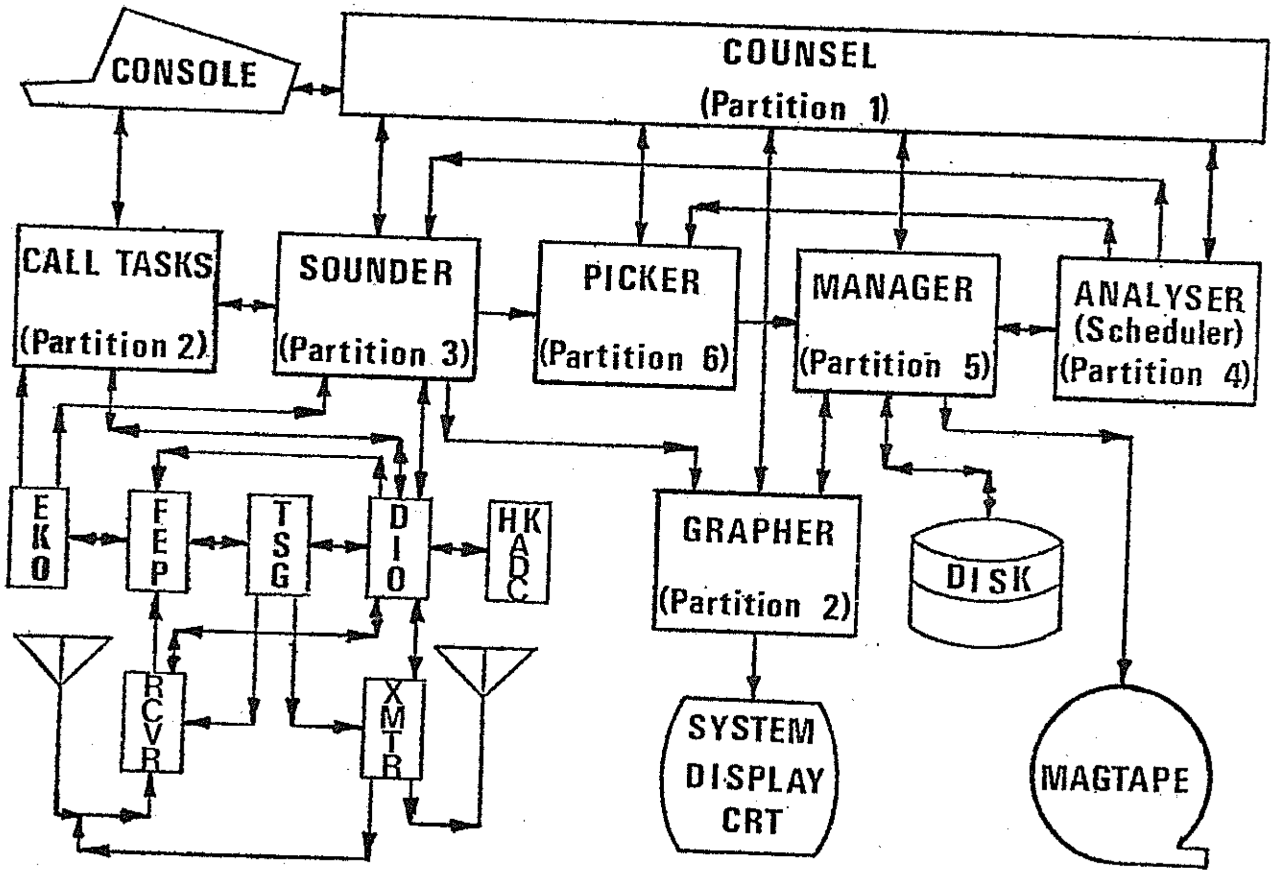


Figure 1.1. PRODUCT TWO BLOCK DIAGRAM

The CALL (previously TEST) feature allows special purpose capabilities by displacing the GRAPHER task with another task. The original application was to execute system diagnostic and test programs, but the mechanism can be used to create special purpose data acquisition modes that do not require PICKER, MANAGER, or GRAPHER.

1.5.2. PICKER

PICKER carries out the processing and formatting necessary between the output of the Front End Processor (FEP) and the input to MANAGER. Depending on the mode of operation selected, this can range from carrying out the majority of the echo recognition and formatting tasks (as in Product One) to merely completing the formatting of data that is entirely specified in the FEP.

Upon receipt of a queue from SOUNDER, PICKER has the job of either passing all of the data on to MANAGER, or of selecting a subset of the data, thus acting as a data filter. When PICKER has selected its data, it sends MANAGER the location of the data. PICKER also informs SOUNDER that it has completed its job so that SOUNDER can reclaim the memory buffers that it made available to PICKER.

1.5.3. MANAGER

MANAGER transfers the data from PICKER to the disc. MANAGER also sends the location of the PICKER data to GRAPHER and ANALYSER. When tape is selected, MANAGER writes the data from disc to tape between soundings. MANAGER is the only task with access to the disc during an actual sounding.

1.5.4. GRAPHER

GRAPHER converts the compacted data to plotter coordinates and puts the plot data into a memory buffer. The plot buffer data are then sent to the CRT refresh device via an Interdata DMA SELector CHannel (SELCH) that it shares with tape; this SELCH is separate from the EKO SELCH. GRAPHER's display of data has no effect whatever on that data as stored on disc or magnetic tape.

1.5.5. COUNSEL

Figure 1.1 shows that all tasks accept queues from COUNSEL. This task interacts mainly with the operator of the radar through the system console. COUNSEL also has the function of initiating task execution and system logging.

1.5.6. ANALYSER

This task was originally intended to provide the capability of adding user data interpretation routines. It has not been used for this purpose because of the very tight constraints on memory and time within the machine. SEL provides a general purpose system scheduler which is run within this partition. This task is named ASKED. See SCHED in Section 10.2.5 for details of its operation.

1.6. Intertask Communication

The HF Radar tasks communicate with each other using a system provided by the Interdata MT2 Operating System (OS). Each task maintains a circular list, or queue, to which any other foreground task can add a 16-bit address by invoking the OS Supervisor Call 6 (SVC6). (SVC4 is a fast version of the SVC6 queueing function added to the OS by J. R. Winkelman of SEL, and SVC4 is normally used by the HF Radar tasks for intertask communication instead of SVC6.) The address added to a task's queue is that of an information block of variable length, referred to as a queue block or Q-block. The process of one task's adding a Q-block address to another task's queue is referred to as queueing. The completion of an I/O and procede call or a call to the OS delay may also cause an entry to be added to a task's queue. However, only intertask queueing is discussed in this section.

The Q-block normally contains the following information. Byte Ø is an even integer (including Ø) defining the function of the Q-block. Byte 1 is the ASCII identifier of the queueing task, "C" for COUNSEL, "S" for SOUNDER, etc. The status field (bytes 2 and 3) is initialized by the queueing task to -1ØØØ (hex). The remaining bytes contain commands or addresses, or both, which direct the response of the queued task. Section 7.8 contains detailed information about all of the Q-blocks used by the HF Radar tasks.

The initial status field value gets changed as follows. The queued task signals the queueing task that the Q-block has been used by adding 1ØØØ + N (hex) to the Q-block's status field, where N is an error condition code and the new value of the status field. This addition should always be done with the AHM instruction, not in registers, to prevent tasks that share a Q-block from interfering with one another's status clearing. (See Example 3, below.) A condition code of N = Ø indicates no error, which tells the queueing task that the Q-block information has been received and that the recipient (queued task) is done with it. A Q-block status greater than Ø signals the queued task's detection of an error associated with the Q-block information. A Q-block status less than Ø indicates that whatever action the queueing prompted is incomplete.

Each of the three following paragraphs describes an example of intertask communication, the first from COUNSEL to SOUNDER, the second from SOUNDER to PICKER, and the third from MANAGER to both GRAPHER and ANALYSER.

Example 1. To change the starting frequency of a sounding sweep, the operator/experimenter types: SFRQ f, where f is the desired new starting frequency expressed in decimal kHz. The COUNSEL task, having read this command from the console keyboard and determined that it is advice to the SOUNDER task, adds the address of a Q-block to SOUNDER's task queue with an SVC4 call. Byte 1 of the Q-block contains ASCII "C" (43 hex) because COUNSEL did the queueing, and byte Ø contains Ø2 to identify an advice information block. We use the first two bytes in reverse order to name the Q-block "CØ2". Bytes 2 and 3 are initialized by COUNSEL to -1ØØØ (hex). The remaining four bytes of the CØ2 Q-block contain the index of the SFRQ command in SOUNDER's command scan table and the byte address of the ASCII string, f, the new starting frequency read into COUNSEL's console input buffer. If SOUNDER determines that frequency f is a valid starting frequency for a sounding sweep, it incorporates this information into the Sounding Configuration Table (SCT, Section 7.1) in field MD.SFRQ and adds 1ØØØ (hex) to the CØ2 Q-block's status field. If f is not a valid frequency, SOUNDER adds 1ØØ2 (hex) to the CØ2 status. COUNSEL waits for its Q-block status to become non-negative. If it becomes Ø, COUNSEL simply displays its prompt and readies itself for more console input, knowing that SOUNDER has accepted the SFRQ advice. If the CØ2 Q-block status becomes greater than Ø, COUNSEL knows that SOUNDER has detected an error in the advice given and displays a message, in this case ERR 2, to alert the operator.

Example 2. When SOUNDER has a new block of echo data ready to deliver to PICKER during a sounding, it informs PICKER by adding the address of an SØ2 Q-block to PICKER's task queue. The Ø2 in byte Ø in this case is the function code for a sounding data information block. An "S" in byte 1 identifies SOUNDER as the queueing task. The status (bytes 2 and 3) again is initialized to -1ØØØ. Bytes 4 and 5 contain the address of the echo data buffer, bytes 6 and 7 contain the address of the corresponding Pulse Configuration Table (PCT, Section 7.2), and bytes 8 and 9 the address of the appropriate frequency table entry (SCT.FTBL, Section 7.4), if the sounding is T- or W-class. SOUNDER maintains 6 buffers with corresponding PCTs and SØ2 Q-blocks and uses them in round-robin fashion, so the potential exists for SOUNDER to begin filling a buffer with new echo data before PICKER has finished with the old. The prevention of such a condition is effected via the SØ2 Q-blocks' status fields. PICKER will not clear an SØ2 status to Ø (by adding 1ØØØ) until it has finished with the buffer associated with that SØ2 Q-block. SOUNDER will not re-use that buffer or its PCT unless its SØ2 status has been cleared to Ø, which is PICKER's signal to SOUNDER that the buffer has been released. An SØ2 Q-block's status field is never greater than Ø. A buffer is either free (status Ø) or busy (status less than Ø).

Example 3. When PICKER has assembled the data from a pulse set (Section 2.5), it notifies MANAGER with the PØ2 Q-block. MANAGER's job in response to the PØ2 queue is to store the data on the disc, but GRAPHER must be notified in order to display the data, and a user-written ANALYSER task must have access to the data for its purposes. Therefore, MANAGER "passes" each PØ2 Q-block that it receives on to both the GRAPHER and ANALYSER tasks. MANAGER's mechanism for passing on a PØ2 Q-block is simply to subtract 2ØØØ (hex) from the status of the PØ2 Q-block. Because the PØ2 status is initialized by PICKER to -1ØØØ, it becomes -3ØØØ. Each of the three tasks (MANAGER, GRAPHER, and ANALYSER) concerned with the data subsequently add 1ØØØ to the PØ2 status when their respective jobs are complete. Only when all three have finished with the PØ2 data does the PØ2 status become Ø, notifiying PICKER that the buffer associated with the PØ2 Q-block is free to accept new data. Notice that the order in which the three tasks complete their functions is unimportant. The PØ2 remains busy as far as PICKER is concerned until all have added 1ØØØ to the PØ2 status. If MANAGER has been informed that ANALYSER has no interest in the sounding data, MANAGER subtracts 1ØØØ from each PØ2 Q-block status and passes the PØ2s on to GRAPHER only. (See Chapter 4 and command NOQUEUE, Chapter 8.) Thus, the general rule is that for each task to which it passes the PØ2 Q-block, MANAGER subtracts 1ØØØ from the Q-block status. MANAGER passes the sounding-start (SØ4) and the sounding-end (SØ6) Q-blocks to GRAPHER and ANALYSER in like manner.

2. SOUNDER

David C. Walden
James R. Winkelman

2.1. Introduction [DCW]

Chapter 2 describes SOUNDER revision 2.Ø2, which logs in dated 1Ø/Ø7/83. For further details about changes made to SOUNDER 2.ØØ and 2.Ø1, refer to Appendix A, dated 1983 November.

The SOUNDER task directs the sequence of events that results in the acquisition of radar echo data into the Interdata computer memory. In so doing, SOUNDER interacts with the radar transmitter-receiver hardware, the Timing Sequence Generator (TSG) and Front End Processor (FEP) microcomputers, a 64-channel "House-Keeping" Analog-to-Digital Converter (HKADC), and the Direct Memory Access (DMA) link between the FEP and the main system computer. This DMA link, operating under an Interdata SELector CHannel (SELCH), is referred to as the EKO device. SOUNDER communicates with the radar hardware, the TSG, FEP, and HKADC by way of the 8-bit Digital Input/Output bus (DIO). Device starts and interrupts for the TSG, FEP, and EKO are handled at the Operating System (OS) level by locally written software drivers; the control of the transmitter-receiver hardware, the reading of the HKADC and of the TSG clock, and the loading of the TSG and FEP memories are handled directly at the SOUNDER task level via the DIO bus.

SOUNDER maintains the global HF Radar system tables, the Environment Table (ET, Section 7.3), the Sounding Configuration Table (SCT, Section 7.1), and the DIO control tables (DIO, Section 7.9), and constructs a Pulse Configuration Table (PCT, Section 7.2) for each transmitted pulse that produces echo data. As each buffer of radar echo data is received via the EKO device from the FEP, SOUNDER passes the buffer's address, together with the address of the associated PCT, to the PICKER task, which in turn passes selected data to MANAGER, GRAPHER, and to ANALYSER.

SOUNDER also provides access to a collection of diagnostic and support tasks, called CALL tasks, that can be run between soundings (Section 1Ø.2), or that can provide alternative data recording modes during sounding (Section 1Ø.3).

2.2. Hardware Interrupts and Timing [DCW]

The sounding software for SOUNDER is designed around the interrupts received from the TSG, FEP, and EKO. SOUNDER invokes Supervisor Call 1 (SVC1) to start these devices and to wait for interrupts from them, signaling that their actions are complete. The sequence of these SVC1 calls in SOUNDER is: Start TSG, Wait EKO, Start EKO, and Wait TSG. (In P2.Ø, the FEP is not under explicit control of the SOUNDER task, but is started when required by the TSG with regard to the BUSY-DONE handshaking flag from the FEP.) The TSG is started every time this sequence of calls is executed; the EKO device is started only when the presence of data demands. (See Sections 2.3 and 2.4.) A device wait without a previous corresponding device start is ignored by the OS. This section discusses the various hardware interrupts and timing requirements of these devices.

After being commanded by the Interdata computer, the TSG waits for its own internal clock interrupt and then initiates an accurately-timed, single-frequency, sounding sequence. In P1.Ø this sequence resulted in a single pulse; the transmitter can also sound with a sequence of pulses having different phases. For a single pulse, the sending of the transmitted pulse and the loading of two FEP input memories (one per receiver channel) take a little over 5 ms from the start of the timing sequence. During this 5 ms, the system computer does not reference any of the instruments on the DIO bus because of various hardware considerations (frequency coherence, bus noise, response times, etc.).

During the time of flight of the pulses, SOUNDER calculates the parameters for the next transmitted pulse. Then it puts itself into a task-wait state until the radar echoes have returned. During this wait state, the Operating System (OS) allows other tasks to execute in order of their priorities. When the TSG has finished waiting for all the pulse echo data to be received into the input memories of the FEP, an interrupt is sent to the Interdata computer. This interrupt occurs about 6 ms after the TSG starts and is called the TSG DONE Interrupt.

The FEP, under the control of the TSG, starts its program execution when all the radar echo data from one pulse have been received. The purpose of the FEP software, when the peak-finding sounding modes are run, is to remove all the non-echo signals from the received data. (See discussion of sounding modes in MAKEM, Section 1Ø.2.1.) It does this by the techniques of signal deconvolution, thresholding, and running averages. The FEP reduces the amount of original data by at least a factor of four by thresholding. The FEP filters the radar data and transfers the data from its input memories to its output memories via the FEP scratch memories. For each receiver channel, the FEP finds the maximum peak (maximum maximorum) in the data, and the maximum peak in the determined echoes (saved data) if FEP PICKER processing is invoked, and includes it in the output for use by SOUNDER for Automatic Gain Control (AGC). The exact sequence of software operations and execution time depends upon the particular TSG and FEP programs selected. An understanding of Chapters 12 and 13 is essential to understanding exactly what happens in the TSG and FEP for any given pulse. The AGC is discussed in Sections 2.7.9, 2.9.1.1, and 2.9.1.3. Ultimately, the FEP puts the processed radar data into its output memory and signals this event via a hardware interrupt to the Interdata computer. This interrupt is called the FEP DATA READY Interrupt; it typically occurs from 2 to 5 ms after the TSG DONE Interrupt. The FEP DATA READY Interrupt enables the SELCH to effect the EKO transfer of the FEP output data. (See PIPLN, Section 10.2.15, for details.)

Upon command from SOUNDER, EKO transfers the data from the FEP output memory to a buffer in the Interdata memory. The transfer takes from 1 to 2 ms, and its completion generates the EKO DONE Interrupt.

2.3. Data Flow [DCW]

This section describes the movement of data from the radar receivers into the Interdata system computer memory. The discussion refers to Figure 2.1 (courtesy Beth Walden), which shows schematically the data flow from the antennas to SOUNDER's EKO buffers in the Interdata computer, and assumes a pulse interval of 20 milliseconds.

The echoes returned from the ionosphere are detected by antennas connected to the two receiver channels. Normally, the antennas are paired north-south and east-west. One pair of antennas is switched on line at a time. Assume for discussion the default sounding sequence of four pulses in a set at any given frequency, F. For each pulse in a set, a pair of antennas is switched on line to the two receivers for about 5 ms, this time, or data collection window, being determined by the FEP WRITE ENABLE in the TSG RF vector.

Table 2.1 illustrates the default transmission sequence. For the first pulse of a set, the N-S pair of antennas is switched on line; the frequency synthesizer is tuned to frequency F. For the

2.3. Data Flow

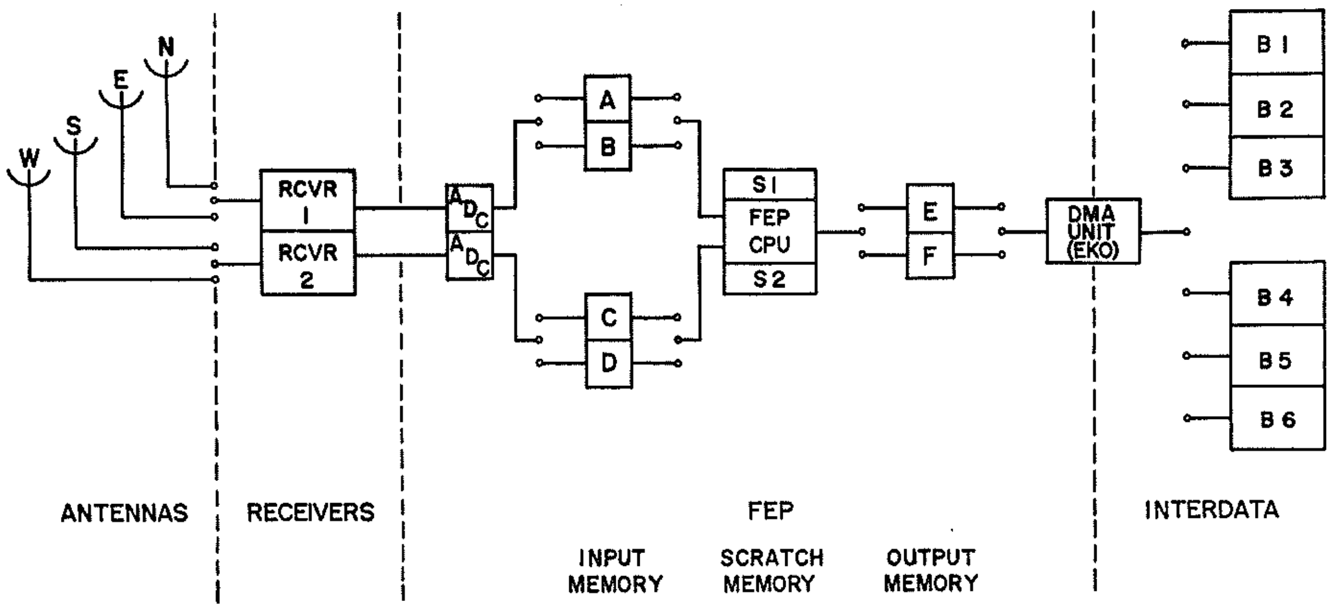


Figure 2.1. DATA FLOW CHART.

second pulse, the E-W antenna pair is switched on line, and frequency F + delta F is selected, where delta F is chosen to optimize group delay measurements. For the third pulse of a set, the frequency is held at F + delta F, and the N-S antenna pair is selected. Finally, for the fourth pulse, the E-W antenna pair and frequency F are selected. Thus, the antenna-receiver switches in Figure 2.1 toggle at every pulse.

During the 5-ms receiving period for each pulse, the received data are digitized by the analog-to-digital converters in the FEP. These ADCs produce one X-Y pair of 12-bit data words every 1Ø microseconds. (The current PICKER allows 2Ø microsecond digitizing to be selected for FEP PAR processing only. See Section 3.1.) The TSG routes these digital data to one of the two pairs of FEP input memories. These are labeled as memory pair AC and BD in Figure 2.1. As illustrated by Figure 2.1, the N-S antenna data for pulse 1 at frequency F go into FEP input memory pair AC, and the E-W antenna data for pulse 2 at frequency F + delta F go into memory pair BD. Pulse 3 and 4 data in turn go into input memories AC and BD, respectively. This means that the input memory switches in Figure 2.1 toggle at every pulse.

The TSG controls selection of the FEP input memories after testing the FEP BUSY-DONE flag. The FEP moves selected data from its input memories to its output memory. When data are ready for transfer into the Interdata computer, the Direct Memory Access (DMA) device (EKO) is started by SOUNDER.

The flow of data from specific antenna pairs into the Interdata computer is now considered for successive time intervals. Assume for discussion that pulses are transmitted uniformly in time into

PULSE NO.	FREQUENCY	ANTENNAS	ELAPSED TIME, ms
1	F1	N-S	Ø
2	F1 + deltaF	E-W	2Ø
3	F1 + deltaF	N-S	4Ø
4	F1	E-W	6Ø
5	F2	N-S	8Ø
6	F2 + deltaF	E-W	1ØØ
7	F2 + deltaF	N-S	12Ø
8	F2	E-W	14Ø
9	F3	N-S	16Ø
1Ø	F3 + deltaF	E-W	18Ø
11	F3 + deltaF	N-S	2ØØ
12	F3	E-W	22Ø

Table 2.1. TRANSMISSION SEQUENCE EXAMPLE.

the ionosphere according to Table 2.1. In the example, the operator has specified the north-south and east-west antenna pairs.

2.4. Data Pipeline Example [DCW]

The preceding ionospheric sounding schedule results in the data pipeline structure displayed in Table 2.2. The four major processes in the pipeline are: (1) the transfer of data from the receiver into the FEP input memories (IM); (2) the processing of input data within the FEP and the loading of FEP output memory (OM) with processed data; (3) the transfer of data from the FEP output memory to the Interdata (ID); and (4) the data processing within the Interdata computer. These processes form the major column headings of the table. The rows of the table correspond to consecutive pulse periods.

During Period 1, data from Pulse 1 at frequency F1, P1D (i.e., pulse 1 data), are recorded on the N-S antennas, digitized, and are transferred into FEP input memory pair AC as data AC1.

2.4. Data Pipeline Example

PERIOD NUMBER	PROCESS 1 RCVR to IM	PROCESS 2 IM to OM	PROCESS 3 OM to ID	PROCESS 4 ID BUFFER	FREQUENCY
1	P1D -> AC1	-	-	-	
2	P2D -> BD1	AC1 -> E1	-	-	-
3	P3D -> AC2	BD1 -> F2	E1 -> B1	-	
4	P4D -> BD2	AC2 -> E3	F2 -> B2	B1	
5	P5D -> AC3	BD2 -> F4	E3 -> B3	B2	F1
6	P6D -> BD3	AC3 -> E5	F4 -> B4	B3	
7	P7D -> AC4	BD3 -> F6	E5 -> B5	B4	
8	P8D -> BD4	AC4 -> E7	F6 -> B6	B5	
9	P9D -> AC5	BD4 -> F8	E7 -> B1	B6	F2
1Ø	P1ØD -> BD5	AC5 -> E9	F8 -> B2	B1	
11	P11D -> AC6	BD5 -> F1Ø	E9 -> B3	B2	
12	P12D -> BD6	AC6 -> E11	F1Ø -> B4	B3	
13	-	BD6 -> F12	E11 -> B5	B4	F3
14	-	-	F12 -> B6	B5	
15	-	-	-	B6	

Table 2.2. DATA-FLOW PIPELINE EXAMPLE.

During Period 2, Pulse 2 echo data from the E-W antennas at frequency F1 + delta F are transferred into FEP input memory BD as data BD1. Also during this period, FEP input memory data AC1 are being processed and the results placed in FEP output memory E via the internal scratch memory. The processed data in the output memory are labeled E1. This latter transfer occurs while the alternate input memory is being filled. The FEP was designed so that one input memory pair can be processed while the other input memory is being filled. At the end of this period, one input memory pair (BD) and the output memory (E) are full.

During Period 3, Pulse 3 echo data from the N-S antennas at frequency F1 + delta F go into memory pair AC as data AC2. As data are being received from the antennas, the data from Pulse 1 that are stored in the FEP output memory are transferred into a memory buffer in the Interdata computer as data B1. During this transfer, the BD1 data from Pulse 2 are transferred into the alternate output memory F and labeled F2. At the end of this period, the pipeline contains Pulse 1 data in the Interdata computer, Pulse 2 data in the FEP output memory, and Pulse 3 data in the FEP input memory.

During Period 4, Pulse 4 echo data are received via the E-W antennas at frequency F1 and are transferred into the BD memory pair as data BD2. Pulse 2 data are transferred from the F output memory into another Interdata buffer as data B2. Pulse 3 data AC2 are transferred from the FEP input memory pair AC into the output memory as data E3. The data from the first 4 pulses are now in the pipeline.

By the beginning of Period 7, data from the first four pulses have all been transferred into the Interdata computer. These data in Interdata buffers B1 through B4, taken over a time period of 80 ms, constitute one set of data, or pulse set, associated with one frequency sounding of the ionosphere, namely, frequency F1. The data set contains eight peak maxima, one per pulse for each of two channels (see the description of the FEP in Section 2.2). SOUNDER uses the greatest of these maxima to control the attenuation in the receivers. (See commands AGC, PKHI, and PKLO, Section 2.7.9.)

Task PICKER starts processing B1 as soon as it is received. When PICKER finishes with B1, it releases B1 to SOUNDER. Normally PICKER finishes B1 before B2 is ready, and lower priority tasks can execute. If PICKER needs more time than what remains of the pulse pace interval after SOUNDER has run, PICKER can run into the next interval. PICKER normally catches up, but if it doesn't, SOUNDER delays one pulse time and then finds out if it has enough buffers to continue.

At the beginning of Period 11, two complete frequency soundings have been made. Four buffers need to be available for the third frequency sounding. Table 2.2 gives the data currently in the pipeline during this and subsequent periods.

2.5. SOUNDER Design and Organization [DCW]

SOUNDER 2.02 was designed with the following main objectives in mind.

1. To provide for more flexible frequency selection and sounding mode definition schemes without sacrificing convenience for routine operations.

2. To provide more flexible and complete hardware control, again without making routine operations a burden.

3. To provide a structure that would accomodate dynamic, or adaptive, sounding. That is, to allow the data from a given sounding to influence the configuration of subsequent soundings.

4. To accomplish the above objectives with a minimum impact on the other HF Radar sounding and support tasks, as well as upon future off-line software.

5. To double the HF Radar Pulse Repetition Frequency (PRF) from 5Ø to 1ØØ pulses per second.

The implemention of the above objectives is discussed in the following paragraphs. Although this document is meant to stand alone, references will be made to the Product One (P1.Ø) design when clarification demands.

The basic unit of a sounding is the individual pulse, and at the heart of the SOUNDER 2.Ø2 design is the concept of the pulse palette. Just as the artist can mix paints with the value, hue, and chroma of his choosing, so can the HF Radar operator define the attributes of the 8 pulses that constitute the pulse palette. (See SCT.PD, Section 7.1 and Section 7.5.)

The principle "ingredient" used in defining a given pulse is the TSG command, an 8-bit byte that, when sent to the TSG by SOUNDER, initiates the pulse and the sequence of events that results in the acquisition of echo data. Specifying a TSG command byte has far reaching implications. These implications are listed below.

1. The basic rate at which the transmitter is keyed.

2. The rate at which incoming data are digitized.

3. The selection of a timing scenario, which specifies the timing, duration, and phasing of the transmitted and calibration pulses; the volume of the acquired data and their ranges relative to the transmitted pulse; and the level, timing, and duration of fast attenuation of the receivers.

4. The phasing of the receivers.

5. The selection of one of several FEP programs available for pre-processing the digitized data.

6. The amount of coherent averaging to be done in the FEP for each time the TSG is started.

7. The type of processing done by PICKER on the data after the FEP has finished its job.

Other essential pulse attributes are the frequency at which the RF system is tuned at the time of the pulse, and the antenna inputs into the two receiver channels. All of the above information is contained, directly or by inference, in each pulse description of the palette. The influence of the pulse description on the frequency of the RF system is by way of the delta F factor. (See commands TSGC, DFAC, and ANT, Section 2.7.6.)

In addition, other conditions of the standard HF Radar hardware contribute to the configuration of a given pulse. Because they are computed, or because they may or may not change on a pulse-to-pulse basis, the values describing these conditions are not recorded in the pulse palette. (The rationale is to not inflate the palette with redundant information.) They are recorded, however, in either the SCT (Section 7.1) or the PCT (Section 7.2). These values are the low- and high-power transmitter attenuations (MD.TXLP and MD.TXHP); the receiver bandwidths (MD.BDW); the actual sounding frequency (PCT.FRQ1 and PCT.FRQ2), derived in part by the delta F factor, DFAC (above); the RF and IF attenuations for the receiver channels (PCT.RFIF); and the calibration attenuation (PCT.CALA). Notice that the computed values are put, pulse-by-pulse, directly into the PCT. (Frequency selection is discussed below.)

Finally, provision is made for the addition of non-standard DIO hardware devices to the HF Radar. (See DIOGEN, Section 1Ø.2.4.) Pulse-to-pulse settings for DIO devices are stored in the DIO.PTP section of the DIO control tables (Section 7.9). The settings for a given DIO device are in the same order as are the pulse descriptions in the pulse palette, and therefore the DIO.PTP table is considered to be an extension of the pulse palette. Because the number of non-standard DIO devices will vary, the DIO.PTP palette extension will be of variable length. DIOGEN (Section 1Ø.2.4) currently allows for a total of 2Ø DIO devices in the DIO.PTP section of the DIO control tables.

Notice that adding pulse-to-pulse devices, or extending the pulse palette, increases the time that the SPC loop takes to execute, and therefore decreases the time available to the other tasks during a given pulse period.

The transmitter attenuator and the receiver bandwidth controllers, although standard DIO devices not normally under pulse-to-pulse control, become part of the DIO.PTP palette extension, if they are configured to be pulse-to-pulse devices by DIOGEN. (If a device is controlled on a pulse-to-pulse basis, its relays must be able to withstand the wear.)

Gaining full command of all of the pulse palette options is not a trivial exercise, nor is it necessary in order to collect useful data. The P2.Ø HF Radar software includes completely operational sounding modes, with default pulse descriptions and sounding sequences, that are based on the combined experience of many workers with a variety of instruments. The defaults also reflect lessons learned from the P1.Ø software design.

If within the structure of the HF Radar a pulse is the basic unit of a sounding, then the first level of pulse organization is the pulse set, established in P1.Ø and before to be a collection of 4 pulses based on a given selected frequency in a manner that allows computation of group delay, angle of arrival, etc. This standard set of 4 pulses, producing 8 sets of echoes because of the dual-receiver system, is the default for most sounding modes under SOUNDER 2.Ø2. Special configurations of the PICKER task allow for 5 or 6 pulses/set (Section 3.1).

The transmission sequence of the pulses in a set is specified as a series of pulse description numbers, or palette indexes. (See command PSEQ, Section 2.7.4.) To specify a pulse set of 4 pulses, any sequence of 4 numbers, in the range 1-8, is valid. The default sequence, which produces the sequence in Table 2.1, is, 1,2,3,4. Equally valid are sequences of the form, 4(1) and 2(1,3), which are respectively equivalent to 1,1,1,1 and 1,3,1,3. The sequence, 2(6),2(8) is equivalent to the 4-pulse set, 6,6,8,8.

Although the default for most sounding modes is 4 pulses/set, SOUNDER can generate pulse sets containing from 1 to nearly 8ØØØ pulses. For example, Pulse Description (PD) 5 could be set up to transmit a full power "heater" pulse, PD 7 could be configured to selectively receive X-polarized echoes, and PD 2 could listen selectively for O. The pulse set sequence, 5,5Ø(7,2) would heat the ionosphere, and then for 1ØØ pulse periods, measure the dissipation of the heater energy in X and O. Such a pulse set would contain 1Ø1 pulses. In order to make use of such a scheme, the user would either have to rewrite some or all of the down-stream tasks, or tape record the sounding data directly from the FEP with a Sounding-mode CALL task, such as TENUS (Section 1Ø.3). The data could then be analysed off-line.

The second level of pulse organization is the sequencing of pulse sets and involves base frequency selection. P1.Ø provided two modes of frequency selection, Ionogram (I-mode) and Kinesonde

(K-mode). I-mode swept from a starting frequency up to an ending frequency, stepping the frequency logarithmically and transmitting a pulse set at each frequency along the sweep. K-mode allowed the user to put 1-1Ø discreet frequencies in a table and to transmit a pulse set at each of these frequencies, round-robin, a specified number of times.

SOUNDER 2.Ø2 has generalized these two frequency selection modes to classes of sounding modes. I-mode becomes the prototype Sweep-class (S-class) mode, and K-mode is the prototype Table-class (T-class) mode. (An additional W-class, resembling T-class for the most part, is described under MAKEM, Section 1Ø.2.1.) Furthermore, the restriction of sounding by frequency index, which is linear in the log of the frequency, has been removed. (See Table 8.1, under command EFRQ.) All frequencies can be expressed and manipulated in 1/1Øths of kHz, this being the control resolution of the Ailtech frequency synthesizer. The SOUNDER 2.Ø2 design allows for the addition of other frequency selection, or sounding, classes. Likely candidates are a pseudo-random class (R-class) and a mixed class (M-class), which would interleave S-class and T-class in various ways to provide detailed information at selected frequencies while giving the overall perspective of an ionogram, with little loss of time resolution. Also under consideration are an extended Table-class (X-class?), which would enlarge the table of 1Ø frequencies for multi-frequency riometry studies, and a double-pulsed, or pulse-pair processing class for Doppler studies.

Each frequency selected by the above classes of sounding sequences is called a base frequency, as noted above, around which a pulse set is formed. In addition, SOUNDER 2.Ø2 provides a third level of pulse organization called ramp frequency selection. The ramp concept is best understood in terms of its evolution. The classic ionogram provides a certain view of the ionosphere, but this view is over a period of time determined by the pulse repetition frequency and the number of pulses transmitted at each frequency. Given a quiet or slowly changing ionosphere, an ionogram is adequate for some analyses, but for other studies, better time resolution may be required. K-mode is a partial solution to this problem, but at the sacrifice of the broader picture given by a frequency sweep. A better solution, originated by J. W. Wright of CIRES, is to do a series of mini-sweeps at each nominal, or base, frequency. SOUNDER's RAMP command (Section 2.7.7) provides a mechanism for defining the dimensions of such a mini-sweep, or ramp, in terms of the number of steps in the ramp, the size of the ramp step, and the number of times the ramp is to be repeated at each base frequency. The ramp step may be expressed either as an increment in frequency index or as a linear frequency. J. W. Wright's original frequency selection scheme as described above was called Basic, or B-mode. As incorporated into the HF Radar, the ramp may be applied to any base frequency of an S- or T-class sounding. (Ramps are not allowed for W-class soundings for reasons detailed in Section 10.2.1.) For this reason, the ramp described above has become called the B-ramp. Notice that the first B-ramp frequency is equivalent to the base frequency for that ramp. Further studies by Wright and M. L. V. Pittaway, of Brunel University, have produced an enhancement of the B-ramp concept that provides the same time and resolution benefits of the B-ramp, but in a fraction of the time and the use of the HF spectrum. This is called the Z-ramp, because if the frequencies are plotted vs. time, a zig-zag figure results. (A detailed description of how the Z-ramp is derived can be seen in Section 2.7.1. See also Section 10.2.1.) Just as a pulse set is transmitted at each base frequency, so an identical pulse set is transmitted at each ramp frequency. I-mode is equivalent to B-mode or Z-mode but with a null ramp specified.

To summarize, a sounding sequence, or the series of pulses transmitted during a sounding, is established as follows.

1. The 8 pulses in the pulse palette are defined with commands TSGC, DFAC, ANT, and TXPH. (TXPH is not fully implemented.) See Chapter 8 for brief descriptions of these commands, Section 2.7.6 for details. Specifying a DIO device to be controlled pulse-to-pulse extends the descriptions in the pulse palette. (See DIOGEN, Section 10.2.4.)

2. The sequence of pulses in a pulse set and hence the pulse set size (normally 4) are specified by command PSEQ. The unextended pulse palette is displayed with command PTP. Command UDIO d displays the palette extension for DIO device, d.

3. The base frequency selection is by sounding class. Command MGET I selects I-mode, the prototype Sweep-class sounding mode; command MGET K selects K-mode, the prototype Table-class sounding mode. A pulse set is transmitted at each base frequency. Sounding sequence specification usually begins with MGET.

4. The RAMP command is used to specify the dimensions of either a B- or a Z-ramp. If specified, a ramp is transmitted at each base frequency of an S- or T-class sounding. A pulse set is transmitted at each ramp frequency, as well as at each base frequency. (The first frequency of a ramp sequence is equivalent to the base frequency of that group of ramps.)

The great flexibility in defining sounding modes described above has obvious benefits, but also creates a significant bookkeeping problem for the experimenter. The number of console options available has greatly increased over P1.Ø, and the definition of a sounding mode is more complex. To salvage the benefits without complicating routine operations, SOUNDER allows the user to save a sounding configuration once it has been constructed and tested. A new sounding mode is given a 2-character name and written to a disc-resident save area. The mode thus created can be recalled, run, modified, or it can provide the nucleus for other hybrid modes. SOUNDER commands that affect the sounding configuration are thereafter automatically retained, and need not be repeated unless something needs to be changed. SOUNDER maintains a directory of all modes thus saved on disc. The first 1Ø modes in the directory are more or less permanent and are created by MAKEM, Section 1Ø.2.1. All user-defined modes are variants of these permanent modes. There can be a total of 43 user modes in a given mode file, and other mode files can be created. The above system can be seen as a mode management system, with a specialized mode editor. (See commands MDIR, MSAV, MGET, MDEL, and RUN, Chapter 8.)

What is actually saved on disc is the core image of the SCT for a given sounding mode. (See SCT, Section 7.1.) These SCT images are recalled from the disc by the commands MGET and RUN, as needed. The core image of the SCT for only one sounding mode is in memory at any given time. This mode is the <u>active</u> sounding mode. The memory address of the active, or <u>current,</u> SCT is available to the other sounding tasks, including ANALYSER, via ET.ASCT (Section 7.3). Also the address of the DIO.PTP palette extension is made global via ET.ADIO. The implication is that an ANALYSER task (yet to be written) can dynamically modify the SCT and the DIO.PTP tables, effecting a feed-back loop, wherein the sounding data influences the collection of subsequent sounding data. This feed-back loop has yet to be closed.

A few words need to be said about design objective 5 (above), that of doubling the HF Radar Pulse Repetition Frequency (PRF), but it is first of all necessary to know how to ascertain whether or not a given PRF is being maintained.

If the down-stream tasks (PICKER, MANAGER, GRAPHER, and ANALYSER) cannot keep up with the data flow through the pipeline, two things can result in SOUNDER. First, at the beginning of each pulse set (from SOUNDER's view of the pipeline), SOUNDER counts the number of free PCT/EKO buffers. If this number is at least 2 less than the number of pulses/pulse set, or a maximum of 6 (NEKOBFS), SOUNDER starts the pulse set. In the normal case, SOUNDER proceeds with the 4-pulse set even if only 2 buffers are free, assuming that PICKER will release 2 more buffers by the time they are needed. If not enough buffers are free to begin the pulse set, SOUNDER sends the TSG a NOP start, which marks time but does not result in any data. This allows time for the data from one pulse to exit the pipeline. After the NOP cycle, SOUNDER again counts the free buffers. (See NXTPLSE, Section 2.9.1.1.) For each time that SOUNDER issues a NOP start to the TSG, MD.PLST in the SCT (Section 7.1) is incremented, counting a lost pulse. A lost pulse affects the time-resolution of the sounding, but does not result in any lost data.

Second, at buffer allocation time, GETBFR (Section 2.9.1.1) makes a final check to see if PICKER has indeed released the buffer needed for the data in the FEP output memory, now awaiting transfer to one of SOUNDER's PCT/EKO buffers. If not, the echo data awaiting transfer are lost, and MD.ELST is incremented. Non-zero values of MD.PLST and MD.ELST are logged at sounding end as "LOST PULSES" and "LOST ECHOES", respectively. These values should be examined by off-line analysis programs. Lost echoes typically result in scrambled pulse sets, although GRAPHER's plot for a sounding that has lost echoes may look deceptively good.

The above paragraphs describe what happens if the down-stream tasks cannot keep up with the flow of data in the pipeline. It is far less obvious, however, when SOUNDER's SPC loop takes too long to execute. The TSG internal clock ticks inexorably, and the TSG executes its program when it senses the first interrupt from its internal clock, after receiving a start from SOUNDER. Therefore, for each cycle through the SPC loop, SOUNDER must send a start to the TSG before the TSG clock ticks. If it does not, a TSG clock interrupt is lost, and SOUNDER's CLOCK subroutine counts one time interval for two that have elapsed. No explicit test is made in SOUNDER or in the TSG for such a time slip. Lost TSG clock interrupts can be detected, however, by initiating a series of back-to-back soundings (SPH 999), then after several tens of soundings, observing if the next sounding start time, as displayed by the TSG panel clock, matches the time displayed by

GRAPHER on the system CRT. The idea here is to allow enough lost TSG clock interrupts to accumulate to be detectable by eye. If time is slipping, jitter in the TX KEY signal can also be observed on an oscilloscope.

Limited testing with the SEL/NSF Interdata 8/16 system at Boot Lake, CO, and with the 7/16 system at the Los Alamos National Laboratory, NM, suggests that a PRF of 1ØØ is attainable for all but P-, G-, and W-modes, if some precautions are taken. First of all, the volume of echo data can be minimized by using KEEP 8, using a reasonably high FEP data threshold (FEP 3 is normal), and setting EPF to a minimum acceptable level, possibly to eliminate multi-hop echoes. (This can be accomplished for T-class soundings with the RNGE command.) The time required to execute SOUNDER's SPC loop can be minimized by keeping the number of pulse palette extensions in DIO.PTP as small as possible. (For example, the bandwidth controls would rarely need to be commanded pulse-to-pulse.) Finally, for unattended operation or for situations in which the system display is less important than a PRF of 1ØØ, command NOQUEUE 2 effectively idles the GRAPHER task. Command [PE terminates the system CRT refresh, saving additional time. (The data can still be reviewed with [PS and GPION.) If all else fails, NFIL 1 reduces the PRF to 8Ø (with the 1Ø ms TSG pace selected), and the system will easily keep up with the data, in all but the most extreme cases.

The above paragraphs have dealt with the design of SOUNDER 2.Ø2. Here we will be dealing with the organization of the SOUNDER task.

SOUNDER 2.Ø2 is divided into 31 modules, which are linked together to constitute the SOUNDER task. Each module resides in its own file on disc. These 31 modules are organized into five groups. The first group contains all of the memory resident parts of SOUNDER (Section 2.6); the remaining four groups each contain the modules for one of SOUNDER's four overlay areas. Overlay areas Ø through 3 are discussed in Sections 2.7 through 2.1Ø, respectively.

The disc file name of a module is included in its subsection title in Sections 2.6 through 2.1Ø. The disc file names are of the form, SNDRxxxx; SNDRxxxx.CAL is the module source, and SNDRxxxx.OBJ is the output from the CAL assembler. Furthermore, SOUNDER's modules are loaded in the order in which they are discussed in Sections 2.6 through 2.1Ø.

2.5. SOUNDER Design and Organization

In the following sections, an individual overlay will be referred to by its area and its position in SYS1:SOUNDER.OVL, SOUNDER's overlay disc file. For example, 1-Ø7 refers to the seventh overlay for overlay area 1 in SOUNDER.OVL. SOUNDER.OVL is discussed in detail in Section 2.6.1.

Routines in memory resident SOUNDER can load overlays into any of the four overlay areas; a routine in a given overlay area can load overlays into any of the other areas, but never into its own area. Overlaying is suspended during sounding in order that MANAGER have full control over the disc. Overlay areas Ø, 1, and 3 are each 512 bytes (2 disc sectors); overlay area 2 is 256Ø bytes in length (1Ø sectors). The sizes of the overlay areas reflect compromises between memory utilization and speed of execution.

SOUNDER executes in one of two main loops after its initialization routine has executed at task start up. The Idle-Mode Advice (IMA) loop is entered after SOUNDER initialization, and SOUNDER remains in the IMA loop, periodically checking for advice (console commands directed at SOUNDER) until a sounding is initiated. Just before the sounding begins, overlays Ø-ØØ, 1-ØØ, 2-ØØ, and 3-ØØ are loaded, and SOUNDER enters the Sounding Pipeline Control (SPC) loop. SOUNDER executes within the SPC loop for a single sounding, or if an SPH schedule is active. until AB is typed or SN is satisfied. (See Chapter 8 for brief descriptions of SOUNDER's commands.) When the SPC loop, which is overlayed except for a few subroutines, is exited, the IMA loop, which is itself memory resident, is reentered. The IMA loop loads overlays as needed to respond to advice from the console.

The IMA loop is deceptively small, requiring only a few lines of code in SNDRADVC (Section 2.6.1). It provides access, however, to all of SOUNDER's advice routines and to the Idle-mode CALL tasks (Section 1Ø.2). The distinction between IMA and SPC function remains intact when Sounding-mode CALL tasks are run (Section 1Ø.3), even though the SPC loop is executed without the involvement of the "down-stream" tasks. Command TPR (tape replay) simulates SPC action by providing a flow of data without executing the SPC loop to acquire the data.

SOUNDER 2.Ø2 recognizes 57 commands which are briefly described in Chapter 8. For a detailed explanation of SOUNDER's commands, refer to Table 2.3, below. The module, or disc file, listed with each command contains the main coding for the command; the section listed explains the main code, referring to set up and subroutines in other sections when appropriate.

COMMAND	MODULE	SECTION
AB	SNDRADVC	2.6.1
AGC	SNDRO008	2.7.9
ANT	SNDRO005	2.7.6
BDW	SNDRO009	2.7.10
BOOT	SNDRBOOT	2.9.2.1
CALA	SNDRO006	2.7.7
CALL	SNDRO007	2.7.8
CDN	SNDRO006	2.7.7
CONT	SNDRO001	2.7.2
DELF	SNDRO006	2.7.7
DESC	SNDRO007	2.7.8
DFAC	SNDRO005	2.7.6
DIO	SNDRO003	2.7.4
DLOF	SNDRSTUP	2.9.2.2
DLON	SNDRSTUP	2.9.2.2
EFRQ	SNDRO006	2.7.7
EPF	SNDRO006	2.7.7
F	SNDRO004	2.7.5
FEP	SNDRBOOT	2.9.2.1
HPAT	SNDRSTUP	2.9.2.2
HPE	SNDRO006	2.7.7
HVOF	SNDRSTUP	2.9.2.2
HVON	SNDRSTUP	2.9.2.2
IFAT	SNDRADVC	2.6.1
KEEP	SNDRO006	2.7.7
LPAT	SNDRSTUP	2.9.2.2
MDEL	SNDRO001	2.7.2
MDIR	SNDRO007	2.7.8
MGET	SNDRO001	2.7.2
MSAV	SNDRO001	2.7.2
NFIL	SNDRO006	2.7.7
NO	SNDRO007	2.7.8
NSET	SNDRO006	2.7.7
PIK	SNDRO103	2.8.4
PKHI	SNDRO008	2.7.9
PKLO	SNDRO008	2.7.9
PSEQ	SNDRO003	2.7.4
PTP	SNDRO008	2.7.9
RAMP	SNDRO006	2.7.7
RFAT	SNDRADVC	2.6.1
RNGE	SNDRO004	2.7.5
RUN	SNDRO001	2.7.2
RXANT	SNDRO005	2.7.6

2.5. SOUNDER Design and Organization

COMMAND	MODULE	SECTION
SFRQ	SNDRO006	2.7.7
SN	SNDRO007	2.7.8
SPH	SNDRO007	2.7.8
STEP	SNDRO006	2.7.7
SYS	SNDRSYST	2.9.3.1
TICK	SNDRBOOT	2.9.2.1
TPR	SNDRO002	2.7.3
TSGC	SNDRO005	2.7.6
TXANT	SNDRO003	2.7.4
TXOF	SNDRADVC	2.6.1
TXON	SNDRSTUP	2.9.2.2
TXPH	SNDRO005	2.7.6
UDIO	SNDRO003	2.7.4
VUF	SNDRO004	2.7.5

Table 2.3. SOUNDER COMMANDS.

In the descriptions that follow, the routines or tables under discussion are delineated with source code sequence numbers. The sequence numbers may not be accurate as the source modules become modified; they should, however, make it possible to locate the source code under discussion.

The narrative in Sections 2.6 through 2.10 is thus intended to accompany the listings of the SOUNDER task. The stress is on function and interconnections rather than method and technique, except when technical detail is essential to understanding function. The descriptions are heavily cross referenced. The references may be ignored during a quick reading to gain overall perspective, or they may be followed to obtain more details about the topic under discussion.

2.6. Memory Resident Modules [DCW]

The memory resident modules contain routines, tables, data buffers, etc., which are permanently in memory after SOUNDER has been loaded and started. Overlayed modules, on the other hand, are disc resident (in SOUNDER.OVL) and are loaded into memory as they are needed. Memory resident subroutines, tables, etc., can be used at any time by routines that are themselves memory resident, or by routines that are overlayed. An effort has been made to overlay all except those parts of SOUNDER which must be available at all times to IMA and SPC routines. There are a few exceptions in cases where an overlay area was too small to accomodate everything needed.

2.6.1. SOUNDER Advice (SNDRADVC) [DCW]

In a sense, SNDRADVC is the main module of the SOUNDER task. It contains the Idle-Mode Advice (IMA) loop, the ADVICE subroutine, and all advice routines that must be memory resident. SNDRADVC also contains the overlay loader, and the routine that does final initializing before a sounding begins.

GETADV (SADØ55ØØ-SADØ93ØØ)

GETADV, the IMA loop initialization routine, is entered from SNDRINIT (Section 2.6.5) after SOUNDER task start up, and from SNDRPIPE (Section 2.9.1.1) and SNDROØØ2 (Section 2.7.3) after a sounding (RUN) or a tape replay (TPR). GETADV updates the run count and the date of the last run in case a sounding mode has been run since the last time the IMA loop was entered. (See SNOLD and TODAY, Section 2.6.4.) In case the SOUNDER command, NO [A,E,T] (Section 2.7.8), was given before the previous sounding, GETADV cancels the effect of the command. The House-keeping ADC (HKADC) is not revitalized unless it is configured into the system by DIOGEN (Section 1Ø.2.4).

IMALOOP (SADØ95ØØ-SAD112ØØ)

IMALOOP is the start of one of SOUNDER's two main loops, the Idle-Mode Advice loop. IMALOOP loads the IMA command overlay 3-Ø1 (Section 2.1Ø.2), if it is not already in overlay area 3; reads the HKADC, if it is configured into the system and if a CALL task is not active; and calls subroutine ADVICE, below. (IMALOOP does not read the HKADC if a CALL task is active to avoid conflicts on the DIO bus.) If command RUN, CONT, or TPR has been given, ADVICE returns from SNDOV (below) with GO = Ø, and control goes to SINIT and the SPC loop (Section 2.9.1.1), where the countdown for sounding begins. (GO and the other pipeline registers are explained in Section 2.9.1.1.) If no sounding has been initialized, IMALOOP delays 5Ø ms and loops again.

ADVICE (SAD114ØØ-SAD158ØØ)

Subroutine ADVICE is called from both the IMA and SPC loops to check SOUNDER's task queue for commands from the console or for messages from other tasks; if the queue is empty, ADVICE simply returns to the calling routine. If the queue is not empty, ADVICE saves R4 (GO) through R15 and checks the alphabetic code of the sender of the Q-block. ADVICE executes at A.ADV, currently a null routine, if the Q-block is from

ANALYSER, at C.ADV if from COUNSEL, or at T.ADV if the Q-block is from a CALL task. T.ADV loads overlay 1-Ø3 and goes to ANSCALL in 1-Ø3, where the CALL task Q-block is decoded (Section 2.8.4). C.ADV, if the Q-block is CØ4 from the CANHF command, turns off the transmitter filaments and HV, and writes the current sounding mode SCT image to disc. SOUNDER then terminates itself with SVC3, code Ø.

If the Q-block is CØ2, indicating a console or scheduled command directed to SOUNDER, C.ADV jumps to the command routine directly, if it is memory resident. Otherwise, the command routine overlay must first be loaded. Only a single overlay is loaded at this time; If a command routine needs other overlays, it loads them itself. The single overlay loaded by C.ADV is specified in the advice tables as explained in Section 2.1Ø.2.

All command, or ADVICE routines (except one) exit the ADVICE subroutine through BADARG, which sets the CØ2 Q-block status to 1; through OKADV, which clears the CØ2 status to Ø; or through CLADV, which sets the status to the ERR code in RØ. (The single exception to the above is the routine for the RUN <task> command that initiates execution of an Idle-mode CALL task (Section 1Ø.2). In this case, clearing the CØ2 Q-block status is deferred so that the CALL task can have temporary control of the console.)

After clearing the CØ2 Q-block status in one of the above three ways, ADVICE recovers R4-R15 and returns to the calling routine.

Get Overlays (SAD16ØØØ-SAD228ØØ)

The overlay loader is called at entries GETOVØ through GETOV3 to load overlays into overlay areas Ø through 3, respectively. In all cases, the overlay number is in R14. The overlay number is specified to the task establisher (TET) and is that overlay's position, counting from Ø, within the overlay disc file, SYS1:SOUNDER.OVL, for a given overlay area. That is, SOUNDER.OVL contains overlays Ø-ØØ through Ø-Ø9, followed by 1-ØØ through 1-Ø5, followed by 2-ØØ through 2-Ø2, followed by 3-ØØ through 3-Ø2. SOUNDER.OVL is a 94-sector, contiguous disc file, allocated by TETSNDR.CSS if it doesn't already exist. The first sector of each overlay contains information used by the MT2 OS SVC5 call to identify the overlay. Because a direct random disc read with SVC1 is twice as fast as an SVC5 call, SOUNDER's overlay loading subroutine uses the former, computing an overlay's sector address within SOUNDER.OVL using parameters defined at the start of the SNDRADVC module (SADØ18ØØ-SADØ29ØØ). If overlays are added to or deleted from SOUNDER, these parameters must be modified, TETSNDR.CSS (Section 9.4) must be modified, and the size of SOUNDER.OVL must be changed. Because of the overlay identifier sector, 3 sectors must be reserved for a 2-sector overlay, 11 sectors for a 1Ø-sector overlay. Because of the way TET writes to an overlay file, SOUNDER.OVL must be at least 3 sectors longer than the number of overlays in it would indicate.

The overlay loader keeps track of what overlay is currently in each overlay area; an overlay request is ignored if the specified overlay is already loaded. The ENTRY, NOV, is put into SOUNDER's load map (SOUNDER.MAP) to permit the operator to check SOUNDER's current overlays using COUNSEL's LINE command (Section 6.4). The four 1/2-words beginning at NOV contain the area and overlay number of the current overlays for areas Ø-3, respectively. For example, a value of Ø1Ø3 means overlay 1-Ø3.

CALLOV is a small routine that allows CALL tasks to request SOUNDER to load overlays that the CALL task needs. This feature is currently not used. The mechanism, however, is described under ANSCALL, Section 2.8.4.

<u>BOOT</u> (SAD23ØØØ-SAD26ØØØ)

The BOOT initialization routine first scans the command argument for "TSG", "FEP", or "DIO". The presence of the first character of a device name is enough to satisfy the scan. If the device name is valid, and the character following the name is a carriage return, the command, BOOT <device>,TWO is assumed; TSGMEM.TWO, FEPMEM.TWO, and DIO.TWO are the P2.Ø defaults. If a file extension is specified, it is used instead of "TWO". After scanning the argument, BOOT loads overlay 2-Ø1 and executes subroutine BTTSG, LDFEP, or GTDIO, depending upon which device is being booted. These subroutines are all in module SNDRBOOT (Section 2.9.2.1). The file extension specified by the BOOT command, or "TWO" by default, is saved in MD.TSGX, MD.FEPX, or MD.DIOX for the current sounding mode. Thus once given for a specific sounding mode, the BOOT command specification is retained; when the sounding mode in question is RUN, SOUNDER automatically sets up the TSG, FEP, and DIO as specified.

GETMAP (SAD26200-SAD27900)

GETMAP is called with the required protected frequencies map identifier in R0. This is the single alphabetic character from MD.MAP (Sections 7.1 and 10.2.1). If the required map is already in memory, GETMAP executes subroutine CKMAP (Section 2.6.2), which verifies that the map in memory has not suffered from inadvertant or purposeful modification. If the required map is not in memory, GETMAP uses GET201 to load overlay 2-01, then executes subroutine GTMAP (Section 2.9.2.2). (GETMAP and BOOT share routine GET201.)

RFAT and IFAT (SAD28000-SAD33800)

Commands RFAT and IFAT are used to set the receiver RF and IF slow attenuators during a sounding from the console, provided the AGC command has been used to put the receiver channel(s) under manual control. (See Section 2.7.9.) These commands are expected to be used mostly with T- and W-class sounding modes, but are accepted for S-class modes as well. The forms of the RFAT and IFAT commands are the same for all classes, except that the frequency number must be specified for a T- or W-class mode, because each table frequency is attenuated independently. If the frequency number specified is not 0-9, "ERR 1" results. No check is made to see if a given frequency has been activated by the F command (Section 2.7.5).

RFAT and IFAT accept one or two attenuations (in dB) for any sounding mode class. If two attenuations are specified and if both channels are under manual control (AGC 0), the first specified controls channel 1, the second controls channel 2. If a single attenuation setting is specified and both channels are manual, both channels are attenuated the same. If a single attenuation is given, and only one channel is manual, only that channel is affected. If either RFAT or IFAT is overspecified, that is, if more attenuations are specified than are manual channels, the extra specifications are ignored. In all cases, the attenuations specified are truncated to multiples of 4, in the range 0-60 dB; for example, 18 becomes 16, 72 becomes 60. No cross checking is done between RFAT and IFAT; the operator must make reasonable settings. Error messages are kept at minimum in order that timing resolution during sounding not be disturbed. Notice that during sounding, only the short forms of these two commands, RF and IF, are accepted.

The attenuations for a T- or W-class mode can be set before sounding begins. The attenuations for S-class modes, however, are initialized at sounding start from the base RF and IF (MD.BAG) specified by command AGC, so presounding RFAT and IFAT commands are ignored for S-class modes.

ABORT (SAD34000-SAD36600)

The ABORT routine is entered whenever the AB command is typed. ABORT first checks the flag, MODE. If MODE < 0, a TPR is being aborted; ABORT clears MODE to 0 to notify the TPR routine to abort. (See Section 2.7.3.) If a TPR is not being aborted, ABORT checks to see if a soundings/hour (SPH) schedule is active. If so, and whether sounding or marking time until the next sounding, ABORT saves the current mode ID and the time of the next scheduled sounding. This information, saved in NEXT (Section 2.6.4), is used by the CONT command to restart the current mode on schedule. (See Section 2.7.2.) Notice that after an SPH schedule is ABorted, any sounding mode can be RUN; CONT reinstates the mode for which SPH was specified and runs it on schedule.

Whether or not an SPH series is in progress, ABORT checks ET.CDN (Section 7.3) to see if a countdown-to-sounding is in progress, and if not, checks register PULSE (Section 2.9.1.1) to see if a sounding is active. If neither of the above is true, SOUNDER is either idle or between soundings, and the next action control word, NXTACT, is set to GETADV (above); the IMA loop is executed next. If the 1-19 second countdown or a sounding is in progress, flags are cleared that force the sounding to end just before the next pulse set begins. The pipeline is emptied, if it contains any data, before the S06 Q-block is sent to PICKER, signaling sounding end. (See Section 2.7.1.)

TXOFF (SAD36700-SAD38500)

TXOFF calls subroutine OFFTX to turn off the transmitter filaments and high voltage, and to select the dummy antenna load whenever TXOF (IMA) or TX (SPC) is typed. The command takes effect immediately. The ADVICE subroutine also calls OFFTX when a C04 Q-block (CANHF) is sensed.

SNDOV (SAD38700-SAD46600)

Routine SNDOV is entered from SETUP in overlay 2-02 (Section 2.9.2.2) to perform final initialization before sounding for the RUN or CONT command. SNDOV loads all of the sounding, or SPC, overlays (Ø-ØØ, 1-ØØ, 2-ØØ, and 3-ØØ), reads the TSG clock with RDTIME, now in overlay 1-ØØ (Section 2.8.1), and if RUN <time> was specified, determines if the operator has allowed enough time for the sounding countdown. If the CONT command was given, SNDOV gets the time of the next sounding from NEXT and compares this with the current time. If the SPH sounding cannot be CONTinued on schedule, it is run at the next scheduled time instead. Register GO is now cleared to Ø to signal IMALOOP (above) that control is to transfer from the IMA to the SPC loop, and the next action control word, NXTACT is set to NXTSND in module SNDRPIPE (Section 2.9.1.1). SNDOV exits the ADVICE subroutine through OKADV.

Numerical Command Input (SAD472ØØ-SAD5Ø7ØØ)

Most SOUNDER commands require numerical input. Subroutines CONVD and CONVH convert decimal and hexadecimal ASCII arguments, respectively, to binary. They call SVC2, code 15, with function codes X'CØ' and X'4Ø', respectively, both of which ignore leading blanks. CONVD and CONVH expect the address of the CØ2 Q-block, gotten from SOUNDER's task queue by ADVICE (above), to be in R2; the converted number is returned in RØ. The first non-numerical character encountered terminates the conversion, and R1, set to the contents of CØ2 + 6 before the SVC2 conversion call, is updated by SVC2 to point to the terminating character. SVC2 sets the condition code (CC) based on the results of the conversion. The CC is tested at CONCOM; if no conversion takes place, or if the number is too large, the BADARG exit is taken from ADVICE, resulting in "ERR 1".

CONVD and CONVH can be entered at CONVD1 and CONVH1. These entries expect R1 to be pointing to the character that terminated a previous conversion, and increment R1 by 1 to skip the terminator, or argument separator (comma, blank, etc.). Thus, CONVD1 and CONVH1 are used to convert a list of arguments after CONVD or CONVH have converted the first argument of a list, thereby initializing R1.

Because of the potential exit through BADARG, the above four subroutines are called only from command, or ADVICE, routines.

2.6.2. Non-reentrant Subroutines (SNDRMSUB) [DCW]

Module SNDRMSUB contains a collection of memory resident subroutines, none of which are reentrant; SNDRMSUB subroutines are called only by other SOUNDER routines.

ERROR and ERRCO (SMSØ25ØØ-SMSØ55ØØ)

ERROR and ERRCO are SOUNDER's error logging routines. They are identical, except that ERROR, after logging an error message through COUNSEL, pauses and waits for operator intervention. After an error message is logged by ERRCO, execution of the calling routine continues.

Both ERROR and ERRCO expect the error status or code in RØ. The error code and the location of the error call (R15 - 4) are written in ASCII to the error message, and subroutine WRLOG (below) is called to log the error message through COUNSEL. ERROR uses a message weight of 6, ERRCO uses 5. (See Section 6.2 for COUNSEL error logging.)

The most common errors seen are of the type, 82dd, indicating that device dd has timed out. The TSG start (8237), TSG page write (8224), TSG page read (8225), TSG clock (8226), HKADC (8234), and the EKO interface (82F1) may all occasionally malfunction and cause an error. A TSG timeout (8237) may be caused by either a TSG or a FEP problem. (See PIPLN, Section 1Ø.2.15.) An error of ØØØ7 during HF Radar start up is most likely caused by files left open on SYS1 from a previous crash. In such a case DISCHECK (DSC2.CSS) can be run to close the files. An error of ØØØ4 indicates that a file has not been found on SYS1. A missing file can either be allocated or copied down from SYSC, whichever is appropriate.

DEVERR (SMSØ57ØØ-SMSØ77ØØ)

DEVERR is called to flag and to count device errors for the TSG, FEP, EKO, DIO, TSG Clock, and for the House-keeping ADC. The flags are set in MD.STAT (Section 7.1); the error counters are those displayed by the SYS command (Section 2.9.3.1). The address of the error counter must immediately follow the DEVERR call. The error counters are in module SNDRTBLS (Section 2.6.4) and are in the same order as are the flag bits in MD.STAT. MD.STAT can be examined by offline software to determine if any device errors were sensed during a sounding. The error counters are zeroed at SOUNDER start up, and are maintained until CANHF is typed. No FEP errors will ever be sensed. An eleventh-hour decision was made to withdraw explicit SOUNDER control over the FEP in order to save time in the SPC loop. See PIPLN, Section 1Ø.2.15 for details.

RESFEP (SMSØ78ØØ-SMSØ94ØØ)

RESFEP issues a STOP command to the EKO SELCH, then resets the FEP program location counter and clears the FEP DATA READY signal (Section 2.2). This subroutine is called after a RUN or CONT command is given and when any FEP program location is loaded. (See commands FEP and BOOT FEP, Section 2.9.2.1.) This FEP reset is identical to the reset used by CALL task FEPEKO (Section 1Ø.2.16) and by the FEP verify routines.

WRLOG (SMSØ96ØØ-SMS125ØØ)

WRLOG is called to request the COUNSEL task to log a message. (See Chapter 6 for details about logging.) Two arguments follow the WRLOG call. The first is the 16-bit message address, the second is the 16-bit message weight, or severity (Ø-9). COUNSEL concatenates the message weight with the single character task identifier to form the message identifier, e.g., "S5". WRLOG puts the message weight into byte Ø of the message Q-block and the message address into bytes 2 and 3. SOUNDER's ID, "S", is in byte 1. WRLOG then makes the logging request by sending the SØn Q-block to COUNSEL.

WTPOST (SMS126ØØ-SMS145ØØ)

WTPOST is called by SNDRPIPE in the SPC loop to wait for all post-sounding activity to cease before undertaking any further action. (WTPOST need not be memory resident, but the SPC overlays are very full.) WTPOST waits in a loop, checking the SØ6 Q-block status every 1ØØ milliseconds until it has been cleared by all of the down-stream tasks. After the SØ6 status indicates that all tasks have finished their post-sounding activities, WTPOST checks to see if an SPH schedule is active, and if so, returns to the caller to run the next sounding. If no SPH schedule is active and a Sounding-mode CALL task (Section 1Ø.3) is occupying GRAPHER's memory partition, WTPOST loads overlay 1-Ø3 and goes to ENDCALL to reinstate GRAPHER. WTPOST is called at entry, GTGRAF, when the number of soundings to run, SN, is satisfied, in case a CALL task needs to be swapped out of GRAPHER's partition.

RDHSK (SMS14700-SMS18300)

RDHSK is called to read the House-keeping ADC from both the IMA and the SPC loops. If the NO A command has been given, RDHSK stores -1's (FFFF hex) for the HKADC data. Otherwise, the HKADC channel indicated by R3 (0-63) is read, and the data is stored in the address in R13. The address in R13 will point either to ET.HKADC (Section 7.3) or to the HKDIO buffer (Section 2.8.1). If the HKADC times out, DEVERR (above) is called to flag the error in MD.STAT and to increment the HKADC system error counter. ERRCO (also above) will log error 8234 through COUNSEL, and RDHSK will flag the House-keeping ADC as dead.

ROULM and RWULM (SMS18500-SMS20900)

ROULM and RWULM are called to set the DIO interface to the read only or read/write state, respectively. ROULM and RWULM set the DIO interface, or Universal Logic Module (ULM), to the desired state, then read back the state just set. If the ULM is not in the correct state, DEVERR (above) is called to flag the error in MD.STAT and to increment the DIO error counter. The same MD.STAT flag and error counter are affected if a DIO data setting fails to verify. See WRDIO, Section 2.9.1.3.

SETDST (SMS21000-SMS23800)

SETDST is used to establish DIO data for an entire sounding and to set pulse-to-pulse data during a sounding. SETDST is used to set data for dedicated DIO ports only. SETDST is called with a DIO port address in R5 and the data in R8. If the DIO port does not exist, that is, if it is not in the current DIO configuration, SETDST returns to the caller immediately with R0 = 0. Otherwise, SETDST finds the entry for the port in the DIO Sounding Table (DST, Section 7.9). If the port expects no data pulse-to-pulse (DST.PMSK = 0), the data in R8 are logically ANDed with the data direction mask (DST.DMSK) and put into DST.PDTA. SNDRPIPE (Section 2.9.1.1) uses DST.PDTA to set the port at sounding start. If the port does expect data pulse-to-pulse, the data in R8 are ANDed with DST.PMSK (result A). The data currently in DST.PDTA are ANDed with the inverted DST.PMSK, and this result is ORed with result A. This final combination of pulse-to-pulse and sounding data, if any, is put in DST.PDTA for use by SNDRPIPE to set up the DIO for the next pulse. For details about DIO configuration set up, see DIOGEN, Section 10.2.4.

2.6.2. Non-reentrant Subroutines (SNDRMSUB)

MAPFRQ (SMS24ØØØ-SMS27ØØØ)

MAPFRQ is called by IMA and SPC routines to check if a 32-bit frequency (in R12,R13) is unprotected and within the range of the HF Radar hardware. (See PROTF, Section 1Ø.2.3.) MAPFRQ tests the input frequency against the protected bands in FRQMAP (Section 2.6.4), and if the frequency is found to be protected, sets RØ = Ø. If the frequency is out of range, RØ = -1. If the frequency is unprotected and in range, RØ = 1. The condition code (CC) is set from RØ before MAPFRQ returns to the caller.

CKMAP (SMS271ØØ-SMS313ØØ)

CKMAP is called to check the integretity of the protected frequencies map, FRQMAP. The map is written to disc by PROTF (Section 1Ø.2.3) with a checksum byte and a 2-byte value that is the count of all bits set in the map. These check bytes are verified if CKMAP is called with RØ > Ø; they are recomputed and set if RØ = Ø. If FRQMAP becomes altered, CKMAP fails, and the SOUNDER task halts.

Delay Subroutines (SMS315ØØ-SMS35ØØØ)

When SOUNDER needs to kill some time, one of three subroutines is called, unless the delay can be effected with a single instruction like a store multiple (STM).

If the delay is greater than 1Ø milliseconds, great accuracy is not required, and the other tasks should be allowed to execute during SOUNDER's inactivity, DLYMT2 is called. The 16-bit argument following the call is the value of the delay in milliseconds. DLYMT2 calls SVC2, code 23, so DLYMT2 is never called from the SPC loop.

To delay without the uncertainties of the MT2 OS SVC2 call, DLYCPU is used. DLYCPU executes a tightly-timed loop the number of times specified by the 16-bit argument following the call to DLYCPU. The loop time is 5.ØØ microseconds for the 7/16 (HSALU) and 5.25 for the 8/16.

DLYTSG kills time by going through a TSG start-wait cycle. The TSG command byte used must invoke the NOP scenario Ø that does not start the FEP or produce any data. DLYTSG calls CLOCK (Section 2.1Ø.1), so the HF Radar time keeping is maintained. DLYTSG allows other tasks to execute and is called from the

SPC loop. DLYTSG assumes that the TSG is configured into the system and has not timed out.

2.6.3. Reentrant Subroutines (SNDREENT) [JRW]

A number of reentrant subroutines are available to the user. These subroutines keep all data in registers and do not modify memory so that they may be used simultaneously by several tasks. Because these subroutines are expected to be widely used, they are described in detail regarding method and technique.

For tasks running as part of the HF Radar system, the locations of the subroutines are given in the Environment Table. For example, to call NTOFR, assuming the location of ET is in R3:

```
LH    RA,ET.ANTFR(R3)
BALR  RF,RA
```

CALL tasks that need to run stand-alone are TETed with SNDREENT also. (See Section 10.2.) These tasks can call the reentrant subroutines directly with BAL, instead of BALR, provided the subroutine names are listed as EXTRN (external) to the main task.

The subroutines are designed more for speed than accuracy. Typical accuracy is 0.1%, more than enough for plotting and quick look.

NTOFR (SRE01200-SRE06200)

Purpose Frequency index to frequency.

Address ET.ANTFR

Registers R0-R7: unchanged.
(R8,9): used.
(RA,B): 8-digit frequency in BCD, Hz.
(RC,D): frequency/100, Hz.
RE: input frequency index, unchanged.
RF: return address.

Method Divide frequency index by 400 giving quotient L and remainder NI, $0 \leq NI < 400$.

```
NF = ((13*NI + 14788)*NI/8192 + 2225)*NI/8 + 160000
FREQ = NF*2**(L-8)
```

2.6.3. Reentrant Subroutines (SNDREENT)

For log sweeps and ramps, the frequency index is used instead of the frequency. This allows frequency to be specified as 12 bits and the logarithmic scanning is automatic.

Processing the data requires that the user be able to determine the exact frequency used in the sounding. That frequency is called F, and the sounding made at F + delta F differs from F by exactly delta F. There is not an index associated with F + delta F. The index determines F and a constant delta F is added to F.

The computer algorithm to calculate F from frequency index NI is given below in two versions. The algorithm was chosen for interchangeability among Fortrans rather than absolute logarithmic accuracy, although it is quite accurate.

Two Fortran versions of NTOFR follow. The first uses double precision floating point integers (Interdata) and the second is for computers with three or more bytes for integer arithmetic (≥24 bits).

```
      FUNCTION FREQ(NI)
C FREQUENCY INDEX (NI) TO FREQUENCY.
C 9ØØØØØØ. IS MORE THAN HALF THE LARGEST FLOATING POINT
C INTEGER.  9ØØØØØØ. FORCES TRUNCATION OF THE CALCULATIONS
C TO INTEGER.
C ON COMPUTERS THAT ROUND (INTERDATAS WITH HARDWARE DOUBLE
C PRECISION), SUBTRACT Ø.5 FROM THE LARGE ADDITIVE CONSTANTS.
C         JRW 12/Ø8/78.
      L = NI/4ØØ
      N = NI - 4ØØ*L
C BREAK INTO 9 OCTAVES WITH 4ØØ STEPS IN EACH.
      F = 13*N + 14788
      A = N
      F = (A*F/8192. + 9ØØ2225.) - 9ØØØØØØ.
      F = (A*F/8. + 916ØØØØ.) - 9ØØØØØØ.
      IF(L.EQ.8) GO TO 2
      L1 = L + 1
      DO 1 J = L1,8
 1    F = F/2.
 2    FREQ = (F + 9ØØØØØØ.) - 9ØØØØØØ.
      RETURN
      END
```

```
      FUNCTION FREQ(NI)
C FREQUENCY INDEX (NI) TO FREQUENCY FOR COMPUTERS WITH 24 OR
C MORE BITS FOR INTEGER ARITHMETIC.
      L = NI/4ØØ
      N = NI - 4ØØ*L
      M = 13*N + 14788
      M = (N*M)/8192 + 2225
      M = (N*M)/8 + 16ØØØØ
      IF(L.EQ.8) GO TO 2
      L1 = L + 1
      DO 1 J = L1,8
 1    M = M/2
 2    FREQ = M
      RETURN
      END
```

<u>FTOBCD</u> (SREØ35ØØ-SREØ62ØØ)

Purpose Frequency to BCD, contained in NTOFR.

Address ET.AFRTB

Registers RØ-R7: unchanged.
(R8,9): used.
(RA,B): 8-digit frequency in BCD, Hz.
(RC,D): input frequency/1ØØ, Hz.
RE: unused.
RF: return address.

Method Repeatedly divide the frequency by 1Ø and pack the 4-bit remainders into a full word.

<u>FRTON</u> (SREØ64ØØ-SREØ9ØØØ)

Purpose Frequency to frequency index. This is not an exact inverse of NTOFR, but is close enough for frequencies input to SOUNDER and GRAPHER.

Address ET.AFRTN

Registers RØ-R9: unchanged.
RD: frequency index, output.
(RA,B): used.
(RC,D): frequency/1ØØ, changed.
RE: used.
RF: return address.

2.6.3. Reentrant Subroutines (SNDREENT)

Method — The frequency NF is normalized to 5ØØØ (X). The number of shifts subtracted from 9 becomes the octave.

$$X = NF/5000, \; 0.5 \leq X < 1.$$

$$FI = 400*octave + ((491*X - 1652)*X + 2419)*X - 857$$

LOG (SREØ92ØØ-SRE113ØØ)

Purpose — Logarithm base 1Ø (for dB).

Address — ET.ALOGF

Registers —
RØ-R9: unchanged.
(RA,B): used.
RC: log (base 1Ø) * 2Ø48.
RD: input number, changed.
RF: return address.

Method — The number X is normalized. The number of shifts subtracted from 14 becomes the characteristic of log base 2.

$$8192 \leq X < 16384$$

$$LOG2 = 23873*(X - 11585)/(4*(X + 11585)) + 1024 + characteristic*2048$$

$$LOG10 = 0.30103*LOG2$$

PHAMP (SRE115ØØ-SRE18ØØØ)

Purpose — Calculate phase and amplitude from East and North components.

Address — ET.APAMP

Registers —
R8: amplitude, same units as RC and RD.
R9: phase*4Ø96, radians.
(RA,B): used.
RC: East component, changed.
RD: North component, changed.
RE: used.
RF: return address.

Method The angle is reduced to the first octant (0-pi/4). The absolute values of RC and RD are taken and the registers exchanged if necessary so that:

RC ≥ RD ≥ 0

T = tanPHI = RD/RC, 0 ≤ T ≤ 1

PHI = T*16384/(18150*T*T/4 + 16384)

A = RC*(1 + 5*PHI*tanPHI/8 - PHI*PHI/8)

arctan(RD/RC) = PHI adjusted to the proper octant

Given the definitions and conditions above, A = RC*secPHI. Consider the following series expansions that hold for PHI in the first octant:

```
secPHI = 1 + PHI**2/2 + 5*PHI**4/24 + 61*PHI**6/720 +
  277*PHI**8/8064 + 50521*PHI**10/3628800 + ...

PHI*tanPHI = PHI**2 + PHI**4/3 + 2*PHI**6/15 +
  17*PHI**8/315 + 62*PHI**10/2835 + ...
```

Truncating both series at PHI**4 we have:

```
5*PHI*tanPHI/8 = 5*PHI**2/8 + 5*PHI**4/24
```

Subtracting PHI**2/8 from both sides we obtain:

```
5*PHI*tanPHI/8 - PHI**2/8 = secPHI - 1
```

Therefore:

```
secPHI = 1 + (5*PHI*tanPHI - PHI**2)/8 + R
```

where R is an error term in PHI**6 and higher powers. Multiplying by RC and ignoring the error term yields the equation for A above.

Consider the error term:

```
R = secPHI - 5*PHI*tanPHI/8 - 1 + PHI**2/8
```

Substituting the series expressions gives:

```
R = (PHI**6)*(61/720 - 2/24) + (PHI**8)*(1385/40320 -
   17/504) + (PHI**10)*(50521/3628800 - 31/2268) + ...
  = PHI**6/720 + 5*PHI**8/8064 + 307*PHI**10/1209600 + ...
```

The maximum allowable PHI is pi/4 so that:

```
R(pi/4) = RMAX = 0.000326 + 0.000090 + 0.000023 + ...
  = c. 0.00044
```

XORO (SRE18200-SRE21200)

Purpose: Discriminate between eXtraordinary and Ordinary echo returns.

Address: ET.XORO

Registers:
- R0-R6: unchanged.
- R7-R9: used.
- R10,R11: cross product result.
- R12: unchanged.
- R13: address of range of echo record, unchanged.
- R14: unchanged.
- R15: return address.

Method: The polarization angle is computed by finding the cross product between the average of the North and South signal vectors and the average of the East and West signal vectors. If the result of this computation is less than zero, the echo is taken as extraordinary. The condition code (CC) is set from the computation and returned to the caller. The result of the computation is also returned to the caller in the register pair, R10,R11.

XORO assumes that the North and East antennas are connected to channel 1, inputs 1 and 2. The South and West antennas are assumed to be connected to channel 2, inputs 1 and 2. The ANT and DFAC sequences are assumed to be the defaults. (See Table 2.1.)

2.6.4. SOUNDER Tables (SNDRTBLS) [DCW]

Module SNDRTBLS contains the HF Radar global tables SCT, ET, and DIO. The PCTs and the EKO buffers immediately follow SNDRTBLS and are discussed in Section 2.6.5. One additional HF Radar global table, SCAN.TAB, is in COUNSEL. (See Section 6.3.)

SNDRTBLS also contains all of SOUNDER's circular lists (stacks), Q-blocks, SVC parameter blocks, TSG-related tables, and miscellaneous byte and 1/2-word storage. These data are made global to SOUNDER's modules only, not to the other HF Radar tasks.

<u>SØ2LST</u> (STBØ36ØØ-STBØ43ØØ)

SØ2LST is SOUNDER's circular list of SØ2 Q-block addresses and is initialized by SNDRINIT (Section 2.6.5.) so that each SØ2LST entry points to one of the SØ2 Q-blocks in SØ2TBL (below). SØ2LST provides the mechanism for using PCTs and EKO buffers in round-robin fashion during a sounding. This means that the buffers are used in the order 1, 2, 3, 4, 5, 6, 1, 2,... Routine GETBFR (Section 2.9.1.1) pops the stack from the top and pushes from the bottom to circulate SØ2LST.

<u>PCTLST-NXTPCT</u> (STBØ44ØØ-STBØ7ØØØ)

The PCT data are assembled by the next-frequency routines and CLOCK (Sections 2.9.1.2 and 2.1Ø.1) into NXTPCT. PCT.TIK (Section 7.2) is also maintained in NXTPCT. Thus, the PCT data for a given pulse are available before that pulse is transmitted. However, these data cannot be put into a real PCT until 2 pulse-periods later, at EKO buffer allocation time, because as the data in the pipeline back up, the risk of the PCT becoming associated with the wrong EKO buffer increases. Therefore, routine STTSG (Section 2.9.1.1) stages the PCT data by copying NXTPCT to one of two areas in TEMPCT. PCTLST contains the addresses of the two PCT staging areas, which are used alternately as STTSG circulates PCTLST. The TEMPCT address used by STTSG to stage the PCT data is pushed to stack STGLST; routine GETBFR (Section 2.9.1.1) pops STGLST and copies the staged PCT data to a real PCT after determining that the PCT and its associated EKO buffer have been released by PICKER.

<u>Q-blocks</u> (STBØ72ØØ-STB1Ø6ØØ)

SØ2TBL contains SOUNDER's SØ2 Q-blocks, one for each PCT and EKO buffer set. SØ2TBL is initialized by SNDRINIT (Section 2.6.5). SØ2 Q-blocks are referenced through SØ2LST (above), and are described in Section 7.8.2.

STSND and ENSND are the sounding-start (SØ4) and sounding-end (SØ6) Q-blocks, respectively. The SØ4 is sent by subroutine CLOCK (Section 2.1Ø.1) at the beginning of the countdown to sounding. The SØ6 is sent after the data pipeline empties at sounding end (Section 2.9.1.1).

FAKECQ is the CØØ Q-block sent by SOUNDER to restart a PICKER task and to restart GRAPHER after being reinstated following the execution of a CALL task (Chapter 1Ø). PICKER and GRAPHER distinguish between the start and the restart CØØ by examining CØØ + 4. This value is the ET address in the start and Ø in the restart CØØ.

AGC Stacks (STB1Ø8ØØ-STB128ØØ)

During a Table-class or W-class sounding, the address of the frequency table (SCT.FTBL) entry for the frequency selected is put into SØ2 + 8, and is used by PICKER for echo acceptance by range. After a T- or W-class frequency is selected, its SCT.FTBL entry address is pushed to AFTLST by CKFILL (Section 2.7.1). Two pulse periods later, or more if MD.NFIL > Ø, GETBFR (Section 2.9.1.1) pops from AFTLST, if the incoming data are the first of a pulse set. This SCT.FTBL entry address is put into SØ2 + 8 for every data buffer for the pulse set.

AGCUP1 and AGCUP2 will be used to update the AGC attenuation bytes for receiver channels 1 and 2 when and if the Pn argument of command AGC is fully implemented.

TSG-related Tables (STB13ØØØ-STB178ØØ)

TSGDLY is a table of TSG range-Ø corrections for TSG scenarios Ø-7. The eight 16-bit, signed values are in microseconds and are computed by subroutine TDLAYS (Section 2.1Ø.3) whenever the TSG is booted, either explicitly by the BOOT TSG command, or automatically when sounding mode switching (RUN, MGET) demands. TSGDLY values are copied into the pulse palette PD.TDLY slots by subroutine TSGASS (Section 2.8.2) whenever command TSGC is given or mode switching occurs. The PICKER task uses the values from PD.TDLY to compute the final ranges of echoes.

ZPCTRL and TSGREP are contiguous in SOUNDER memory and contain the first 26 contiguous bytes of the last zero page written to the TSG. (See Section 12.4.) The FEP control vectors and the FEP program starting addresses from ZPCTRL are copied to the PD.FEPC and PD.FEPA slots of the pulse palette by TSGASS (above) as a result of command TSGC or mode switching. The four repeat counts for FEP averaging from TSGREP are copied to MD.REPS by BTALL (Section 2.9.2.2) when mode switching occurs.

These values are also used by CLOCK for HF Radar timekeeping, as described below and in Section 2.10.1.

TENMS contains 10 millisecond interval counts per TSG pace. TENMS is indexed by bits 1 and 2 of the TSG command byte (0 = MSB). A TENMS entry is multiplied by the appropriate TSGREP entry by CLOCK (above) to determine the time-value of a TSG start in 10 millisecond intervals. The default TENMS values are 1, 2, 10, and 100, and assume the standard TSG pace strapping. If the strapping on the TSG board is non-standard, command TICK must be given in order for timekeeping to be correct. TICK computes the time-values of the four TSG paces, rounding up to the next 10 ms interval, if necessary. The SOFT ticks displayed by command SYS are the TENMS values, times 10.

Table TPACE contains the TSG paces in milliseconds exactly as computed by the TICK command. These values are only used by the SYS display as the HARD ticks. The default TPACE values are 10, 20, 100, and 1000.

The ET (STB18000-STB21200)

The Environment Table (ET) is the first HF Radar glogal table discussed is this section. The ET may be considered the central global table, in as much as it is used to globalize the other SOUNDER global tables and SCAN.TAB in COUNSEL. The ET address is passed to COUNSEL at SOUNDER start up; COUNSEL in turn passes the ET address to the other sounding tasks as they are started. Having the ET address, the other tasks have access to all of the global tables, and also may make temporary use of SOUNDER's EKO buffers. The ET also contains the addresses of all of SOUNDER's reentrant subroutines (Section 2.6.3) and the HF Radar timekeeping counters. These counters are ET.SEC, the time-of-day in seconds since 1/1/77; ET.CDN, the countdown-to-sounding value in seconds, or -1; and ET.TNXT, the number of seconds until the next sounding. Timekeeping in SOUNDER is started when the TSG clock is read at sounding start (Section 2.8.1); the timekeeping counters are maintained by CLOCK (Section 2.10.1) only while SOUNDER is executing in the SPC loop (Section 2.5). See Section 7.3 for more ET details.

2.6.4. SOUNDER Tables (SNDRTBLS)

The SCT (STB21400-STB24100)

The Sounding Configuration Table (SCT) is the most frequently referenced table within SOUNDER and is included at the beginning of every recorded data set by MANAGER. There is an SCT memory image stored on disc file SYS1:SMODES.xxx for every sounding mode that can be run by the HF Radar software. (See Section 2.5.) The file extension, xxx, is "TWO" unless the HF Radar tasks are loaded with, HFRADAR xxx. The full SCT is currently exactly 2 disc sectors, or 512 bytes, in length. All 512 bytes are written with each data set by MANAGER; 382 (SCT.MD through SCT.DISC) bytes are saved on disc as the sounding mode description. (See RDMODE and WRMODE, Section 2.8.2.)

The SCT record flag in a recorded data set is hex 8000, and this flag is added to the SCT size, 512 or 200 hex, to form the 8200 hex value in the first 1/2-word of the SCT, SCT.LEN.

The SCT is divided into several sections. (Refer to the SCT. STRUC under sequence number STB03000.) SCT.LEN-SCT.XTRA contain values which are maintained independently of sounding mode. The block SCT.LOC-SCT.ELOC contains the local site parameters copied from SYS1:LOCAL.TWO, sector 0, during SOUNDER initialization. SCT.MD-SCT.END constitute a sounding mode definition. It is this section that is written by MSAV and recalled from disc by the RUN and MGET commands. Most of the values in this section are affected by SOUNDER console commands; MD.ECHO, MD.EKOF, and MD.EKOS are set by PICKER. See Section 7.1 for details. SCT.PD is the palette of pulse descriptions, modified by commands TSGC, DFAC, ANT, and TXPH. SCT.PSEQ contains the pulse set sequence, set by command PSEQ. The palette indexes build down from SCT.PSEQ, while the repeat counts build up from SCT.PREP. SCT.FTBL is the frequency table for T- and W-class sounding modes, and is roughly equivalent to the P1.0 table, KMODE. SCT.FTBL is detailed in Section 7.4, SCT.PD in Section 7.5.

FRQMAP (STB24200-STB25000)

FRQMAP is the protected frequencies map against which all sounding frequencies are checked by subroutine MAPFRQ (Section 2.6.2). FRQMAP contains all protected frequencies as bands which include the 50 kHz guard bands. Protected frequencies maps are stored on disc by PROTF (Section 10.2.3) and are read into FRQMAP by subroutine GTMAP (Section 2.9.2.2) as needed when mode switching occurs. MD.MAP specifies the map required by a given sounding mode. FRQMAP is periodically checked to ensure its integrity.

(FRQMAP is made contiguous in memory with SCT.FTBL in anticipation of adding X-class sounding modes. See Section 2.5.)

<u>The DIO Control Tables</u> (STB25200-STB26500)

Table DIO contains all of the DIO control tables for SOUNDER. These tables are detailed in Section 7.9. The DIO tables are generated by CALL task DIOGEN (Section 10.2.4), which writes sectors 0-8 of a contiguous disc file, SYS1:DIO.xxx. Subroutine GTDIO (Section 2.9.2.1) reads only sectors 6-8 of such a file into table DIO as required by the BOOT DIO command or by mode switching (RUN or MGET).

<u>Byte storage</u> (STB26700-STB28600)

Values in byte storage are stored as 8-bit quantities and normally need not begin on 1/2-word boundaries. Exceptions will be noted below. The byte storage labels are listed below in the order in which they appear in SNDRTBLS.

TSGBFR TSGBFR is the data buffer for the STTPB SVC1 parameter block (below) used to start the TSG. As such, TSGBFR holds the last command byte sent to the TSG. (See STTSG, Section 2.9.1.1.) CLOCK (Section 2.10.1) uses the command byte in TSGBFR to compute the time-value of each TSG start. Because SVC1 demands, TSGBFR must be on a 1/2-word boundary.

TSGNXT TSGNXT is set by the next-frequency routine (Section 2.9.1.2) to the next command byte that will be sent to the TSG. This command byte comes from one of the eight PD.TSGC entries in the pulse palette. STTSG (Section 2.9.1.1) copies TSGNXT to TSGBFR before the command byte is sent to the TSG, but after CLOCK (Section 2.10.1) has examined TSGBFR.

FEPINIT This is the FEP-initialize control byte used by subroutine RESFEP (Section 2.6.2).

FEPRES The FEP-reset byte is also used by subroutine RESFEP in resetting the FEP.

2.6.4. SOUNDER Tables (SNDRTBLS)

MODIX — This byte holds the index, or the SMODES disc file sector number, of the active sounding mode. It is used by WRMODE (Section 2.8.2) for the automatic updating of sounding modes on disc.

HKCHAN — If the House-keeping ADC is configured into the system by DIOGEN, it is read during the SPC loop and during the IMA loop, unless a CALL task is active. One channel (Ø-63) is read per loop. HKCHAN holds the next channel number to be read. (See RDHSK, Section 2.6.2.)

HOUR — This is the current hour of the day (Ø-23), initialized by RDTIME (Section 2.8.1) and updated by CLOCK (Section 2.1Ø.1).

FEPX — This 3-byte field contains the extension of the last FEPMEM disc file specified by the BOOT FEP command, or set by mode switching. When a new mode is loaded FEPX is compared with MD.FEPX to see if the FEP must be reloaded. (See BTALL, Section 2.9.2.2.)

TSGX — This is the current TSGMEM extension. During mode switching, BTALL (Section 2.9.2.2) compares TSGX with MD.TSGX to see if the TSG must be rebooted.

DIOX — This is the current DIO disc file extension. During mode switching, BTALL (above) compares DIOX with MD.DIOX to see if new DIO control table are needed for the new sounding mode.

PIKT — PIKT contains the file name of the currently loaded PICKER task. The mode-switching code compares PIKT with MD.PIK, set by command PIK, to see if a different PICKER task is to be swapped in for the new sounding mode. Current configurations of the PICKER task are described in Section 3.1.

C.ID — C.ID is the COUNSEL task name, used by several routines to queue COUNSEL or to display COUNSEL's name. Because C.ID (and the other two task names following) is moved with a LM-STM sequence, it must begin on a 1/2-word boundary.

P.ID — This is the PICKER task name, no matter which version of PICKER is running. P.ID is used like C.ID, above.

G.ID This is the GRAPHER task name, also assumed by any CALL task executing in GRAPHER's memory partition. (See Chapter 1Ø.) G.ID is used like C.ID and P.ID, above.

Half-word Storage (STB288ØØ-STB335ØØ)

Values in 1/2-word storage are multiples of 16-bit quantities, each of which must begin on a 1/2-word boundary. The 1/2-word labels are listed below in the order in which they appear in SNDRTBLS.

NXTACT NXTACT is used to switch control between SOUNDER's IMA and SPC loops. NXTACT is initialized to the address of GETADV (Section 2.6.1), because SOUNDER first executes in the IMA loop. (See Section 2.5.) Control goes from the IMA to the SPC loop when command RUN, CONT, or TPR is given, and NXTACT is thus set to NXTSND (Section 2.9.1.1). If no SPH schedule is active, control returns to the IMA loop after a single sounding; if SPH is active, control remains in the SPC loop until SN is satisfied or until AB is typed. AB always returns control to the IMA loop. If any of the above conditions dictate exiting the SPC loop, NXTACT is set to GETADV, and when the data pipeline empties, SOUNDER reenters the IMA loop.

CALQ When a CALL task is run (Section 1Ø.1), the CØ2 queue from COUNSEL that initiated CALL task execution is either cleared immediately, or after the CALL task has completed executing. The latter case is true if the CALL task needs temporary control of the system console. In such a case RUNCALL (Section 2.8.4) saves the address of the CØ2 Q-block in CALQ, then when a TFF queue from the CALL task signals that it is done, ENDCALL gets the address from CALQ and clears the CØ2 Q-block status, returning control of the console to COUNSEL.

CALF The CALL flag is 1 when a CALL task is active, Ø when not. If CALF = 1, RUN <task> is rejected with ERR 1Ø, TPR with ERR 12. CALF is tested to see if GRAPHER need be reinstated after an SPH sounding series has terminated, and to determine whether or not the HKADC is to be read in the IMA loop (Section 2.6.1).

2.6.4. SOUNDER Tables (SNDRTBLS)

NEXT — When an SPH schedule is aborted, the identifier of the active sounding mode and the time of the next scheduled sounding are saved in NEXT. A subsequent CONT command reinstates the SPH active mode, if not still active, and attempts to run the next sounding on schedule, based on NEXT. If it cannot, the next sounding scheduled by SPH is attempted, and so on, until a sounding can be run on schedule.

NOxxx — The device-is-dead flags are in TSG, FEP, EKO, ADC order for command NO (Chapter 8). Ø means the device in question is alive and assumed to be normal, -1 means the device is dead for a single sounding or for an SPH series, and 1 means the device has not been configured into the system by DIOGEN (Section 1Ø.2.4). NOTSG is set to -1 for a TPR run.

ERRxxx — The system device error counters are in TSG, FEP, DIO, TSG CLK, HKADC order for subroutine DEVERR (Section 2.6.2). The counter position relative to ERRTSG allows DEVERR to set the corresponding error bit in MD.STAT (Section 7.1).

NXTFIX — During a sounding, NXTFIX is the index of the base frequency, and is used as the base of any log ramp. NXTFIX is -1, if no sounding is in progress (CLOCK, Section 2.1Ø.1), and is Ø if a sounding has ended, but data remain in the pipeline. (See Sections 2.7.1 and 2.9.1.1 and 2.)

NXTFREQ — NXTFREQ is the 32-bit base frequency in Hz/1ØØ, and is used as the base of any linear ramp.

SUBFIX — SUBFIX is the ramp frequency index, and is initialized from NXTFIX at the start of each log ramp.

SUBFREQ — The 32-bit ramp frequency is initialized from NXTFREQ at the start of each linear ramp, and is in Hz/1ØØ.

RSTEP — RSTEP allows storage of four 32-bit ramp step values, because although the B-ramp frequency step is simply set from MD.SUBS (Section 7.1), Z-ramps use two steps alternately. When the first half of RSTEP is initialized, it is copied into the second half also, to obviate the need to recalculate the initial values of the Z-ramp steps. (See Sections 2.7.1 and 2.9.1.2 for details about the Z-ramp algorithm.)

NRND — NRND is initialized to MD.NSET at sounding start (STFIX, Section 2.9.1.2), and is decremented each time a pass is made through the base frequency selection. For an S-class sounding, a base frequency pass is from SFRQ through EFRQ; for T- and W-class, a base frequency pass is once through the frequency table, SCT.FTBL.

NFILL — After a pulse set has been transmitted, NFILL is set from MD.NFIL, and until Ø, NFILL is decremented for each TSG NOP start between pulse sets.

LOFREQ — LOFREQ is the minimum system frequency in Hz/1ØØ, and is derived from SNDR parameter, LOFR (Section 7.7), by SNDRINIT (Section 2.6.5).

HIFREQ — HIFREQ is the maximum system frequency in Hz/1ØØ, and is derived from SNDR parameter, HIFR, by SNDRINIT. (See MAPFRQ, Section 2.6.2.)

A1STF — The address of the 1st active T- or W-class frequency table entry is put into A1STF by subroutine CKFTBL (Section 2.8.3) at sounding start. During the sounding, AFTF (below) is initialized from A1STF at the start of each base frequency pass, so that the 1st active frequency table entry need not be found each pass.

AFTF — AFTF is initialized from A1STF (above) at the start of each base frequency pass, and is advanced to the address of the next active frequency table entry as base frequency selection progresses. As each T- or W-class base frequency is selected, the address in AFTF is pushed to stack AFTLST. (See AGC stacks, above.)

AFTQ — When GETBFR pops from AFTLST, the address of the T- or W-class frequency table entry is put into AFTQ. This address is used for SØ2 + 8 for the duration of the current pulse set.

AGCPUP — AGCPUP contains two counters, one for each receiver channel, which are used by subroutine UPAGC (Section 2.9.1.3) in updating the AGC.

2.6.4. SOUNDER Tables (SNDRTBLS)

MODID — MODID contains the ASCII version of the active sounding mode's identifier. Control characters are replaced by asterisks, which are not allowed in a mode name, and carriage returns are replaced by blanks. MODID is used by MLIST (Section 2.8.2) and PLIST (Section 2.7.9). MD.MREV (Section 7.1) contains MODID in recorded data sets.

MODIM — The binary image of a sounding mode name is kept in MODIM, and is used in searching the mode directory for the mode. (See CKIDD, Section 2.7.2.)

MAXPK — During a sounding, the maximum data peak for each receiver channel is computed for control of the AGC (Section 2.9.1.1). These maxima are kept in MAXPK and are used by UPAGC (Section 2.9.1.3).

LOAGC — LOAGC is the ADC count equivalent of the MD.PKLO mV specification of the low end of the receiver dynamic range for channels 1 and 2. LOAGC is computed at sounding start by SETUP (Section 2.9.2.2).

HIAGC — HIAGC contains the high end of the receiver dynamic range in ADC counts for receiver channels 1 and 2, and is derived from MD.PKHI, or the PKHI command. (See Chapter 8.)

SWATTN — The RF and IF attenuation bytes for S-class soundings are maintained in SWATTN. SWATTN is initialized from MD.BAG, the base attenuation settings specified by command AGC (Section 2.7.9), at sounding start by NXTSND (Section 2.9.1.1). SWATTN is thereafter updated by UPAGC. The byte order within SWATTN is RF1, IF1, RF2, IF2. (T- and W-class frequencies each have their own sets of attenuation bytes. See Section 7.4.)

FRSEC — The fractional time-of-day second is maintained in FRSEC in 1/100ths of a second. FRSEC is initialized from the 1/10ths of seconds as read from the TSG clock by RDTIME (Section 2.8.1), and is thereafter updated by CLOCK (Section 2.10.1), but only while SOUNDER executes in the SPC loop.

HRSEC — HRSEC contains the seconds into the current hour, and is initiatialized by RDTIME and maintained by CLOCK.

MODE MODE is -1 if a TPR is running, otherwise, MODE is set from SCT.MODE. Among other things, MODE determines what action is taken when AB is typed.

TODAY TODAY is set to day number and year - 1900 at SOUNDER start up from the TSG clock, and is used for the creation and last-run dates in the sounding mode directory. (See Section 2.7.2.) If the TSG clock cannot be read, TODAY is set to 1 January 1981 (00D1 hex). The number of the day of the year is in the high order 9 bits; the year - 1900 is in the low order 7 bits.

SNOLD In order to maintain the run count in the sounding modes directory, SNOLD is compared to, then set from, SCT.SNO every time the IMA loop is entered at GETADV (Section 2.6.1).

xREGS AREGS-FREGS are general register storage areas. AREGS is used by ADVICE, and other routines use BREGS-FREGS as needed, but not in rigorous order.

<u>SVC Parameter Blocks</u> (STB33700-STB35900)

The SOUNDER global SVC parameter blocks are described below. Other modules may contain parameter blocks if they are only used by that module. For details about SVC calls, consult the <u>OS/16 MT2 PROGRAMMER'S REFERENCE MANUAL</u>. The block labels below are in SNDRTBLS order.

SSVC6 This is SOUNDER's SVC6 parameter block used for intertask communication (queueing) and canceling and loading tasks for CALL/GRAPHER and PICKER task swapping.

DLAYPB This SVC2, code 23 parameter block is used to invoke the MT2 OS delay. See subroutine DLYMT2, Section 2.6.2.

ASMAP ASMAP is an SVC7 parameter block used to assign a protected frequencies map disc file to LU6. See GTMAP, Section 2.9.2.2.

CLSPB CLSPB is used to close LU6, SOUNDER's scratch logical unit.

2.6.4. SOUNDER Tables (SNDRTBLS)

SNDRQ — SNDRQ is not a parameter block (sorry), but is SOUNDER's task queue. SNDRQ is checked by ADVICE for intertask messages. See Section 2.6.1.

STTPB — STTPB is used to start the TSG and to allow SOUNDER to continue executing. STTPB is used by the pipeline (Section 2.9.1.1), by DLYTSG (Section 2.6.2), and by command TICK (Section 2.9.2.1).

WTTPB — A call to STTPB is usually followed by a call to WTTPB, which puts SOUNDER in a wait state until the TSG DONE interrupt is sensed.

PCTs and EKO Buffers (STB36100-STB36600)

The end of the SNDRTBLS module is the start of the PCT/EKO buffer area. Defining the start of the PCT/EKO buffer area here demands that SNDRINIT be loaded immediately after module SNDRTBLS. The module loading order is controlled by TETSNDR.CSS.

2.6.5. SOUNDER Initialization (SNDRINIT) [DCW]

The code in module SNDRINIT is executed when COUNSEL starts SOUNDER, after the tasks are loaded by HFRADAR.CSS. This code is only executed once and is used to initialize the SOUNDER task, initialize the mode storage disc file, assign logical units, etc.

The beginning of SNDRINIT is also the start of the PCT/EKO buffer area. Once sounding begins, the initialization code is over-written by sounding data.

INIT (SIN02500-SIN11400)

The SOUNDER task begins executing at INIT. INIT first assigns the console to Logical Unit 1 (LU1), then gets the address of COUNSEL's C00 Q-block from SNDRQ, SOUNDER's task queue. INIT puts the address of the Environment Table (ET) into C00 + 4, the essential link in making the ET, SCT, and the DIO table global, and initializes ET.ABUF, ET.LBUF, and ET.AHKD (Section 7.3). INIT then clears the C00 Q-block status.

INIT next issues an explicit STOP command to the EKO SELCH and initializes the FEP/EKO interface by setting command bits 11 and 12 of device 77 high. INIT then puts the synthesizer in range by setting it to 11 MHz and tests the synthesizer status. If the status is not 7, the synthesizer is on standby, under local control, or both. SOUNDER logs the status, pauses to allow the operator to correct the switch settings, and then when the operator continues SOUNDER, the synthesizer sequence is repeated. The initialization process will not proceed until the synthesizer is properly set up.

INIT then assigns LU5 to SOUNDER's overlay disc file, SOUNDER.OVL. (LU7 is assigned to SMODES.xxx by HFRADAR.CSS.) INIT can go no further if the LU5 assignment fails, for next it loads overlay 1-Ø1 in order to make duplicate copies of the permanent sounding modes. (See MAKEM, Sectiom 1Ø.2.1.) The ten permanent sounding modes are duplicated in reverse order, using the SCT as a buffer, thus leaving the first permanent mode (normally I-mode) as the active sounding mode when all tasks have started. If this mode is not compatible with the current revision of SOUNDER's SCT, ERROR ØØ13 is logged, and SOUNDER halts. The operator has selected the wrong SMODES file. Otherwise, with the active mode established, INIT gets its protected frequencies map and its DIO control tables. With the DIO tables established, INIT can load the FEP program memory and boot the TSG, as specified by the active mode's SCT. INIT now resets the transmitter, selecting the dummy load, but leaving the filaments and HV unaffected.

INIT next reads the local site parameters from SYS1:LOCAL.TWO, sector Ø, into SCT.LOC-SCT.ELOC (Section 7.1), and clears SCT.SKDX to zero (null), to indicate that no SRUN schedule is in progress. At this point, the SCT is fully initialized. INIT sets the devices on the DIO bus to their idle states. (See DIOGEN, Section 1Ø.2.4.)

Logical Units (SIN116ØØ-SIN136ØØ)

Logical units 1 and 5 are assigned by INIT (above), and LU7 is assigned by HFRADAR.CSS. LU2, 3, and 4 are assigned here. SOUNDER LU assignments are summarized below. The device name, if any, recognized by the MT2 OS is enclosed in parentheses.

LU1	Console (CON:)
LU2	TSG (TSG:)
LU3	EKO (EKO:)
LU4	FEP (FEP:)
LU5	SOUNDER.OVL
LU6	Scratch: LOCAL.TWO, xPROT.TWO, TSGMEM.xxx, FEPMEM.xxx, DIO.xxx, CALL tasks, GRAPHER.TSK, PICKER tasks, mag-tape (MAG:)
LU7	SMODES.xxx

2.6.5. SOUNDER Initialization (SNDRINIT)

All logical units are permanently assigned, except LU6. All disc files are on SYS1, except the CALL tasks, which are on SYSC. HFRADAR xxx assigns LU7 to SMODES.xxx; if xxx is not specified to HFRADAR.CSS, SMODES.TWO is assigned. Although LU4 is assigned to the FEP, the FEP is not under direct SOUNDER control in P2.Ø. (See PIPLN, Section 1Ø.2.15 for details.)

SETIME (SIN138ØØ-SIN272ØØ)

The SETIME routine reads the TSG clock and sets the MT2 OS clock to the TSG clock time. SETIME is a variation of RDTIME (Section 2.8.1) provided by J. R. Winkelman to format the TSG time for SVC2, code 1Ø, also written by JRW. If the TSG clock cannot be read correctly, or if it is not configured into the system by DIOGEN (Section 1Ø.2.4), TODAY (Section 2.6.4) is set to 1 January 1981. TODAY is used only for date-marking the sounding modes directory. In case the day changes during a sounding session, TODAY is not updated.

Final Initialization (SIN273ØØ-SIN323ØØ)

Because frequencies in SOUNDER 2.Ø2 are specified and manipulated in Hz/1ØØ, SNDRINIT converts SNDR parameters LOFR and HIFR (Section 7.7) to the 32-bit Hz/1ØØ values LOFREQ and HIFREQ. MAPFRQ (Section 2.6.2) uses these values to check for out-of-range frequencies.

Next, SØ2LST and SØ2TBL (Section 2.6.4) are initialized, using SNDR parameters EKOSIZE, SØ2SZ, and NEKOBFS, and the PCT size from the PCT STRUC. This routine would need to be changed if the structure of the SØ2 Q-block were to change. (See Section 7.8.2.) To change the number of SOUNDER buffers, simply change NEKOBFS in SNDR (Section 7.7). The total size of the PCT/EKO buffer area is parameterized in SNDRINIT; the sizes of SØ2LST and SØ2TBL are parameterized in SNDRTBLS.

The SOUNDER task is now fully initialized, and SNDRINIT informs COUNSEL of the fact by sending the SØØ Q-block, which contains the address of SOUNDER's command scan list, SMNEM (Section 2.1Ø.1), in SØØ + 4, and the address of SOUNDER's log in message in SØØ + 2. In anticipation of sounding, the PICKER task name, P.ID, is put into the SVC6.ID field of the SVC6 parameter block, SSVC6 (Section 2.6.4). Finally, SOUNDER's registers (GO-P.EKO, Section 2.9.1.1) are initialized, and the IMA loop is entered at GETADV (Section 2.6.1).

2.7. Overlay Area Ø [DCW]

Overlay area Ø is 512 bytes, or 2 disc sectors, in length and is situated in memory just past the PCT/EKO buffer area. Most, but not all, of SOUNDER's commands cause an overlay to be loaded into area Ø. One of ten current overlays can occupy area Ø. These overlays, Ø-ØØ through Ø-Ø9, are described in Sections 2.7.1 through 2.7.1Ø, respectively.

2.7.1. Overlay Ø-ØØ (SNDROØØØ) [DCW]

A RUN or CONT command causes SNDOV (Section 2.6.1) to load overlay Ø-ØØ. Also loaded at this time are 1-ØØ, 2-ØØ, and 3-ØØ. These overlays remain in memory for a single sounding or for an entire SPH sounding series. Overlay Ø-ØØ is part of the next-frequency routine, NXTFRQ (Section 2.9.1.2). Its function is to sequence the pulse sets, that is, it controls base and ramp frequency selection. (See Section 2.5.) Overlay Ø-ØØ sequences pulse sets for all Sweep-, Table-, and W-class sounding modes.

OVØ (SØØØ2ØØØ)

Because SOUNDER's overlay loader (Section 2.6.1) does not rely on the MT2 OS overlay loading system, each overlay must make its memory address known. OVØ, an ENTRY of Ø-ØØ, does nothing but establish the address of overlay area Ø. The first overlays of areas 1, 2, and 3 each have a corresponding OVn as an ENTRY.

NXTFRQ and Ø-ØØ Registers

Because this overlay is part of the next-frequency routine, NXTFRQ, the register usage for both NXTFRQ and Ø-ØØ is listed here. NXTFRQ has exclusive use of register storage area FREGS (Section 2.6.4) throughout a sounding.

2.7.1. Overlay Ø-ØØ (SNDROØØØ)

REG	USE
RØ:	Scratch.
R1:	Number of ramps on base frequency (MD.NRAM).
R2:	Number of steps in each ramp (MD.SUBN).
R3:	PSEQ loop repeat count.
R4:	PSEQ 1st-of-loop index.
R5:	SCT.PREP index.
R6:	SCT.PSEQ index.
R7:	SCT address (S.).
R8:	RSTEP index (Ø or 4).
R9:	DIO tables address (D.).
R1Ø:	RSTEP base address.
R11:	Scratch.
R12:	High order base or ramp frequency, scratch.
R13:	Low order base or ramp frequency, scratch.
R14:	Base or ramp frequency index, scratch.
R15:	Address of next NXTFRQ entry, scratch.

Table 2.4. NXTFRQ REGISTER USAGE.

The Z-ramp Algorithm

To keep the ramp frequency code as small and as fast as possible, both B-ramp and Z-ramp steps are kept in RSTEP in SNDRTBLS (Section 2.6.4). RSTEP contains four 32-bit values that are used as follows.

The B-ramp frequency step is simply set from MD.SUBS, but Z-ramps use two steps alternately. These two steps are derived from MD.SUBS and MD.NRAM. (See below.) For example, assume a series of frequencies that are numbered with consecutive integers. The command, RAMP 4,5,2,Z produces the following ramp structure.

```
Ramp Ø:  1 15 17 31 33   (Step 14 and 2)
Ramp 1:  3 13 19 29 35   (Step 1Ø and 6)
Ramp 2:  5 11 21 27 37   (Step 6 and 1Ø)
Ramp 3:  7  9 23 25 39   (Step 2 and 14)
```

Each row of numbers is a ramp; there are 4 ramps, numbered Ø-3, with 5 steps in each ramp. The basic ramp step is 2, but in sequencing the Z-ramps, the two steps enclosed in parentheses for each ramp must be applied alternately. As can be seen, these two ramp steps must be adjusted ramp-to-ramp.

In the above example, the ramp dimensions as stored in the SCT are,

```
MD.NRAM = 4  (No. of ramps on base frequency.)
MD.SUBN = 5  (No. of steps in each ramp.)
MD.SUBS = 2  (The basic ramp step.)
```

Because the first frequency in a Z-ramp sequence is equivalent to the base frequency of the ramp, the first base frequency in the example is numbered 1. Therefore, the first frequency, f1, of any ramp (Rn) is computed by INSUBN as,

```
(a.)    f1(Rn) = 1 + (n * MD.SUBS)
```

The two 32-bit Z-ramp steps are initialized by RESDTA in NXTFRQ as follows.

```
(b.)    Z-ramp step(1) = MD.SUBS * (2 * MD.NRAM - 1)
        Z-ramp step(2) = MD.SUBS
```

Z-ramp step(1) is put into RSTEP at +Ø and at +8, Z-ramp step(2) at +4 and at +12. The second copies of Z-steps (1) and (2), at +8 and +12, are used to reinitialize the Z-ramp steps when a new base frequency is selected by NXTBAS, below. Thus, the time and space required to recalculate the initial values are conserved. For a B-ramp, all four RSTEP entries are set to MD.SUBS.

After each pulse set, CKSTEP (below) toggles the RSTEP index (R8) between Ø and 4 without regard to ramp type. For a Z-ramp, this results in using the two Z-ramp steps alternately; for a B-ramp, it makes no difference, but no time is lost making a test on the ramp type. Notice also that the steps in RSTEP may be in either frequency or frequency index form.

At the end of each ramp, the two Z-ramp steps in RSTEP at +Ø and +4 are adjusted by CKRAMP.

```
(c.)    Z-ramp step(1') = step(1) - (2 * MD.SUBS)
        Z-ramp step(2') = step(2) + (2 * MD.SUBS)
```

This adjustment is made MD.NRAM - 1 times at each base frequency. The MD.SUBN dimension appears to be passive in this algorithm, but controls the number of times that the RSTEP index (R8) is toggled.

The above simple arithmetic allows square or rectangular ramps of any reasonable dimensions to be specified. For an S-class sounding, command STEP, in connection with RAMP, determines the relationship between ramp sequrnces. For example, ramps can be overlapped by making STEP less than the product of the first two RAMP arguments. The following commands would produce an S-class sounding of only one base frequency, but with a ramp of 1296 pulse sets, covering 1588 through 14982 kHz.

```
SFRQ 1588          (Index = 1867)
EFRQ 15ØØØ         (Index = 3162)
STEP 13ØØ          (1867 + 13ØØ > 3162)
RAMP 36,36,1,Z
```

EFRQ and STEP can be anything, provided the end of sounding is sensed at the selection of the second base frequency. Notice that care must be taken to not generate very large frequency indexes, which could cause divide faults in NTOFR (Section 2.6.3).

<u>NXTF1</u> <u>and</u> <u>NXTF2</u> (SØØØ23ØØ-SØØØ26ØØ)

As noted above, Ø-ØØ is part of NXTFRQ. NXTFRQ enters this overlay only at NXTF1 or NXTF2. The two ENTRY points are standardized in case overlays need be added to area Ø to sequence pulse sets for future sounding classes. In overlay Ø-ØØ, NXTF1 jumps immediately to INRAMP, NXTF2 to CKSTEP. These routines are detailed below.

<u>INRAMP</u> (SØØØ27ØØ-SØØØ33ØØ)

INRAMP is entered via NXTF1 from NXTFRQ after the calibration cycle to initialize for the first pulse set of a sounding. Thereafter, INRAMP is entered from NXTBAS (below) after each base frequency is selected to initialize for a ramp sequence, if any, and for the first pulse set at the new base frequency.

INRAMP gets the number of ramps (MD.NRAM) into R1, and continues executing at INSUBN, below.

<u>INSUBN</u> (SØØØ34ØØ-SØØØ64ØØ)

INSUBN initializes for the next frequency ramp, if any, on the base frequency just selected. The first B-ramp frequency or index, as specified by MD.CTRL (Section 7.1), is simply set to the base frequency or index. (See NXTFIX, SUBFIX, NXTFREQ, and

SUBFREQ, Section 2.6.4.) The first Z-ramp frequency or index is computed as in (a.), above. Thus, SUBFREQ or SUBFIX is initialized for the next ramp, if specified. INSUBN gets the number of ramp steps (MD.SUBN) in R2, sets R8 to select the first ramp step from RSTEP, and returns to NXTFRQ at entry INPSET, to initialize for the first pulse set of the ramp. (R8, the RSTEP index, is toggled between Ø and 4 in order to use the two Z-ramp steps alternately. See the explanation of the Z-ramp algorithm, above.)

CKSTEP (SØØØ65ØØ-SØØØ91ØØ)

CKSTEP is entered via NXTF2 (above) from NXTFRQ after a complete pulse set has been transmitted; it is not entered from any Ø-ØØ routines.

CKSTEP is entered at the end of a pulse set to check for another step in a ramp, if specified. R2 is decremented and if it goes to Ø, the ramp is complete, and control goes to CKRAMP, below. Otherwise, R8, the RSTEP index, is toggled to select the correct Z-ramp step, and SUBFIX or SUBFREQ is incremented as specified by MD.CTRL. R15 is set to INPSET in NXTFRQ, to initialize for the next pulse set, and control goes to CKFILL, below.

CKFILL (SØØØ92ØØ-SØØ1Ø6ØØ)

CKFILL is entered before a new pulse set is begun to check for a TSG NOP fill sequence between pulse sets. (See command NFIL, Chapter 8.) At this point, it is known that the previous pulse set has been completely transmitted, and in case the sounding is T- or W-class, the address of the next base frequency table entry, AFTF, is pushed to stack AFTLST. (See AFTF, AFTQ, and AFTLST, Section 2.6.4.) CKFILL then checks MD.NFIL. If the NOP fill count is Ø, the routine whose address is in R15 is executed, leading to the beginning of the next pulse set. Otherwise, MD.NFIL is put into NFILL, GO is set to hex 8ØØØ, and the fill sequence is begun. NXTFRQ will decrement NFILL to Ø before continuing with the next pulse set.

CKRAMP (SØØ1Ø7ØØ-SØØ126ØØ)

At the end of a ramp, CKRAMP is entered to check if another ramp is specified on the current base frequency. R1 is decremented and if it goes to Ø, the ramp sequence is finished, and control goes to NXTBAS to select the next base frequency. Otherwise, if the Z-ramp is specified by MD.CTRL, the two Z-ramp steps in RSTEP are adjusted. (See (c.) in the Z-ramp algoritm, above.) R15 is set to INSUBN, and control goes to CKFILL. INSUBN will be executed after the TSG NOP fill sequence, if specified.

NXTBAS (SØØ128ØØ-SØØ189ØØ)

NXTBAS is entered when MD.NRAM ramps have been transmitted on the current base frequency, or when the ramp sequence is finished. For an S-class sounding, NXTFREQ or NXTFIX is incremented by MD.STEP; for T- or W-class, NXTFREQ is set to the next active frequency in SCT.FTBL, and AFTF is set to the active frequency's SCT.FTBL entry address.

If EFRQ, the ending frequency, is reached for an S-class sounding, NRND (set from MD.NSET) is decremented and if it goes to Ø, the sounding is over. Otherwise, the next base frequency is SFRQ, the starting frequency, and another pass from SFRQ through EFRQ is begun. If the end of SCT.FTBL is reached for a T- or W-class sounding, and if NRND does not go to Ø when decremented, the next base frequency is gotten via A1STF (Section 2.6.4), AFTF is set to A1STF, and the next pass through SCT.FTBL is begun. If NRND does go to Ø, the T- or W-class sounding is finished.

Finally, the ramp steps are reinitialized by copying the second half of RSTEP to its first half, and control goes to CKFILL. INRAMP (R15) will be executed after the NOP fill sequence, if any.

ENDSND (SØØ19ØØØ-SØØ205ØØ)

ENDSND is entered when the normal end of a sounding is detected, or if an AB is typed at the console. If ENDSND is entered as the result of AB, NXTACT (Section 2.6.4) is set to GETADV to force control back to the IMA loop after the data pipeline empties. In any case, SCT.ETS, the sounding end time, is set from ET.SEC, the current time-of-day, GO is set to -1, and NXTFIX is cleared to Ø (Section 2.6.4). NXTFIX remains Ø until the data pipeline clears.

OVSZ (SØØ217ØØ-SØØ222ØØ)

As noted above, SOUNDER's overlay loader must know the memory addresses of the overlay areas. The overlay loader must also know the size of each overlay area in order to set up the SVC1 disc read. (See Section 2.6.1.) To keep things simple, while wasting very few bytes of memory, overlay areas are forced to even multiples of 256, the size in bytes of one disc sector. Overlay area Ø is forced to 512 bytes by OVSZ.

2.7.2. Overlay Ø-Ø1 (SNDROØØ1) [DCW]

Overlay Ø-Ø1 is the mode-handling overlay. This is the first overlay loaded in response to the RUN, CONT, MGET, MDEL, and MSAV commands. All command routines in this overlay load 1-Ø1 (Section 2.8.2) to read and write sounding modes and the sounding modes directory. The sequence of events leading to a sounding begins in this overlay. Overlay Ø-Ø1 calls 2-Ø1 which does all initializing required for a sounding or for an SPH series. Control goes from 2-Ø1 to SNDOV (Section 2.6.1), which loads the SPC overlays, reads the TSG clock, then enters 2-ØØ and the SPC loop at SINIT (Section 2.9.1.1).

CONT (SØ1Ø21ØØ-SØ1Ø31ØØ)

Command CONT is rejected with ERR 24 unless an SPH schedule has been aborted, i.e., unless there is something with which to continue. Otherwise, CONT gets the aborted mode name in R3, sets NEXT to 2 (Section 2.6.4), and enters the RUN routine at RECMODE, where the aborted mode is recovered, if necessary. CONT proceeds just like RUN until routine SNDOV (Section 2.6.1).

RUN (SØ1Ø32ØØ-SØ1Ø6ØØØ)

The RUN command routine first scans for a sounding mode name. If no mode name is specified, the currently active mode is run. If a name is given, RUN checks to see if that name is in the mode directory. If the mode name has one or two characters and is not listed in the directory, it is rejected with ERR 1. If the mode name has more than two characters, RUN assumes that the operator wants to run a CALL task (Section 1Ø.2), loads 1-Ø3, and goes to RUNCALL (Section 2.8.4). Otherwise, RUN loads the specified sounding mode, if it is not already in the SCT. RUN sets SCT.MODE from MD.TMDE and clears the device error bits in MD.STAT (Section 7.1). If the sounding mode is T- or W-class, RUN loads 1-Ø2 and calls subroutine CKFTBL (Section 2.8.3), which does special initialization for T- and W-class. If no frequency table entries are active (Section 7.4), the RUN command is rejected with ERR 5. Otherwise, RUN loads 2-Ø1, and continues at SETUP (Section

2.9.2.2). SETUP performs all initialization that can be done before the SPC overlays are loaded by SNDOV (Section 2.6.1).

MGET (SØ1Ø61ØØ-SØ1Ø71ØØ)

MGET first scans for a sounding mode name (like RUN, above). If no name is specified, MGET loads 1-Ø1 and goes to MLIST (Section 2.8.2), where the currently active mode is displayed to the console. If the mode name is not valid, it is rejected with ERR 1. Otherwise, MGET loads the specified mode into the SCT, and then goes to MLIST in 1-Ø1.

MDEL (SØ1Ø73ØØ-SØ1Ø85ØØ)

The MDEL command routine first scans for a sounding mode name. The MDEL command is rejected with ERR 1 if no mode is specified, if the specified name has only one character (a permanent mode), or if the name is not in the mode directory. Otherwise the mode is deleted from the directory by writing a Ø (null) over its name. Notice that the SCT image is not lost from SMODES until the deleted mode's sectors are required by a subsequent MSAV command. (See DUTIL, Section 1Ø.4.1, but use great caution.)

MSAV (SØ1Ø86ØØ-SØ1121ØØ)

The MSAV command routine scans the sounding mode name. The MSAV command is rejected with ERR 1 if no name is given, if the given name has only a single character (a permanent mode), if the first character of the name is numeric, or if the name contains an asterisk character. A numeric character could be taken for a starting time; the asterisk (*) has special meaning to the scheduler. See Section 1Ø.2.5. If the name is valid, MSAV scans the mode directory to see if the mode already exists. If not, the directory is scanned again for an empty slot. An empty slot has a name of Ø, or is null. See MDEL, above. If there is no room in SMODES for a new mode, the command is rejected with ERR 15. Otherwise, the message, "OLD MODE" or "NEW MODE" is logged, and the mode is written to SMODES and the directory updated. If an old mode is being resaved, its creation date is marked with TODAY (Section 2.6.5), but its run count is left intact. The run count for a new mode is zeroed.

SCARG (SØ1123ØØ-SØ117ØØØ)

Subroutine SCARG is called by the above command routines to scan the command argument for a mode name of valid form. SCARG takes one of three exits, depending on the results of the argument scan.

Call+Ø: Carriage return only, or * in mode name.
Call+2: Numeric argument.
Call+4: Good name returned in R3 for CKIDD, below.
Call+4: Name > 2 characters; R3 = 2 (CKIDD mismatch forced).

Different command routines take different action as the result of a given SCARG exit.

CKIDD (SØ1172ØØ-SØ1197ØØ)

Subroutine CKIDD is called to read the SMODES directory from disc and to check if the mode name in R3 exists. If CKIDD finds the name in the directory, it returns the directory entry address in R12 and the first sector address of the SCT image in R4. If no match is found, R4 is set to -1.

CKIDD has a second entry, CKID, which is used when it is known that the SMODES directory is already in memory.

LOADM (SØ1198ØØ-SØ1221ØØ)

Subroutine LOADM is called by the command routines for RUN, CONT, and MGET to load the SCT image of the mode specified into the SCT. LOADM is called with the first sector address of the SCT image in R4, and the mode's directory entry address in R12. LOADM first checks to see if the mode is already active, i.e., already in the SCT. If it is, LOADM simply returns to the caller. Otherwise, LOADM loads overlay 1-Ø1, writes the current mode to SMODES, then reads the new mode into the SCT. (See WRMODE and RDMODE, Section 2.8.2.) After the mode is in the SCT, RDMODE compares its MD.MREV to SCT.MREV (Section 7.1). If the mode is not compatible with the current SOUNDER, ERR 13 results. If the mode is compatible, LOADM loads overlay 2-Ø1 and calls BTALL to get the required protected frequencies map, DIO control tables, FEP and TSG programs, and PICKER task.

2.7.3. Overlay Ø-Ø2 (SNDROØØ2) [DCW]

SOUNDER command TPR causes overlay Ø-Ø2 to be loaded. The TPR command is useful for debugging new routines in PICKER, MANAGER, and GRAPHER, because the same input data can be played through the system repeatedly. Furthermore, TPR can be run on systems with no RF hardware, TSG, FEP, or EKO interface.

TPR can also be used to establish data sets on the disc for GPION or CALL task review. (See MENUS, Section 1Ø.3.3.)

A tape file read by TPR must have four records preceding the data. As is true of all HF Radar data sets, the first record is the SCT. (See TENUS, Section 1Ø.3.1.) The tape file is rejected unless SCT.MODE of the SCT tape record is flagged with 5Ø hex.

TPRUN (SØ2Ø25ØØ-SØ2Ø92ØØ)

TPRUN first checks the CALL mode flag, CALF (Section 2.6.4). If a CALL task is active, the TPR command is rejected with ERR 12, primarily to avoid magnetic tape conflicts. TPRUN also logs an error message if it cannot assign LU6 to MAG:.

TPRUN next loads 1-Ø1 and calls WRMODE (Section 2.8.2) to save the currently active sounding mode, then loads 1-ØØ, 2-ØØ, and 3-ØØ, the standard SPC overlays for areas 1, 2, and 3. These overlays are only needed for the countdown, which is executed to allow time for the down-stream tasks to initialize for what they assume to be a normal sounding.

TPRUN then sets the HKDIO flag word to AØØ2 hex for MANAGER. The HKDIO size is set to the mimimum 2, because the HKADC and DIO ports are not read during a TPR run. The mode-independent values of the current SCT are copied to the HKDIO area in 1-ØØ, and the first tape record is read into the SCT. If SCT.MODE is not flagged with 5Ø hex, TPRUN restores the displaced SCT and rejects the tape file with ERR 11. The tape is left positioned after the first record.

If the tape file is legitimate, the next three records are skipped, leaving the tape positioned at the start of the PCT/EKO data. ET.TNXT is set to MD.CDN + 1, and NOTSG is set to -1, flagging the TSG dead for the TPR run. MODE (Section 2.6.4) is set to -1, GO is cleared to Ø for IMALOOP (Section 2.6.1), and NXTACT is set to NXTSND, or the SPC loop. TPRUN then goes to OKADV to clear the CØ2 Q-block status from the TPR command. IMALOOP, sensing GO = Ø, will go to SINIT

(Section 2.9.1.1), and the countdown-to-sounding will begin.

GOTPR (SØ2Ø94ØØ-SØ2152ØØ)

GOTPR is entered from NXTSND (Section 2.9.1.1) after the countdown is finished. GOTPR simulates the SPC loop action by reading the PCT/EKO data from tape into one of SOUNDER's PCT/EKO buffers, then queueing PICKER with the SØ2 Q-block associated with that buffer. As in the normal SPC loop, GOTPR will not use a buffer until PICKER has released it.

If the sounding is T- or W-class, PICKER will use the ranges for FØ for echo acceptance, because GOTPR puts the address of the first SCT.FTBL entry (Section 7.4) into each SØ2 + 8. This is thought to be adequate for debugging purposes, and in any case, overlay Ø-Ø2 has no room for expansion.

After queueing PICKER with each SØ2, GOTPR calls ADVICE (Section 2.6.1). If AB has been typed, the ABORT routine, sensing MODE = -1, clears MODE to Ø to inform GOTPR of the AB. If an AB is typed, GOTPR skips to the next EOF on tape, and sends the SØ6 to PICKER. Otherwise, GOTPR proceeds until the tape EOF is read.

After the TPR run is finished, the diaplaced sounding mode (SCT) is restored, leaving SOUNDER as it was before the TPR command. MODE is unconditionally cleared to Ø, NXTACT is set to GETADV, and the IMA loop is entered.

WARNING: The mode-independent section of the SCT (Section 7.1) is saved to HKDIO and restored with SVC2, code 18, which copies a maximum of 127 (hex 7F) bytes. The mode-independent section of the SCT, however, is 13Ø bytes long. Only SCT.XTRA (73 hex) bytes are saved, so if bytes in SCT.XTRA are set by the user, they may be lost after a TPR run.

2.7.4. Overlay Ø-Ø3 (SNDROØØ3) [DCW]

Overlay Ø-Ø3 is loaded in response to the UDIO, DIO, PSEQ, or TXANT command. These commands invoke routines UDIO, CDIO, PSEQ, and TXANT, respectively.

UDIO (SØ3Ø21ØØ-SØ3Ø74ØØ)

The UDIO command is used to set or display a user-defined DIO pulse palette extension. (See Sections 2.5 and 7.9.) The first argument of the UDIO command is a hex DIO port address

(Ø-63). If the port so specified has not been configured by DIOGEN (Section 1Ø.2.4) to accept user-specified data pulse-to-pulse, the UDIO command is rejected with ERR 19.

If the DIO port address is the only argument given, UDIO simply displays the hex data bytes specified for the 8 pulse descriptions. These data bytes are of course displayed in the same order as are the pulse palette descriptions by command PTP. The order in which the bytes are sent to the DIO port each pulse set during a sounding is controlled by command PSEQ, below.

One to eight hex arguments may follow the DIO port address. These are the data bytes (Ø-FF) to be sent to the DIO device in PSEQ order. If less than eight bytes are specified, the last byte specified is used for the unspecified bytes. For example, UDIO d,F5,A, sets the byte for PD 1 to F5 and sets the bytes for PDs 2-8 to A for DIO port d. Only the low order 8 bits are used; for example, 3FF is taken as FF.

After the bytes are input, they are displayed back, as described above.

The UDIO command is included in SOUNDER to allow the user to quickly modify a given pulse palette extension for experimental purposes. The bytes are modified only in memory, in the DIO.PTP section of the DIO control tables, not on disc. Therefore, no permanent record of these data are kept; they are destroyed by the next BOOT DIO command. Once the user has determined what the DIO port data should be, the lengthier process of calling DIOGEN can be used to make the data permanent. The data bytes set by UDIO are, however, recorded with the sounding data sets by MANAGER.

The receiver input multiplexors for channel 1 (DIO 1Ø) and 2 (DIO 11) are normally configured as DIO.PTP devices, with the attribute of being user-defined. Although these two ports are dedicated by definition, the SPC loop treats them as any other pulse-to-pulse port for reasons of efficiency. Also, although pulse-to-pulse data can be specified for these ports, both by UDIO and DIOGEN, these data are overridden, at sounding start, by specifications from the ANT command (Section 2.7.5). The above is also true for the receiver bandwidth controls, but only if they are defined pulse-to-pulse. The BDW command, however, only specifies the bandwidths for an entire sounding, not pulse-to-pulse. The transmitter attenuator and the wide-band filters, if installed, are treated similarly.

CDIO (SØ3Ø75ØØ-SØ31Ø6ØØ)

The DIO command is used to write data immediately to any of the devices on the DIO bus. In response to the command, DIO AA,DD[,MM], CDIO reads DIO port AA and logically ANDs the data byte just read with mask MM (old data). CDIO then ANDs the specified data byte, DD, with the inverted mask (new data). The old data are ORed with the new data and written to DIO AA. If no mask is specified, a mask of Ø is used, which saves no old data. All arguments to the DIO command are 8-bit hexadecimal. If more than 8 bits are specified, only the low order 8 bits are used.

The DIO command routine is entirely independent of the DIO control tables and does not recognize, nor protect, dedicated DIO ports. A DIO command specification may be overridden by a DIOGEN specification once sounding begins. (See DIOGEN, Section 1Ø.2.4.)

PSEQ (SØ31Ø8ØØ-SØ3174ØØ)

The sequence of the pulses in each pulse set is specified by command PSEQ. The PSEQ command can specify simple or repeated sequences, in any combination, provided that repeated sequences are not nested. The maximum number of arguments, including both Pulse Description (PD) indexes and repeat counts, is currently 32. The PD indexes entered are 1-8, the repeat counts must be in the range 2-256. For the present, PSEQ should specify four pulses per pulse set, unless PICK5PPS or PICK6PPS is selected. (See Sections 2.5 and 3.1.)

The PSEQ sequence is saved in the 32-byte SCT.PSEQ area of the SCT (Section 7.1). The PD indexes are stored in SCT.PSEQ in the order in which they are encountered in the command argument list. Repeat counts are also stored in order, but from the end of SCT.PSEQ (SCT.PREP) toward the table start. One is subtracted from the PD indexes and repeat counts before they are stored, one per byte. If a PD index is the first of a repeated sequence, it is flagged with hex 8Ø; the last PD index of a repeated sequence is flagged with hex 4Ø. The last PD index of the pulse set sequence is flagged with hex 2Ø. A PD index byte may include any or all of the flags; repeat counts are not flagged.

Argument list format errors result in ERR 16. If the PSEQ command is not subsequently given, error free, an attempted

RUN will also result in ERR 16. The PSEQ sequence must be established in order for a sounding to run. The default sequence is, PSEQ 1,2,3,4. (See Table 2.1.)

Once the pulse set sequence is established, PSEQ can set the number of pulses in a pulse set, MD.PPS. MD.N is set to MD.PPS * 2, because there are two receiver channels, and if the new MD.N is less than MD.M, MD.M is set to the new MD.N. (See KEEP, Section 2.7.7.) Finally MD.HPE, 1/2-words per echo, is set to (MD.PPS * 4) + 1 (One X,Y pair per pulse per channel, plus the range word). MD.HPE can also be set explicitly with command HPE (Section 2.7.6) if complementary coding or embedded TSG NOPs are used. In such cases, MD.PPS is greater than the effective, or data-producing, number of pulses per set, and the explicit HPE must follow the PSEQ command.

The SCT.PSEQ table is decoded by NXTFRQ (Section 2.9.1.2) to sequence the pulses within a pulse set at each base and ramp frequency, as specified.

TXANT (SØ3176ØØ-SØ3193ØØ)

The TXANT command is used to specify a transmitting antenna (Ø or 1) for a T- or W-class sounding mode, or to specify the frequency at which to switch transmitting antennas for an S-class sounding mode. TXANT determines the form of the command argument by testing the high order bit of MD.CTRL (Section 7.1). The antenna number or the frequency index of the specified switching frequency, depending upon the class of the active sounding mode, is stored in MD.TXAN.

Currently, only the BAS system can make use of this command. Unfortunately, time did not permit its implementation in the SPC loop.

2.7.5. Overlay Ø-Ø4 (SNDROØØ4) [DCW]

Overlay Ø-Ø4 contains only two fully implemented command routines. Routine TFRQ services all forms of the F command, while RNGE specifies the range of PICKER echo retention. Command VUF is not yet implemented in SOUNDER 2.Ø2.

TFRQ (SØ4Ø2ØØØ-SØ4117ØØ)

Command F is used to set T- and W-class frequencies and to display these frequencies and the echo range limits associated with them. All forms of the F command are rejected with ERR 14 unless the currently active sounding mode is T- or W-class.

TFRQ first loads overlay 1-Ø2 (Section 2.8.3) which contains display routines DISPTF and DSPALL. Also in 1-Ø2 are QSØ8, which sends the SØ8 Q-block to GRAPHER (Section 7.8.2) and GETF, which gets frequencies from command argument lists.

DSPALL is called to display all active SCT.FTBL entries (Section 7.4), if command F is given with no arguments. If only a frequency number (Ø-9) is specified, DISPTF displays only that frequency. If "X" follows the frequency number, the specified SCT.FTBL entry is deactivated by storing -1,Ø as its 32-bit frequency (FR.FREQ). Also, when FnX is given, the ranges for T-class modes are set to the default values Ø (FR.RMIN) and 3333 (FR.RMAX) microseconds, which correspond to Ø and 5ØØ kilometers, and the attenuation word (FR.ATTN) is set to MD.BAG, the base attenuation settings specified by AGC (Section 2.7.9). For this reason, if other than the default base attenuations of Ø are desired, the AGC command must be given before the F command. FR.RMAX is set to 6666 microseconds (1ØØØ km) for W-class sounding modes. Setting FR.RMIN to Ø for T- and W-class modes causes the calibration data to be retained by default.

If the argument following the frequency number is "+" or "-", the frequency in FR.FREQ is adjusted up or down 1ØØ Hz, the control resolution of the synthesizer. If a decimal integer, n, follows the "+" or "-", the adjustment is n steps of 1ØØ Hz. If an attempt is made to adjust the frequency in an inactive SCT.FTBL entry, the command is rejected with ERR 7. An inactive entry has -1,Ø in FR.FREQ. If the adjusted frequency is found to be protected by MAPFRQ (Section 2.6.2), the upward or downward adjustment continues until an unprotected frequency is found or until HIFREQ or LOFREQ (Section 2.6.4) is reached.

Any argument that follows the frequency number except "X", "+", or "-" is assumed to be a frequency. If the specified frequency is protected, it is adjusted upward as explained above. The next lower unprotected frequency would be selected with a subsequent command, Fn-. When a frequency is selected or adjusted, its associated FR.ATTN is initialized to MD.BAG, and the new frequency is displayed.

Anytime a frequency is displayed with the F command, SOUNDER sends GRAPHER the SØ8 Q-block, which contains the frequency number at displacement +4. In response, GRAPHER displays an echo range bar, superimposed at that frequency on an ionogram.

RNGE (SØ411900-SØ413800)

The RNGE command is used to set FR.RMIN and FR.RMAX in SCT.FTBL for the specified frequency number. The frequency number (Ø-9) is the first argument of the RNGE command, and has the exact meaning as its counterpart in the F command.

If a single range is specified, it is taken to be for FR.RMIN; FR.RMAX is not changed. The ranges specified to RNGE are in kilometers, but are stored in FR.RMIN and FR.RMAX in microseconds. PICKER retains all echoes between FR.RMIN and FR.RMAX that meet the KEEP criterion. The range limits for T- and W-class frequencies are conveyed to PICKER during sounding by putting a frequency's SCT.FTBL entry address into SØ2 + 8 by QPCKR (Section 2.9.1.1).

If given for a T- or W-class sounding mode, RNGE causes the SØ8 Q-block (above) to be sent to GRAPHER. However, command RNGE can also be used for S-class sounding modes to set up PICKER's RANGE command. In such a case, GRAPHER is not queued. For example, the commands,

```
RNGE 2,85,115
RANGE 2
```

tell PICKER to ignore all echoes below 85 km and above 115 km for all frequencies of an S-class sounding.

GTATF (SØ4139ØØ-SØ4151ØØ)

Subroutine GTATF is called by both TFRQ and RNGE to get a frequency number from the command argument list. GTATF returns the frequency number in R6 and the address of the corresponding SCT.FTBL entry in R5. These values are also used by subroutines DISPTF and QSØ8 (above) in overlay 1-Ø2.

2.7.6. Overlay Ø-Ø5 (SNDROØØ5) [DCW]

Overlay Ø-Ø5 contains all of the main pulse palette description commands, DFAC, TXPH, TSGC, and ANT, plus command TXANT. See

Section 7.5 for the SCT.PD (pulse palette) layout.

DFAC (SØ5Ø21ØØ-SØ5Ø64ØØ)

Command DFAC is used to set PD.DFAC in the eight Pulse Description (PD) sections of SCT.PD. PD.DFAC is the delta F factor for a given PD. The frequency of a given pulse is computed by adding the signed product, PD.DFAC * MD.DELF, to the base or ramp frequency of the pulse set. MD.DELF is set by command DELF (Section 2.7.7). (Also see PSCONT in NXTFRQ, Section 2.9.1.2.) The delta F factors are entered as signed decimal integers in the range -128 through +127; they are stored in PD.DFAC in 8-bit two's-complement form. If no sign ("+" or "-") is given, a value is assumed to be positive. From one to eight values may be entered as arguments to DFAC. If fewer than eight values are entered, the last value in the argument list is put into the remaining PDs. For example, the default sequence, DFAC Ø,1,1,Ø puts a 1 in PD.DFAC for PD 2 and PD 3, and puts zeroes into all the others. (See the default sequence in Table 2.1.)

The order in which the delta F factors are applied is controlled by command PSEQ (Section 2.7.4).

TXPH (SØ5Ø65ØØ-SØ5Ø91ØØ)

The TXPH command is used to specify special transmitters and transmitter polarization (PD.TXPH) for the eight PDs. Currently, only the MPI system has a need for this command, which has not as yet been implemented in the SPC loop.

From one to eight values may be entered; the range of the values is Ø-7, where Ø, 2, 4, and 6 select the HF Radar 1Ø kW transmitter. Values of 1, 3, 5, and 7 select the two MPI partial reflection 1ØØ kW transmitters and phasing, as follows.

1: TX 2 and TX 1 are fed in phase.
3: TX 2 leads TX 1 by 27Ø degrees.
5: TX 2 leads TX 1 by 9Ø degrees.
7: TX 2 leads TX 1 by 18Ø degrees.

If the low order bit of a command argument is set (i.e., the number is odd), and the 1ØØ kW transmitters are not known to DIOGEN, the TXPH command is rejected with ERR 18. The wide band, low power RF filters can be selected by BDW W (Section 2.7.1Ø).

If fewer than eight values are specified in the TXPH argument list, the last value specified is used for unspecified PDs, as explained in DFAC, above. Again, the order of transmitter selection and phasing is controlled by command PSEQ.

TSGC (SØ5Ø92ØØ-SØ5121ØØ)

Command TSGC is used to specify PD.TSGC, or the TSG command byte, for the eight PD sections of SCT.PD. (See Section 2.5 for the implications of TSG command byte selection.) From one to eight 8-bit hex values may be entered. Unless the high order bit of an argument is set, indicating a scenario request to the TSG, the TSGC command is rejected with ERR 1. (A command to the TSG with the high order bit Ø is a page read/write request.) If the command byte invokes scenario numbers 1, 3, 5, or 7, the TSGC command is rejected with ERR 4, unless the active sounding mode is W-class. This is because SOUNDER 2.Ø2 uses the specified scenario number, plus one, if a protected frequency is encountered during a sounding, often as the result of a ramp or delta F computation. (See PSCONT in NXTFRQ, Section 2.9.1.2.) W-class modes, although allowing odd-numbered scenarios, are prevented from selecting protected frequencies by disallowing commands RAMP and DELF. (The ramp is initialized null and delta F is initialized to Ø kHz for W-class modes.)

If fewer than eight values are entered, the last value in the argument list is used for all undefined PD.TSGC values. For example, TSGC A6 sets the PD.TSGC byte for all eight PDs to A6.

After setting the PD.TSGC bytes, TSGC loads overlay 1-Ø1 and calls TSGASS to set TSGC-associated values in the pulse palette. These are the TSG delay (PD.TDLY), the FEP control vector (PD.FEPC), and the FEP program starting address (PD.FEPA). These values are displayed by command PTP.

The order in which the TSG command bytes are sent to the TSG during a pulse set is controlled by command PSEQ.

Files TSGMEM.TWO and TSGMEM.TOO, the most commonly used zero-page images, are set up so that EQUALREF processing is invoked by scenario 2, SNDR by 4, and FEP PICKER processing by 6. TSGMEM.TOP, which must be used with FEPMEM.TOO, uses scenario 2 for PARTIAL processing.

ANTS (SØ5123ØØ-SØ5151ØØ)

Command ANT is used to specify the receiver input selection for a given PD in the pulse palette. Unlike the TSGC, DFAC, and TXPH commands above, the ANT command affects only a single PD. The ANT command is structured in this manner because the ANT argument list can be quite long and complex.

The ANT command normally has three arguments, the first of which is the PD number (1-8). The next two arguments normally specify the receiver inputs (1-4) for channels 1 and 2, respectively (PD.1RIN and PD.2RIN). ANT accepts one receiver input argument, if only one of the receiver inputs is configured pulse-to-pulse by DIOGEN (Section 1Ø.2.4). If two receiver inputs are specified, and only one is configured pulse-to-pulse, the ANT command is rejected with ERR 18.

The receiver input arguments may be simple (1-4) or if the receiver inputs are fed by outboard multiplexors, they may be complex. For example, ANT 3,1,1:2-4 means that for PD 3, receiver inputs 1 for both channels are selected, but the channel 2 input 1 is fed by outboard mux 2, for which input 4 is to be selected. An outboard mux can be fed by another mux, etc. If a contradictory or circular argument is given, the last specification in the argument is used. The same rules given for simple arguments in the above paragraph apply equally to complex arguments. If an outboard mux number (1-8) is referenced by the ANT command that has not been set up by DIOGEN, the command is rejected with ERR 18. Complex ANT arguments temporarily modify DIO pulse palette extensions the same way the UDIO command does. (See the notes under UDIO, Section 2.7.4.)

The order in which receiver and outboard mux inputs are selected during a pulse set is specified by the PSEQ command (Section 2.7.4).

RXANT (SØ5152ØØ-SØ5161ØØ)

The RXANT command is used to set MD.RXAN in the SCT. The number in MD.RXAN (1-1Ø) is the number of the record, or sector, of disc file SYS1:LOCAL.TWO that contains the description of the geometry of the receiving antenna array used by the sounding mode. This disc file is copied to magtape by MANAGER when the TAPE command is given. MD.RXAN * 256 is the byte offset to the RXA table in this tape file. (See LOCAL, Section 1Ø.2.2 and the RXA STRUC, Section 7.6.)

2.7.7. Overlay Ø-Ø6 (SNDROØØ6) [DCW]

Overlay Ø-Ø6 contains a collection of twelve SOUNDER command routines including those for Sweep-class base frequency selection. The RAMP command routine is also in this overlay.

EPF (SØ6Ø21ØØ-SØ6Ø27ØØ)

Command EPF sets echoes per frequency (MD.EPF) for the PICKER task. Although the EPF argument limits are 1-52, the ultimate restraint on EPF must be imposed by PICKER based on its available memory. (See MAX.EPF, Section 3.1.) Only the partial reflection configurations of PICKER (PICK4PPS, PICK5PPS, and PICK6PPS) allow 52 echoes per frequency. The standard PICKER allows up to 16. (See Section 3.14 for details.)

HPE (SØ6Ø28ØØ-SØ6Ø34ØØ)

Command HPE sets MD.HPE, 1/2-words per echo, explicitly for modes that employ complementary coding or that use TSG NOPs embedded within a pulse set. In such cases MD.PPS, the number of TSG starts per pulse set, is greater than the effective number of pulses per set, or the number of data-producing pulses. Because MD.HPE is also set implicitly by command PSEQ (Section 2.7.4), if used, command HPE must be given after PSEQ. The limits of the HPE command are 5-25, allowing for sets of from 1 to 6 pulses. The ultimate limit on 1/2-words per echo is in the PICKER task and is parameterized as MAX.HPE (Section 3.1).

CDN (SØ6Ø35ØØ-SØ6Ø41ØØ)

Command CDN is used to specify the countdown-to-sounding, in seconds (1-19), for MD.CDN. At sounding start MD.CDN is copied into ET.CDN, allowing for a partial second. (ET.CDN is -1 if no countdown is in progress. See CLOCK, Section 2.1Ø.1.)

KEEP (SØ6Ø42ØØ-SØ6Ø48ØØ)

KEEP specifies the M-out-of-N criterion (MD.M), which PICKER uses to determine what are range-coincident echoes. The upper limit on KEEP is the current value of MD.N, normally 8 for the 4-pulse pulse set, but is set to two times the number of pulses in the pulse set (MD.PPS) by PSEQ (Section 2.7.4). PSEQ also limits MD.M to MD.N, but KEEP functions just like P1.Ø KEEP if the pulse set size is held at 4.

KEEP applies to all sounding modes except G-, P-, and W-mode and their derivitives. (See Section 1Ø.2.1.) See Sections 3.12 and 3.13 for details about FEP EQR and FEP PICKER processing and their affect on KEEP.

CALA (SØ6Ø49ØØ-SØ6Ø56ØØ and SØ6Ø87ØØ-SØ61Ø8ØØ)

Command CALA is used to set the calibration attenuation (MD.CALA). The calibration pulse is a low level replica of the transmitter pulse normally injected at the receiving antennas as a part of each TSG timing sequence. The calibration attenuator is set in 8-dB steps from Ø-12Ø dB. If the attenuation specified by the CALA command is out of step, it is truncated to a multiple of 8, e.g., CALA 47 is taken as CALA 4Ø.

For installations that have the receiving antenna array at great distance from the central HF Radar hardware, CALA allows the specification of up to four frequencies at which to step the calibration attenuation -8 dB, in order to compensate for the cable losses, which are frequency dependent. From one to four frequencies may follow the base calibration attenuation in the CALA argument list. The frequencies, if specified, are converted to frequency indexes and are stored in order in MD.CSTP. The frequencies must be given in ascending order of kHz. MD.CSTP entries for which no frequency is specified are set to HIIX (SNDR, Section 7.7). Thus, if no frequencies are specified, the calibration attenuation is not stepped. (See PSCONT, Section 2.9.1.2.)

NFIL (SØ6Ø58ØØ-SØ6Ø63ØØ)

Command NFIL sets MD.NFIL, the number of TSG NOP fill starts (Ø-32767) issued between pulse sets during a sounding. NFIL is accepted regardless of the class of the active sounding mode and is used to guarantee enough time for pulse sets or to select the desired time-spacing between pulse sets at a given frequency. At the end of each pulse set, MD.NFIL is copied to NFILL by CKFILL (Section 2.7.1). NFILL is decremented until it becomes Ø by NXTFRQ (Section 2.9.1.2), at which time the next pulse set is begun. MD.NFIL TSG NOP starts are issued between all pulse sets, whether at a base or a ramp frequency. For T- and W-class sounding modes, NFIL is the exact equivalent of the P1.Ø K-mode command, KFIL. The time value of a TSG NOP fill is that of the most recent TSG start that produced data, normally that of the final pulse of the previous pulse set.

NSET (SØ6Ø64ØØ-SØ6Ø7ØØØ)

NSET specifies the number of passes to be made through the base frequency selection. The argument of the NSET command (1-32767) is put into MD.NSET. NSET can be specified for a sounding mode of any class. For an S-class sounding mode, a pass through the base frequency selection is from SFRQ through EFRQ (below). For example, multiple ionograms can occupy the same data set with no time lost between ionograms. For a T- or W-class sounding, a pass through the base frequency selection is a pass through SCT.FTBL, sounding in order at each active frequency (Section 7.4). MD.NSET is copied to NRND at the beginning of a sounding. NRND is decremented at the end of each pass through the base frequency selection and when Ø, the sounding is finished. For T- and W-class sounding modes, NSET is the exact equivalent of the P1.Ø K-mode command, KSET.

BHCOM (SØ6Ø71ØØ-SØ6Ø85ØØ)

Routine BHCOM is common to the above single-argument SOUNDER commands. BHCOM is entered with the address of the SCT parameter in R3, the lower and upper limits of the command argument in R4 and R5, respectively, and a byte flag (Ø) or 1/2-word flag (1) in R6. BHCOM gets the command argument, checks it against the limits in R4 and R5, and sets the byte or 1/2-word field in the SCT based on R6, unless the argument is out of bounds, is which case, ERR 1 results.

SFRQ and EFRQ (SØ611ØØØ-SØ6145ØØ)

Commands SFRQ and EFRQ set the lower and upper limits of a Sweep-class sounding. These commands are rejected with ERR 14 unless the active sounding mode is S-class. These two commands execute common code at FRCOM. FRCOM makes certain that the required protected frequencies map is in FRQMAP and honest (Section 2.6.4), loads overlay 1-Ø2, and calls subroutine GETF to get the starting or ending frequency of the sweep from the command argument list. This frequency is checked by MAPFRQ, and if it is protected, it is adjusted as are T- and W-class frequencies (Section 2.7.5). If SFRQ as specified is found to be protected, it is adjusted upward in

1ØØ Hz steps until an unprotected frequency is found. If EFRQ is protected as given, it is similarly adjusted downward until an unprotected frequency is found. The 32-bit frequency is saved in MD.SFRQ or MD.EFRQ in Hz/1ØØ, and the corresponding frequency index is saved in MD.SFIX or MD.EFIX.

STEP (SØ6146ØØ-SØ6157ØØ)

Command STEP is used to specify the frequency or frequency index step for an S-class sounding mode sweep. The STEP command is rejected with ERR 14 unless the active mode is S-class. STEP enters the RAMP command routine at STEPC (below) and accepts a frequency index step (1-3275) or a frequency step (FØ.1-F3275.Ø) for MD.STEP. A frequency step is stored as Hz/1ØØ. The "F" is required only to specify a linear frequency step. The logarithmic frequency index step is expressed as an integer, as was DFIX in P1.Ø. The same holds true for the RAMP step, described below.

RAMP (SØ6159ØØ-SØ6219ØØ)

The RAMP command is used to specify the dimensions of either a B- or a Z-ramp sequence for all but W-class sounding modes. For W-class modes, RAMP results in ERR 14. (See TSGC, Section 2.7.6.) The three dimensions of a ramp sequence are (in the order typed): the number of ramps on each base frequency (MD.NRAM), the number of steps in each ramp (MD.SUBN), and the basic ramp step (MD.SUBS). The ramp type, B or Z, is flagged in MD.CTRL (Section 7.1).

The B-ramp is derived directly from the ramp dimensions. For example, RAMP 6,5,1,B produces 6 identical ramps, each with 5 steps of 1 increment of frequency index. If a given base frequency is numbered 1, the ramp, repeated 6 times at this base frequency, would be 1 2 3 4 5. The Z-ramp derivation, more complex, is described in detail in Section 2.7.1.

If no arguments follow the RAMP command, any existing RAMP dimensions are cancelled, exactly as if the command, RAMP 1,1,Ø,B, were given. This results in what is termed a null ramp. Otherwise, all RAMP arguments must be specified, including the ramp type, "B" or "Z".

At STEPC, the argument list is checked for the linear frequency ramp step flag, "F". If the "F" is present, the argument following is a ramp step (or S-class STEP, above) specified as a frequency in kHz. Otherwise, the ramp step (or

STEP) is taken to be a step in frequency index. STEPC flags MD.CTRL (Section 7.1) accordingly.

DELF (SØ622ØØØ-SØ6226ØØ)

Command DELF is used to set MD.DELF, the delta frequency, for all but W-class sounding modes. DELF results in ERR 14 for W-class modes. (See TSGC, Section 2.7.6.) For all but W-class modes, DELF is specified in kHz and is stored in Hz/1ØØ. Delta F is multiplied by the delta F factor (DFAC, Section 2.7.6) for each pulse of a pulse set. This signed product is then added to the base or ramp frequency of the pulse set to determine the frequency to which the RF system is tuned for a given pulse.

2.7.8. Overlay Ø-Ø7 (SNDROØØ7) [DCW]

The first three command routines in overlay Ø-Ø7, those for SPH, SN, and NO (DEBUG), are indifferent to what sounding mode is active when the commands are given. CALL, and DESC both set 8-byte ASCII fields in the SCT and are sounding mode dependent. Command MDIR lists the directory of sounding modes.

SPH (SØ7Ø2ØØØ-SØ7Ø33ØØ)

Command SPH is used to specify the number of soundings to run per hour (SCT.SPH). SPH, by itself, does not imply a particular sounding mode; an SPH series of soundings is initiated by RUN for whatever mode is specified by the RUN command. The argument of the SPH command is 1-255 or 999. If 999 (free run) is specified, SCT.SPH is cleared to Ø, and after RUN is given, soundings are run back-to-back as close together as MD.CDN (Section 2.7.7) allows. If 1-255 is specified, the time between sounding starts is computed as 36ØØ/SCT.SPH (truncated integer). If the sounding mode specified by RUN cannot run to completion within the time allotted by the SPH schedule, an error message is logged, and the free run SPH schedule is resorted to. (See CLOCK, Section 2.1Ø.1.)

An SPH schedule is terminated when AB is typed or when SN (below) is satisfied. An SPH schedule terminated with AB can be restarted with CONT (Section 2.7.2). Whenever the SPH command is given, NEXT (Section 2.6.4) is zeroed to disable the CONT command. CONT remains disabled until the new SPH schedule is aborted.

If SOUNDER is biding time until the next sounding of an SPH schedule is due to start, "B1DE" is displayed in the hex panel lights.

SN (SØ7Ø34ØØ-SØ7Ø43ØØ)

Command SN sets SCT.SN, the number of soundings to be run (1-32767), and resets SCT.SNO, the sounding count, to Ø. SN also clears SNOLD to Ø. (See GETADV, Section 2.6.1.) SN is usually given in conjunction with SPH (above) so that an SPH series will terminate when an unattended magnetic tape is expected to be full.

SCT.SN is initialized to -1 at SOUNDER start up, and is ignored as long as it remains less than Ø. If SCT.SN is set to 1-32767, it is compared to SCT.SNO at the end of every sounding. (SCT.SNO is incremented at the start of each sounding by CLOCK, Section 2.1Ø.1.) When SCT.SNO = SCT.SN, SN is said to be satisfied, and SCT.SN is reset to -1. Control returns to the IMA loop, and the SPH schedule, if active, is terminated. (The end of sounding is described in Section 2.9.1.1.)

DEBUG (SØ7Ø44ØØ-SØ7Ø57ØØ)

Routine DEBUG is entered when the NO command is given to allow temporary debugging without the TSG, FEP, EKO, or HKADC. Command NO accepts the arguments "T", "F", "E", and "A" for the above devices. Any or all of the arguments can be given and in any order. NO F has no effect, because SOUNDER 2.Ø2 no longer has explicit control over the FEP. (See PIPLN, Section 1Ø.2.15.) The NO command has its own argument scan list. This list determines the order of the device-is-dead flags in SNDRTBLS. (See NOxxx, Section 2.6.4.)

CALL and DESC (SØ7Ø59ØØ-SØ7Ø88ØØ)

Commands CALL and DESC share routine CDCOM, which copies up to 8 ASCII characters from the command argument to MD.CALL or MD.DESC. If a carriage return character is seen before 8 characters are copied, the field in question is blank filled.

Command CALL specifies a CALL task to occupy GRAPHER's memory partition at run time. The existence of the CALL task is checked when the sounding mode is run, not when the CALL command is given. (See Section 1Ø.3.)

DESC specifies a short (8-character) mode or experiment description. This description (MD.DESC) is displayed on the system CRT by GRAPHER at run time.

MDIR (SØ7Ø9ØØØ-SØ7175ØØ)

Command MDIR lists the sounding mode directory from SYS1:SMODES.xxx to LU1, normally the console (CON:). The sounding modes are listed, with date-markings and run counts, in the order in which they appear in the directory, with the permanent modes (Section 1Ø.2.1) listed first. The first permanent mode listed is the active mode after HF Radar start up.

Non-printing control characters in user-defined mode names are replaced with asterisks. (Note that the asterisk character (*) cannot be used when assigning a mode name. See MSAV, Section 2.7.2.)

UNPK is a revision of a subroutine written by J. R. Winkelman that unpacks a 16-bit day-year word into an ASCII string of the form, MM/DD/YY. The day-year word is packed like TODAY (Section 2.6.4).

2.7.9. Overlay Ø-Ø8 (SNDROØØ8) [DCW]

The Automatic Gain Control commands, AGC, PKLO, and PKHI, are together in overlay Ø-Ø8. This overlay is also loaded when commands TICK and PTP are given, but these commands also require other overlays.

AGC (SØ8Ø21ØØ-SØ8Ø84ØØ)

Command AGC specifies gain control for receiver channels 1 and 2. Gain control can be either automatic or manual, and the receiver channels are controlled independently. Channels under manual gain control can have their RF and IF attenuators set by console commands during sounding. (See RFAT and IFAT, Section 2.6.1.) Channels under AGC are unaffected by commands RF and IF; their attenuators are set dynamically in SOUNDER's SPC loop during sounding. The dynamic range limits for a channel under AGC are set by commands PKLO and PKHI, below.

2.7.9. Overlay Ø-Ø8 (SNDROØØ8)

The first argument given the AGC command (Ø-3) is saved in MD.AGC and specifies the type of control, as follows:

Ø: Both channels under manual control.
1: Channel 1 automatic, channel 2 manual.
2: Channel 1 manual, channel 2 automatic.
3: Both channels under AGC.

The first AGC argument determines the maximum number of arguments that may follow. For each channel declared to be under AGC, two additional arguments are accepted. Any or all of these additional arguments may be omitted, and if given, may be given in any order. One of the additional arguments is the base RF attenuation (Ø-6Ø dB), the other, flagged with "P", is the number of pulses after which to update the RF and IF attenuation bytes. If too many of either kind of these arguments is given, that is, if a channel under AGC is over-specified, the AGC command is rejected with ERR 17. If both channels are under AGC, and if two arguments of either kind are given, the first encountered is taken for channel 1, the second for channel 2. If both channels are under AGC, and if only a single argument of a given kind is specified, it applies to both channels. For example, AGC 3,12,P4,16 sets the base RF attenuation for channel 1 to 12 dB and the base RF for channel 2 to 16 dB. The attenuation bytes for both channels are updated every 4 pulses.

The "P"-flagged argument sets MD.PPU, but should not be used, because it is not fully implemented. If this argument is omitted, MD.PPU defaults to MD.PPS, the number of pulses per pulse set. (This is the way the AGC was handled in P1.Ø.)

The base IF attenuation is set 2Ø dB below the RF if the RF is 24-6Ø dB. Otherwise the base IF is set to Ø dB. If the base RF is specified to be greater than 6Ø dB, both the base RF and IF are set to 6Ø dB to allow testing at full attenuation. The base attenuations are truncated to multiples of 4, that is, 11 is taken as 8 dB. The base RF and IF attenuations for both channels are stored in MD.BAG in the order, RF1, IF1, RF2, IF2.

At the beginning of each S-class sounding, SWATTN (Section 2.6.4) is initialized from MD.BAG. The appropriate bytes in SWATTN are updated during sounding by UPAGC (Section 2.9.1.3), which does not allow the attenuation levels in SWATTN to go below those specified in MD.BAG.

2.7.9. Overlay Ø-Ø8 (SNDROØØ8)

For T- and W-class sounding modes, FR.ATTN (Section 7.4) is set from MD.BAG whenever a table frequency is set or adjusted (Section 2.7.5). For this reason, if base attenuations other than the default Ø are desired, the AGC command should be given before the F command. During a sounding, the bytes in FR.ATTN for each SCT.FTBL entry are adjusted like SWATTN (above), but on an SCT.FTBL entry basis. The attenuation settings in each FR.ATTN are carried over from sounding to sounding for a given mode, unless reset to MD.BAG by changing the frequency setting with some form of command F.

PKLO and PKHI (SØ8Ø86ØØ-SØ8124ØØ)

Commands PKLO and PKHI set the limits, in millivolts, of the dynamic ranges of the receiver channels under AGC. The minimum PKLO is 1ØØØ mV; the maximum PKHI is 1ØØØØ mV. The minimum PKHI - PKLO is 1ØØØ mV, although PKHI should be set to at least 1.6 times PKLO. This is because the minimum attenuation adjustment is 4 dB; setting PKLO and PKHI too close together will cause the attenuators to toggle unnecessarily between two adjacent settings. These commands each accept one or two arguments. If a single argument is given, it applies to both receiver channels; if two arguments are given, they apply in order to channel 1 and channel 2.

The PKLO arguments are saved in MD.PKLO, PKHI in MD.PKHI, in channel 1, channel 2 order. When command RUN or CONT is given, MD.PKLO and MD.PKHI are converted from mV to FEP ADC counts and are stored in LOAGC and HIAGC, respectively. These values are the AGC limits used by UPAGC (Section 2.9.1.3) for receiver channels not under manual gain control.

TICK (SØ8126ØØ-SØ8135ØØ)

Command TICK, which times the four selectable TSG paces, causes overlay Ø-Ø8 to be loaded, but the main code for this command is in overlay 2-Ø1 (Section 2.9.2.1).

TICK loads 2-Ø1, then calls subroutine PACE, which times the four TSG paces. After PACE has executed, TICK loads 2-Ø2, then calls subroutine SYSTAT, which produces the SYS display (Section 2.9.3.1).

PLIST (SØ8137ØØ-SØ8168ØØ)

Routine PLIST is entered when command PTP is given to display the pulse palette. PLIST loads overlay 2-Ø2 for the display subroutines, PAGE and WRCON (Section 2.9.3.2), then loads 1-Ø4, which contains most of the PTP command routine. The loop for displaying the 8 pulse descriptions starts at PDLP in this overlay. The loop continues at PCON1 in 1-Ø4; PCON1 then returns to PDLP for the next pulse description display.

The TSG command byte (TSGC) is broken down into the time-value of the TSG pace (TIK), the FEP coherent averaging repeat count (REP), and the TSG timing scenario number (SCN). Also displayed are the TSGC-associated values. These are the range of the start of the FEP WRITE ENABLE window, in km (FEPR), the FEP program starting address (FEPA) in hex, and the FEP control vector (FEPC). FEPC is 52 for 1Ø microsecond FEP ADC sampling and 12 for 2Ø microsecond sampling. FEPR, as displayed, is only correct if the calibration pulse follows the TX KEY. The FEP WRITE ENABLE window is assumed not to be split for the FEPR display. The TSG delays in TSGDLY and PD.TDLY, however, are correct for split windows. (See SNRIO, Section 1Ø.2.1Ø.)

DFAC is the delta F factor, which is multiplied by the delta frequency (MD.DELF) to compute the offset from the base or ramp frequency of a given pulse set. RX1 and RX2 are the channel 1 and 2 inputs. TXPH, the special transmitter and polarization specification; and SPIX, the special routine index, are not fully implemented.

2.7.1Ø. Overlay Ø-Ø9 (SNDROØØ9) [DCW]

Overlay Ø-Ø9 contains the BDW command routine and part of the MGET display routine.

BDW (SØ9Ø19ØØ-SØ9Ø7ØØØ)

Command BDW is called to specify the receiver bandwidths in kHz, and to specify whether the narrow or wide band low power RF system filters are to be selected for a sounding mode. Only the SEL/NSF and MPI systems have these filters installed to date. These specifications, in DIO command form, are saved in MD.BDW.

2.7.1Ø. Overlay Ø-Ø9 (SNDROØØ9)

If command BDW is given with no arguments, the current receiver bandwidths are preserved, and the narrow band RF filters are selected when sounding begins. If BDW W is given, the bandwidths are preserved, but the wide band filters are selected, allowing for 3Ø microsecond transmitter pulses. The BDW command also accepts one or two numeric arguments, the receiver bandwidths, in kHz. If only one argument is given, it is taken as the bandwidth of both receiver channels. If both are given, they apply in order to channels 1 and 2. If "W" follows the bandwidths, the wide band RF filters are selected; otherwise the narrow band filters are specified. Valid bandwidth specifications are 1, 3, 1Ø, and 3Ø kHz. These are converted to DIO commands Ø, 1, 2, and 3, respectively, and are stored in the high order 4 bits of MD.BDW in channel 1, channel 2 order. If the wide band RF filters are specified, MD.BDW is flagged with 8. The low order 3 bits of MD.BDW are unused and are always Ø.

MCON1 (SØ9Ø72ØØ-SØ9138ØØ)

When command MGET is typed, the sounding configuration of the mode specified is displayed. Assembly language display routines are notoriously inefficient in terms of memory usage; this is the second of four overlays invoked by the MGET command. The rule in SOUNDER is that display routines are of low priority and will be seldom modified; they are used to fill gaps in overlays.

If MCON1 is modified, notice that MCON1 is entered from MLIST in 1-Ø1 with RØ-R15 saved in EREGS. MCON1 in turn loads 1-Ø2 to display T- and W-class frequencies, if necessary, then after displaying the BOOT DIO,xxx line, loads 1-Ø5 and continues at MCON2.

2.8. Overlay Area 1 [DCW]

Overlay area 1, like areas Ø and 3, is 512 bytes, or 2 disc sectors long. One of the overlay area 1 overlays is a primary command routine overlay (1-Ø3), and all are loaded from primary command routines, usually from overlay area Ø. One of six current overlays can occupy area 1. These overlays, 1-ØØ through 1-Ø5, are described below in order in Sections 2.8.1 through 2.8.6.

2.8.1. Overlay 1-ØØ (SNDRO1ØØ) [DCW]

Commands RUN and CONT cause SNDOV (Section 2.6.1) to load overlay 1-ØØ. Also loaded at this time are Ø-ØØ, 2-ØØ, and 3-ØØ. These overlays remain in memory for a single sounding or for a series of soundings, depending on SPH and SN (Section 2.7.8).

Overlay 1-ØØ serves two purposes. After being loaded by SNDOV, subroutine RDTIME (below) is called to read the TSG clock. After the TSG clock has been read and SOUNDER's time-keeping initialized, SOUNDER cannot load overlays, or use the disc in any way, until there is an hiatus in sounding, because during sounding, MANAGER must have total control of the disc in order to keep disc track seeks at a minimum, etc. Furthermore, reading the disc might introduce an uncertainty in time that would confound SOUNDER's precise time-keeping.

After RDTIME has executed, 1-ØØ has served its first purpose. SNDOV marks the overlay area 1 entry in NOV to force 1-ØØ to be reloaded by the next RUN or CONT command. (See, Get Overlays, Section 2.6.1.) Overlay 1-ØØ is thus prepared to serve its second purpose, that of serving as a buffer area for House-keeping ADC (HKADC) and DIO readings. The address of this buffer (HKDIO) is put into ET.AHKD (Section 7.3) for MANAGER at SOUNDER start up. MANAGER writes this buffer to disc and tape, if selected, at sounding end.

HKDIO is organized as follows. Bytes Ø and 1 contain the buffer size, in bytes, in the low order 12 bits; the size is flagged with hex A in the upper 4 bits to identify the record as the HKDIO buffer. The size of HKDIO for a normal sounding is 258 bytes, but is set to 2 for a debugging TPR run. (See Section 2.7.3.) Bytes 2 through 65 contain the latest readings from HKADC channels Ø-31, copied at sounding start from ET.HKADC (Section 7.3) by NXTSND (Section 2.9.1.1). The user is advised to connect critical sensors to HKADC channels Ø-31. These channels, together with channels 32-63, are read again at sounding end by HD.LP (Section 2.9.1.1). These readings of the 64 HKADC channels are put into bytes 66-193 of HKDIO. HD.LP also reads DIO ports Ø-63 into bytes 194-257 of HKDIO. The data from DIO Ø should be ignored, because DIO Ø is hardware disabled. The HKDIO data for a given sounding are displayed by CALL task HKDIO (Section 1Ø.2.19).

Off-line analysis programs thus have an overall perspective of all 64 HKADC channels and 63 DIO ports. Because the critical HKADC channels, Ø-31, are read and recorded both before and after sounding, the off-line program is made aware of certain hardware failures that may have occurred during sounding and that may have affected the data. The above assumes that the off-line program is informed about how the HKADC sensors and DIO devices are connected. These assignments may be included in the LOCAL file by a user-supplied mechanism.

OV1 (S1ØØ22ØØ)

Just as OVØ (Section 2.7.1) establishes the address of overlay area Ø for the overlay loader (Section 2.6.1), so OV1 makes global the address of overlay area 1.

HKDIO (S1ØØ23ØØ-S1ØØ24ØØ)

Label HKDIO establishes the address of the buffer for the HKADC and DIO readings for ET.AHKD and the routines in SNDRPIPE, discussed above.

RDTIME (S1ØØ25ØØ-S1Ø119ØØ)

Subroutine RDTIME is called by SNDOV (Section 2.6.1), after command RUN or CONT is given, to read the TSG clock, and from this reading, to initialize SOUNDER's time-keeping counters, discussed below.

The method of RDTIME is to read the TSG clock repeatedly with subroutine RDCLK (below) until the tenths of seconds changes. When the tenths of seconds changes, the TSG clock is immediately read again to confirm the change. If this reading shows no change, the TSG clock has been successfully read to the nearest tenth of a second; otherwise the process is repeated. After a successful clock read, RDTIME converts the twelve BCD time elements (Table 2.5, below) to seconds since midnight, 1/1/77 (ET.SEC, Section 7.3), current hour of the day (HOUR, Section 2.6.4), seconds into the current hour (HRSEC), and the hundredths of seconds into the current second (FRSEC). Subroutine CLOCK (Section 2.1Ø.1) maintains these counters during sounding and between soundings during an SPH schedule. The basic unit of time in SOUNDER is 1Ø ms, or 1/1ØØth of a second.

The TSG clock is read in this manner at SOUNDER start up (Section 2.6.5) and whenever commands RUN and CONT are given. If the TSG clock has not been configured into the DIO software system, RDTIME sets the clock counters to midnight, 1 January 1977. (See DIOGEN, Section 1Ø.2.4.) If RDTIME fails to successfully read the TSG clock in 1Ø attempts, ERROR 8226 is logged, and the time is set to 1/1/77, as above.

RDCLK (S1Ø121ØØ-S1Ø195ØØ)

Subroutine RDCLK is called by RDTIME (above) and by SETIME (Section 2.6.5) to make one reading of the 12 BCD elements of TSG time. The 12 elements are read in order (Ø-11) by first writing the element number to the TSG clock DIO address (select), then reading the selected BCD element data together with its number. If the element number selected does not match the element number read, zero is used for that element, and an error message is logged that displays the good and the bad element numbers. A TSG clock read error causes a call to DEVERR (Section 2.6.2) to count the error and to flag it in MD.STAT (Section 7.1). SOUNDER continues executing after a TSG clock read error. Because the TSG clock is functionally separate from the TSG itself, soundings can be run in spite of a malfunctioning TSG clock; only the sounding start and end times will be in error. The twelve BCD TSG time elements are listed in Table 2.5.

CODE	BCD TIME ELEMENT
Ø	Tenths of seconds.
1	Units of seconds.
2	Tens of seconds.
3	Units of minutes.
4	Tens of minutes.
5	Units of hours.
6	Tens of hours.
7	Units of days.
8	Tens of days.
9	Hundreds of days.
A	Units of years.
B	Tens of years.

Table 2.5. TSG CLOCK BCD TIME ELEMENTS.

OVSZ (S1Ø197ØØ-S1Ø2Ø2ØØ)

Like OVSZ in Ø-ØØ, this OVSZ forces the size of overlay area 1 to be 512 bytes, or an even 2 disc sectors, in length.

2.8.1. Overlay 1-ØØ (SNDRO1ØØ)

2.8.2. Overlay 1-Ø1 (SNDRO1Ø1) [DCW]

Overlay 1-Ø1 contains routine MLIST and subroutines that support the mode-handling command routines. All SYS1:SMODES.xxx I/O is done in 1-Ø1.

MLIST (S11Ø21ØØ-S11Ø46ØØ)

Routine MLIST is entered from the MGET command routine in overlay Ø-Ø1. MLIST is the beginning of the sounding mode configuration display routine. MLIST loads Ø-Ø9, where the display routine continues, then loads 2-Ø2, which contains the display subroutines PAGE and WRCON (Section 2.9.3.2) and the final segment of the display code. MLIST saves RØ-R15 in EREGS, writes the MGET header line to LU1 (normally CON:), then continues at MCON1 in Ø-Ø9 (Section 2.7.1Ø). MCON1 loads 1-Ø5, MCON2, which in turn goes to the final segment of the MGET display routine, MCON3 in 2-Ø2.

TSGASS (S11Ø48ØØ-S11Ø72ØØ)

Subroutine TSGASS is called to set values PD.TDLY, the time delay between the transmitted pulse and the FEP data acquisition window; PD.FEPC, the FEP control vector; and PD.FEPA, the FEP program starting address, for each of the 8 pulse descriptions in the pulse palette (SCT.PD, Section 7.5). These values are all associated with, or implied by, the TSG command byte (TSGC, Section 2.7.6). PD.TDLY is from table TSGDLY (Section 2.6.4). The values in TSGDLY are computed from the TSG zero-page scenario tables by subroutine TDLAYS (Section 2.1Ø.3) whenever the TSG is bootstrapped. Values PD.FEPC and PD.FEPA are copied from table ZPCTRL (Section 2.6.4), which is also set up by TDLAYS.

TSGASS is called whenever the TSG zero-page changes (by BOOT TSG or by mode switching) and whenever the TSG command byte sequence is changed by command TSGC.

SETMID (S11Ø73ØØ-S11Ø98ØØ)

Subroutine SETMID is called by the mode-handling overlay Ø-Ø1 (Section 2.7.2) and by the MDIR command routine (Section 2.7.8) with the binary image of a mode name in R5. SETMID puts R5 in MODIM, and converts MODIM to printable form, if necessary, for MODID (Section 2.6.4). Any non-printing characters are changed to asterisks for MODID, and the second character of a permanent mode name, always a carriage return (hex D), is changed to a space (hex 2Ø). MODIM comes directly from the mode directory; MODID is the printable version of MODIM used by the MGET, PTP, and MDIR displays. MODID is also the version put into MD.MREV for an SCT to be recorded with a data set.

<u>RDDIR</u> <u>and</u> <u>WRDIR</u> (S111ØØØØ-S11111ØØ)

RDDIR and WRDIR are special entries to subroutines RDMODE and WRMODE called to read and write the sounding mode file directory. R5, the record or relative sector number of the sounding mode, is set to Ø, the record number of the directory.

<u>RDMODE</u> <u>and</u> <u>WRMODE</u> (S11112ØØ-S11162ØØ)

Subroutines RDMODE and WRMODE are called to read sounding modes from SYS1:SMODES.xxx and to write them back, in order to retain any changes made to the sounding mode configuration: changes made as the result of operator commands, recalculation of RF and IF attenuation bytes by the AGC system for T- and W-class modes, or in future, by ANALYSER tasks that "tune" a sounding mode configuration.

RDMODE and WRMODE share routine RWCOM. RWCOM is entered with a read or write function code in RØ, which is put into the RWMODES SVC1 parameter block, and with the first record number of the mode or mode directory in R5. If R5 = Ø, indicating mode directory I/O, the data transfer is set up using ET.ABUF (Section 7.3) as the buffer for the mode directory. Otherwise, the mode-dependent section of the SCT, SCT.MD-SCT.DISC (Section 7.1), is used. SOUNDER does not expect the directory to remain in ET.ABUF after the execution of a mode-related command. The directory is written back to disc immediately.

After reading a new mode from disc, RDMODE compares MD.MREV with SCT.MREV. If the two values are not the same, the new mode is not compatible with the current revision of SOUNDER's SCT. The mode just displaced is reinstated, and ERR 13 is logged.

Sounding modes and the mode directory occupy two disc sectors. If these two sectors were contiguous on the disc, the disc would have to make two revolutions in order for the data transfer to be completed. Therefore, sounding modes and the mode directory occupy sectors n and n + 4, allowing both sectors to be read or written in one revolution of the disc, thus speeding up mode switching.

2.8.3. Overlay 1-Ø2 (SNDRO1Ø2) [DCW]

Overlay 1-Ø2 contains subroutines that are called mainly by the Table-class command overlay, Ø-Ø4 (Section 2.7.5). Subroutines GETF and GETDF are called by all command routines that accept a frequency in their argument list. These are the only subroutines in SOUNDER that perform this function.

DSPALL (S12Ø22ØØ-S12Ø37ØØ)

Subroutine DSPALL is called by the F command routine (Section 2.7.5) and by the MGET display routine (Section 2.7.1Ø) to display all active entries in the frequency table, SCT.FTBL (Section 7.4). For each active T- or W-class frequency, DSPALL calls subroutine DISPTF (below) to display the frequency and its associated echo range limits.

DISPTF (S12Ø38ØØ-S12Ø55ØØ)

Subroutine DISPTF is called by DSPALL (above) and by commands Fn and RNGEn to display an SCT.FTBL entry, n, if active, and to display that entry whenever modified. DISPTF converts FR.RMIN and FR.RMAX (Section 7.4) from microseconds to kilometers for the display, and calls QSØ8 (below) to send the SØ8 Q-block to GRAPHER.

QSØ8 (S12Ø57ØØ-S12Ø74ØØ)

Subroutine QSØ8 is called by DISPTF whenever an SCT.FTBL entry is displayed, and is called by the command routine for FnX, which deletes a T- or W-class frequency, or deactivates an SCT.FTBL entry. QSØ8 sends the SØ8 Q-block (Section 7.8.2) to GRAPHER with the frequency number, or the SCT.FTBL entry number in SØ8 + 4. GRAPHER responds by displaying or deleting an echo range bar.

GETF and GETDF (S12Ø76ØØ-S12126ØØ)

Subroutines GETF and GETDF are called by routines of all SOUNDER commands that contain frequencies in their argument lists. GETF is called by commands SFRQ, EFRQ, CALA, and F. The frequency limits for GETF are LOFR and HIFR (SNDR, Section 7.7). GETDF is called by commands DELF, RAMP, and STEP, all in Ø-Ø6 (Section 2.7.7). The frequency limits for GETDF are SNDR parameters LODF and HIDF.

GETF and GETDF share routine FCOM. FCOM scans the command argument list for a frequency. All frequencies input to SOUNDER commands are expressed in kHz, with an accuracy of Ø.1 kHz. All frequencies are stored in integer form as Hz/1ØØ, or kHz*1Ø. By no coincidence, the command resolution of the frequency synthesizer is also Ø.1 kHz. FCOM accepts any form of a frequency, provided it is within the limits set by GETF and GETDF. For example, Ø.1, .9, 1234, 1234., and 1234.5 are acceptable. A frequency of the form 1234.567 is truncated to 1234.5. All frequencies are returned to the caller of GETF or GETDF as 32-bit, Hz/1ØØ values in the reqister pair, R12,R13.

CKFTBL (S12128ØØ-S12186ØØ)

Subroutine CKFTBL is called whenever command RUN or CONT invokes a T- or W-class sounding mode (Section 2.7.2) and whenever a table frequency is set, adjusted, or deleted (Section 2.7.5). CKFTBL scans SCT.FTBL (Section 7.4) and finds the lowest and highest frequencies selected. The lowest frequency and its corresponding index are put into MD.SFRQ and MD.SFIX. Similarly, the highest frequency selected sets MD.EFRQ and MD.EFIX. This allows GRAPHER and off-line programs to plot T- and W-class sounding data in S-class form. Each frequency selected is again checked against the protected frequencies map. Any frequency found to be protected at this time is deleted. The operator is not informed. This is part of a deeply-imbedded system to prevent sounding on protected frequencies, whether by accident or by design. The address of the first active SCT.FTBL entry is put into A1STF and AFTF, in anticipation of sounding. (See NXTBAS, Section 2.7.1.) The number of active frequencies, or active SCT.FTBL entries, is put into MD.NTFR, and is returned to the caller of CKFTBL in R3. If R3 = Ø, the RUN or CONT commands are rejected with ERR 5 (Section 2.7.2).

2.8.4. Overlay 1-Ø3 (SNDRO1Ø3) [DCW]

Overlay 1-Ø3 swaps GRAPHER for a CALL task when RUN <task> is given or if command CALL has specified a CALL task for run time. (See Sections 1Ø.2 and 1Ø.3.) This overlay also responds to all queues from CALL tasks, and reinstates GRAPHER after a CALL task has executed. 1-Ø3 also contains the PIK command routine and subroutine GTPICK, called to swap versions of the PICKER task as required by different sounding modes.

RUNCALL (S13Ø23ØØ-S13Ø32ØØ)

RUNCALL is entered from Ø-Ø1 (Section 2.7.2) when the RUN command argument has more than 2 characters, indicating that a CALL task name, not a mode name, is specified. If CALF (Section 2.6.4) indicates that a CALL task is already active, the RUN <task> command is rejected with ERR 1Ø. Otherwise, RUNCALL calls subroutine GTCALL (below) which, with the help of other subroutines, cancels GRAPHER and loads and starts the Idle-mode CALL task (Section 1Ø.2). RUNCALL saves the address of the CØ2 Q-block, which informed SOUNDER of the RUN command, in CALQ (Section 2.6.4), and exits the ADVICE subroutine (Section 2.6.1) without clearing the CØ2 status. The CALL task thus usurps control ot the system console and can accept command input directly and can present its own tabular and graphic displays without COUNSEL intervention. When the CØ2 Q-block status is cleared by ENDCALL after the CALL task executes (below), the CALL task abdicates control of the console.

ENDCALL (S13Ø33ØØ-S13Ø56ØØ)

ENDCALL is entered from WTPOST (Section 2.6.2) to reinstate GRAPHER after a Sounding-mode CALL task has executed (Section 1Ø.3). ENDCALL also is entered from ANSCALL (below) to swap in GRAPHER after an Idle-mode CALL task has requested termination. (See Section 1Ø.2.)

After canceling the CALL task and loading GRAPHER into its memory partition, ENDCALL starts the GRAPHER task and sends it a CØØ Q-block, like COUNSEL does at HF Radar start up, except that Ø is put into CØØ + 4 instead of the address of the ET. This signals GRAPHER that it is being restarted rather then being started for the first time.

ENDCALL reinitializes SØ2LST (Section 2.6.4) to cycle through all of SOUNDER's buffers, in case the CALL task just terminated used them. (See GTCALL, below.)

After reinstating GRAPHER, ENDCALL checks CALQ (above) to see if RUNCALL deferred clearing the CØ2 Q-block status. If the status clearing was deferred, ENDCALL goes to OKADV (Section 2.6.1) where the status is now cleared, returning control of the console to COUNSEL. (Anyone who learns to say this last phrase ten times without error wins a free HF Radar.)

2.8.4. Overlay 1-Ø3 (SNDRO1Ø3)

GTCALL (S13Ø57ØØ-S13Ø72ØØ)

Subroutine GTCALL is called by RUNCALL to cancel GRAPHER and to load and start a CALL task. GTCALL calls the general purpose task swapper, SWPTSK (below), which sets up the actual swap by SWAP, also below. Before GRAPHER is canceled, it is sent an MØ4 Q-block (Section 7.8.4) to terminate the system CRT I/O. After the CALL task has been started, it is queued with the address of CALLST, which contains the address of the ET (Section 7.3), the number of SOUNDER PCT/EKO buffers initially available to the CALL task (currently 4), and the address of SOUNDER's SØ2LST (Section 2.6.4). The CALL task thus has access to the HF Radar global tables, as well as to SOUNDER's buffers. (For an example, see PLOTEKO, Section 2.1Ø.7.) GTCALL initializes SØ2LST to cycle through 4 buffers, in case the CALL task needs them.

SWPTSK (S13Ø74ØØ-S131Ø8ØØ)

SWPTSK is the general purpose task swapper that is called to swap tasks in both PICKER's (GTPICK, below) and GRAPHER's (GTCALL, above) memory partitions.

SWPTSK is entered with R1 pointing to the new task's disc file name, R4 pointing to the current partition task name, and with the address of a pre-canceling Q-block (like MØ4, above) in R6. R6 is Ø, if no queue is required before the task is cancelled.

SWPTSK copies the new task file name (R1) to the AS6TSK SVC7 parameter block, calls ASWAP (below) to assign LU6 to the new task's disc file, then queues the currently running task with R6, if required. If the task is queued, SWPTSK delays 5Ø ms to give it time to respond. (See DLYMT2, Section 2.6.2.) SWPTSK then calls SWAP (below) which does the actual SVC6 cancel, load, and start.

QNEWT (S131Ø9ØØ-S13127ØØ)

Subroutine QNEWT is called by ENDCALL to send the restart CØØ Q-block to GRAPHER, to send the CALLST queue to a newly started CALL task, and to send the restart CØØ Q-block to PICKER. QNEWT logs an error message if the task cannot be queued. The address of the Q-block to be sent is in R6 when QNEWT is entered. QNEWT also sets or clears CALL mode, by putting the entry value of R1 into CALF (Section 2.6.4).

2.8.4. Overlay 1-Ø3 (SNDRO1Ø3)

SWAP (S13129ØØ-S13156ØØ)

Subroutine SWAP is called by ENDCALL to swap in GRAPHER and by SWPTSK to swap in a CALL or PICKER task. By the time SWAP is called, the swap is all set up. SWAP cancels the currently running task and delays 1 second. Anything less than a second delay leads to occassional errors in the task load, below. The new task file, assigned to LU6, is then loaded into memory and started. Checksum errors are not handled very elegantly at the end of SWAP, but they rarely occur.

ANSCALL (S13157ØØ-S13173ØØ)

ANSCALL is entered from ADVICE (Section 2.6.1) when SOUNDER receives a TFF Q-block from a CALL task. The CALL task request flags are in TFF + 4. If the high order bit (bit Ø) is set, the CALL task is requesting that SOUNDER load an overlay. The overlay number is in TFF + 6 and the area number is in TFF + 8. ANSCALL returns to ADVICE routine CALLOV where the overlay is loaded. This feature is currently not used, although it has been tested.

In addition to bit Ø (MSB), bits 13-15 of TFF + 4, if set, signal CALL task requests. Bit 13 is set if the CALL task wants all of SOUNDER's buffers made available. (See TENUS, Section 1Ø.3.1.) Bit 14 is set if the CALL task is requesting to be terminated. Bit 15 is set if the CALL task wishes to release control of the console back to COUNSEL. If a bit Ø request is made, any other request is ignored. Bit 13-15 requests can be made in any reasonable combination; TFF + 4 = 3 or 5, are common. Bits 1-12 are currently not assigned.

PIK (S13174Ø4-S13174 24)

PIK is the command routine for command PIK, given to specify a PICKER task to be associated with the currently active sounding mode. If the task specified cannot be found on SYS1, ERR 6 results. Otherwise, PIK immediately swaps in the specified task by calling GTPICK, below. The task thus specified is automatically loaded into PICKER's memory partition whenever the mode in question is accessed with MGET, RUN, or CONT.

GTPICK (S1317428-S1317492)

Subroutine GTPICK is called by the PIK command routine and by BTALL (Section 2.9.2.2) to swap a task into PICKER's memory partition. GTPICK calls SWPTSK, above, to effect the swap, then calls QNEWT to send the restart CØØ to the newly started PICKER task. The file name of the new task is saved in PIKT (Section 2.6.4) and in MD.PIK (Section 7.1). Whenever the mode affected is accessed again, this task will be loaded into PICKER's memory partition, unless it is already there.

ASWAP (S13175ØØ-S13185ØØ)

Subroutine ASWAP is called by ENDCALL (above) to assign LU6 to SYS1:GRAPHER.TSK and by SWPTSK to assign LU6 to files SYSC:<CALL task>.TSK or SYS1:<PICKER task>.TSK. An assignment error results in ERR 9 for GRAPHER or a CALL task and ERR 6 for a PICKER task.

2.8.5. Overlay 1-Ø4 (SNDRO1Ø4) [DCW]

Overlay 1-Ø4 is loaded only by PLIST in Ø-Ø8 (Section 2.7.9) when the PTP command is given to display the pulse palette.

PCON1 (S14Ø18ØØ-S14Ø98ØØ)

PCON1 is entered from PLIST in Ø-Ø8 and is a continuation of the PLIST display routine. The 8 pulse descriptions in the palette are displayed in a loop that spans overlay areas Ø and 1. The loop start is PDLP in Ø-Ø8. See PLIST, Section 2.7.9, for details about the PTP command display.

After displaying the pulse palette, the sequence of pulses in a pulse set, as specified by the PSEQ command (Section 2.7.4), is displayed. The PSEQ command is recreated from SCT.PSEQ, using ET.ABUF (Section 7.3) as a temporary buffer.

2.8.6. Overlay 1-Ø5 (SNDRO1Ø5) [DCW]

Overlay 1-Ø5 is loaded only by MCON1 in Ø-Ø9 (Section 2.7.1Ø). This is the third of four overlays invoked by the MGET command display of a sounding configuration.

MCON2 (S15Ø18ØØ-S15Ø7ØØØ)

MCON2 continues the MGET display from MCON1 in Ø-Ø9. MCON2 displays the AGC command specifications through the beginning of the data control section of the display. When practical, the exact form of the SOUNDER command as typed is displayed. When finished, MCON2 goes to MCON3 in 2-Ø2, already loaded for the display subroutines. MCON3 (Section 2.9.3.1) is the final segment of the MGET display routine.

CHAGC (S15Ø71ØØ-S15Ø92ØØ)

Subroutine CHAGC is called by MCON2 to display the AGC command specifications, as well as PKLO and PKHI, for the two receiver channels.

2.9. Overlay Area 2 [DCW]

Overlay area 2, unlike the other three overlay areas which are 512 bytes long, is 256Ø bytes, or 1Ø disc sectors in length. Also, unlike the other areas, each of the three overlays that can occupy area 2 contain two or more modules. The three area 2 overlays, 2-ØØ, 2-Ø1, and 2-Ø2, are described below in Sections 2.9.1, 2.9.2, and 2.9.3, respectively. Each module is described in its own subsection.

2.9.1. Overlay 2-ØØ [DCW]

Overlay 2-ØØ is the SPC overlay for area 2. It is loaded together with Ø-ØØ, 1-ØØ, and 3-ØØ by SNDOV (Section 2.6.1) after a RUN or CONT command is given. These overlays remain in memory for a single sounding or for an SPH series, depending on SPH and SN (Section 2.7.8).

Overlay 2-ØØ contains three modules. Module SNDRPIPE (Section 2.9.1.1) contains the Sounding Pipeline Control (SPC) loop, one of the two main loops in the SOUNDER task. The other is the IMA loop (Section 2.6.1). Module SNDRNXTF (Section 2.9.1.2) contains subroutine NXTFRQ, responsible for sequencing the pulses that comprise a sounding, and for composing the Pulse Configuration Table (PCT, Section 7.2) associated with each pulse. Module SNDRDSUB (Section 2.9.1.3), the disc resident counterpart of SNDRMSUB (Section 2.6.2), contains non-reentrant subroutines used only by the SPC loop.

2.9.1.1. SNDRPIPE [DCW]

Module SNDRPIPE contains the main routine executed during sounding, known as the SPC loop. The SPC loop is executed once for each pulse of a sounding, including TSG NOP fill starts, which are pulses only in the sense that their time-values are accounted for just as if they were data-producing TSG starts. Notice that a pulse in this sense can result in a series of transmission, data acquisition cycles, if coherent averaging in the FEP is selected. This activity in the TSG and FEP, however, results in only one data transfer from the FEP output memories, and is therefore treated like any other pulse by the SPC loop.

SOUNDER executes in the SPC loop for a single sounding or for a series of soundings of the same mode, if specified by command SPH (Section 2.7.8). Only as long as the SPC loop is being executed are SOUNDER's time-keeping counters maintained. (See RDTIME, Section 2.8.1, and CLOCK Section 2.10.1.) The software clock stops when control goes from the SPC to the IMA loop (Section 2.6.1). It is restarted when command RUN or CONT is given, or when control returns to the SPC loop.

SOUNDER Pipeline Registers

The SPC loop reserves registers R4-R12 for special purposes. These registers are given their SPC names by SNDR in PCB.CAL (Section 7.7).

GO (R4) defines the pipeline condition. Only when GO = 1 do new data enter the pipeline. If GO = -1, SOUNDER is between soundings, and the TSG is given NOP starts to mark time until the next sounding begins. If GO = 8000 (hex), a TSG NOP fill sequence is in progress, as specified by command NFIL (Section 2.7.7). If GO = 0, a data overrun has occurred; the TSG is given NOP starts until the pipeline clears enough to continue with normal sounding.

When a sounding is initiated by command RUN, CONT, or TPR, GO goes from -1 to 0 to signal IMALOOP (Section 2.6.1) to exit the IMA and to enter the SPC loop. GO is again set to -1 when the SPC loop is entered and remains -1 until the countdown to sounding is finished.

NPIPE (R5) is the count of data-producing pulses in the pipeline, from SOUNDER's point of view. STTSG (below) increments NPIPE when a data-producing start is given the TSG, and QPCKR decrements NPIPE when SOUNDER is through with the pulse, i.e., when the data have been passed to PICKER.

PULSE (R6) is the count (modulo 32768) of all pulses, data-producing and NOP, since sounding start. At PULSE = Ø, the receiver calibration cycle is executed. At PULSE = 1, the non-pulse-to-pulse DIO devices are set up for the sounding. At PULSE = 4, the receiver DC offsets are copied to the SCT. Between soundings, PULSE = -1.

S. (R7), E. (R8), and D. (R9) are set by SNDRINIT (Section 2.6.5) to contain the addresses of the SCT, ET, and DIO table, respectively. These three registers are preserved by the IMA loop as well as by the SPC loop.

P.TSG (R1Ø) is set by STTSG to define a pulse when the pulse enters the pipeline as the result of a TSG start. The LSB is set if the pulse will produce data (mask 1). The 2nd LSB is set if PICKER is to be queued as a result of the pulse (mask 2). Thus, P.TSG = Ø for a NOP TSG start, and P.TSG = 3 for a normal TSG start.

P.FEP (R11) is set from P.TSG by WTTSG (below), indicating the progress of a pulse from the TSG to the FEP stage of the pipeline. P.FEP controls the action taken by GETBFR and STEKO, below.

P.EKO (R12) is set from P.FEP by GETBFR, indicating the progress of a pulse from the FEP to the EKO stage of the pipeline. P.EKO controls routines WTEKO and QPCKR, below.

<u>OV2</u> (SPIØ34ØØ)

Label OV2 establishes the address of overlay area 2 for the overlay loader (Section 2.6.1). The size of overlay area 2 is established in module SNDRSYST (Section 2.9.3.1).

<u>SINIT</u> (SPIØ35ØØ-SPIØ47ØØ)

The SPC loop is entered at SINIT from IMALOOP (Section 2.6.1) after command RUN or CONT is given. Command TPR also causes SINIT to be entered, but control remains in the SPC loop only until the countdown to sounding is complete. (See Section 2.7.3.) SINIT is entered with register GO = Ø (above), the signal to IMALOOP to enter the SPC loop. SINIT sets GO to -1, and GO remains -1 until the countdown is complete. GO is then set to +1 by NXTSND, below. SINIT initializes SØ2LST (Section 2.6.4), so that GETBFR (below) begins with the first PCT/EKO buffer; initializes HKCHAN to Ø, so that the critical Houskeeping ADC channels (Section 2.8.1) will be read first; and calls subroutine RESFEP (Section 2.6.2) to reset and initialize the FEP.

NXTSND (SPIØ48ØØ-SPIØ97ØØ)

Routine NXTSND is executed only at the beginning of each sounding. NXTSND is entered from SINIT (above) after command initiation of a sounding, or because the next action switch, NXTACT (Section 2.6.4) is set to NXTSND. In the latter case, NXTSND is reentered at the beginning of each of an SPH series of soundings.

NXTSND first checks ET.CDN (Section 7.3) to see if the countdown to sounding is still in progress. If it is, control goes to NXTPLSE (below); the TSG is given NOP starts to mark time until the countdown is complete. Between SPH soundings and during the countdown, NXTSND is the start of the SPC loop; once sounding begins, however, NXTPLSE becomes the top of the SPC loop.

When the countdown is complete, NXTSND performs that initialization which must be done for each sounding. First, ET.CDN is set to -1, signaling to CLOCK (Section 2.1Ø.1) that no countdown is in progress. MODE (Section 2.6.4) is checked, and if it is -1, NXTSND has been entered as the result of a TPR command. Control goes to GOTPR (Section 2.7.3) for the tape replay, SPC simulation run.

If a normal sounding is starting, NXTSND zeroes the lost pulse and echo counters (MD.PLST and MD.ELST) and initializes the stata of the SØ2 Q-blocks to zero. If an SPH schedule is active (NEXT = Ø, Section 2.6.4), and if free run is not specified, ET.TNXT, the number of seconds until the next sounding, is set to 36ØØ/SCT.SPH (truncated integer). CLOCK (Section 2.1Ø.1) will decrement ET.TNXT each second throughout and after the sounding just started. When ET.TNXT = MD.CDN, the countdown to the next sounding will begin. If free run is specified (SCT.SPH = Ø, Section 2.7.8), ET.TNXT will be set to MD.CDN + 1 at sounding end.

Next, NXTSND initializes SWATTN to MD.BAG (Section 2.7.9), in case the sounding just started is S-class. The readings for House-keeping ADC channels Ø-31 are then copied from ET.HKADC to HKDIO. (See the introduction to Section 2.8.1 for the HKDIO layout.)

The status of the SØ4 Q-block, sent at the beginning of the countdown to PICKER (or a Sounding-mode CALL task) by CLOCK, is now checked. If the SØ4 Q-block has been acknowledged by all down-stream tasks, the sounding can begin. Otherwise, an error message is logged that displays the non-zero status. This only rarely happens if a down-stream task is hung up or has been paused for some reason. If the sounding can begin, NXTSND calls subroutine RUNST (Section 2.9.1.3), which does final checking of the synthesizer and transmitter hardware, setting appropriate flags in MD.STAT (Section 7.1).

GO is now set to 1, PULSE and NPIPE are initialized to Ø, and NXTFRQ (Section 2.9.1.2) is called to set up for the receiver calibration cycle.

NXTPLSE (SPIØ99ØØ-SPI131ØØ)

Routine NXTPLSE becomes the start of the SPC loop once the countdown to sounding has finished. NXTPLSE first checks register PULSE. If PULSE < Ø, the SPC loop is marking time between SPH soundings. Control goes directly to STTSG, below. Also, if GO < Ø, the SPC loop is either marking time during a countdown or executing the specified TSG NOP fill sequence, and control goes to STTSG, below. (See NFIL, Section 2.7.7.)

If GO is Ø (data overrun) or 1 (normal), NXTPLSE checks PCT.TIK in NXTPCT (Section 2.6.4) to see if the next pulse is the start of a pulse set. If not, NXTPLSE assumes that enough PCT/EKO buffers are free for the remainder of the pulse set, and control goes to SETDIO, below. Otherwise, the number of free buffers is determined by testing the status of each SØ2 Q-block. For the normal 4-pulse set, 2 free buffers are required before a pulse set of a normal sounding is begun. If a Sounding-mode CALL task (Section 1Ø.3) is running, 4 free buffers are required, because the PCT/EKO buffer data are written to tape before the buffer is released, slowing down the pipeline. If not enough buffers are free at the start of a pulse set, GO is cleared to Ø (data overrun), and control goes to STTSG. The TSG will be given NOP starts until the pipeline clears enough for the next pulse set to begin. When enough buffers are free, GO is again set to 1, and the DIO devices for the first pulse of the next set are set up by SETDIO, below.

SETDIO and DIOLOOP (SPI13300-SPI15100)

Routine SETDIO calls RWULM (Section 2.6.2) to put the DIO interface in the read/write state for DIOLOOP. DIOLOOP writes to the devices on the DIO bus to set them up for when the TSG is started by STTSG, below. The DIO Sounding Table (DST, Section 7.9) specifies which DIO devices are set, and when they are set. Devices that are not declared to be read only and are not pulse-to-pulse devices are set only when PULSE = 1. These settings are left unchanged for the entire sounding. Pulse-to-pulse devices, on the other hand, are set for each pulse, even though only a part of the data byte may change on a pulse-to-pulse basis. Subroutine WRDIO (Section 2.9.1.3) is called to do the actual write and to verify the write, if so specified. (See DIOGEN, Section 10.2.4.)

Notice that at this time in the pulse cycle, the data from the previous pulse are in the FEP input memories, and the FEP ADCs are inactive. The DIO bus can now be active without fear of its noise interfering with the digitization process.

STTSG: Start the TSG (SPI15300-SPI24000)

Although STTSG is not the start of the SPC loop, this routine starts the transmission, data acquisition cycle. STTSG either sends the TSG a normal, data-producing start, or a NOP fill start to mark time. Register GO determines what kind of start, or command byte, STTSG sends to the TSG.

If GO = 1 and PULSE = 0, the sounding has just begun, and the receiver calibration cycle is initiated by sending the TSG a command byte of A1 (hex), selecting the special FEP program that computes the DC offsets of the receivers. (See SCT.1X0, etc., Section 7.1.) The receiver DC offsets are part of the SYS display (Section 2.9.3.1). TSG command byte A1 normally selects the 20 ms TSG pace, which allows ample time for the FEP to collect and process 512 data samples at the 10 or 20 microsecond digitization rate. For the receiver calibration cycle, STTSG sets P.TSG to 1, meaning that the TSG start will produce data, but that the data will not be passed to PICKER for processing of any kind. However, NPIPE is incremented to count the receiver calibration pulse in the pipeline.

If GO = 1 and PULSE > 0, a normal data pulse cycle is about to begin. The TSG command byte in TSGNXT is sent to the TSG and P.TSG is set to 3, meaning that the pulse will produce data that will be passed to PICKER. NPIPE is incremented, counting the pulse in the data pipeline. The TSG command byte in TSGNXT comes from the pulse palette. (See PSCONT, Section 2.9.1.2.)

If GO = -1, the SPC loop is marking time either during a countdown or between SPH soundings. STTSG sends command byte CØ to the TSG, selecting the NOP TSG scenario and normally the 1ØØ ms TSG pace. P.TSG is zeroed, meaning the pulse produces no data or queue to PICKER.

If GO = 8ØØØ, the TSG NOP fill sequence specified by command NFIL (Section 2.7.7) is in progress. If GO = Ø, a data over-run condition has been detected by NXTPLSE, above. In either case, the TSG command byte is composed of scenario Ø and the pace of the last normal TSG command byte sent, from TSGNXT; the TSG will mark time, but the pulse will produce no data, so P.TSG is zeroed.

If the user has specified that scenario Ø be used as part of the pulse set by command TSGC (Section 2.7.6), P.TSG is set to 2, meaning that no data are produced by the TSG start, but that PICKER should be queued to keep the pulse set intact. PICKER will discard the PCT, because the EKO buffer length will be Ø.

STTSG displays different numbers in the hex panel lights for the different TSG starts. For a normal start, the frequency index from SUBFIX is displayed. For the receiver calibration cycle, hex 1111 is displayed. During the countdown, STTSG leaves the display alone, because CLOCK (Section 2.1Ø.1) has displayed the value in ET.CDN. When marking, or biding, time between soundings, STTSG displays BIDE; if marking time during a data overrun, STTSG displays DEAF. Finally, during the specified NOP fill sequence, STTSG displays the value from NFILL (Section 2.6.4), or the number of TSG NOP starts left until the next pulse set starts. For the most part, these numbers change too rapidly to be readable. However, they are useful for debugging or to tell at a glance whether SOUNDER is executing in the SPC or IMA loop and whether an SPH series is active.

If the TSG is not configured into the DIO system, or if command NO T was given before sounding, STTSG calls DLYMT2 (Section 2.6.2) to simulate the TSG pacing, for debugging without the TSG.

The TSG internal clock ticks synchronously at the four paces strapped on the TSG board. Because SOUNDER 2.Ø2 makes more liberal use of the various paces, it must sometimes NOP the TSG up to a certain timing mark before sending a specified TSG start. For example, if at 13Ø ms into a sounding, the user calls for a TSG start at the 2Ø ms pace, STTSG issues a 1Ø ms NOP start to the TSG, getting to the next 2Ø ms timing mark, before the 2Ø ms start is issued.

After starting the TSG, STTSG checks if there is a PCT to stage. (See PCTLST-NXTPXT, Section 2.6.4.) If GO < 1 or if PULSE = Ø, the SPC is marking time or the receiver calibration cycle has just begun. In either case, there is no PCT to stage. Otherwise, STTSG copies NXTPCT to one of the TEMPCT staging areas and pushes the address of the staged PCT to STGLST. GETBFR (below) will pop the address of the staged PCT from STGLST when the data transfer associated with the PCT is set up.

STTSG next calls ADVICE (Section 2.6.1) to check if command AB, TX, IF, or RF (Section 2.1Ø.1) has been given. Subroutine NXTFRQ (Section 2.9.1.2) is then called to set up the next pulse and to compose the associated PCT.

<u>WTEKO: Wait for EKO DONE</u> (SPI242ØØ-SPI285ØØ)

WTEKO first ckecks register P.EKO to see if the EKO driver was started the last time through the SPC loop. If not, control goes to QPCKR, in case a scenario Ø TSG start was specified by command TSGC. (See STTSG, above.) If an EKO transfer was initiated during the last SPC cycle, WTEKO checks the SVC1 parameter block status to see if the transfer is complete. If so, a significant amount of time can be saved by not calling SVC1 to wait for the transfer to complete. If the SVC1 wait call is made, however, the down-stream tasks get a chance to execute at this time. If the EKO device times out (ERROR 82F1), subroutine DEVERR (Section 2.6.2) is called to count the error and to flag it in MD.STAT (Section 7.1). SOUNDER continues to execute, but the EKO device is flagged as dead until the sounding ends or AB is typed. (See NOEKO, Section 2.6.4.)

When the EKO transfer has successfully terminated, WTEKO checks register PULSE. If PULSE = 4, the data are from the second normal data-producing TSG start, and the DC offsets in the SCT are set. (See SCT.1XØ, etc., Section 7.1.) The DC offsets are taken from the second normal EKO buffer because the first may be from a complementary coded sounding, and the odd-numbered buffers of such a sounding do not contain the DC offsets.

At PEAKS, the 1/2-word count in the first two bytes of the transferred data is checked. If less than 15 1/2-words were transferred, WTEKO ignores the data and goes directly to QPCKR, below. Next, the high order bit of PCT.FRQ1 (Section 7.2) is checked. If the data were from a pulse at a protected frequency, control goes directly to QPCKR so that the AGC will be unaffected by these data. Otherwise, the maximum amplitudes seen in the data of the two receiver channels by the FEP are compared to the largest peaks seen in previous data since the AGC was last updated by UPAGC (Section 2.9.1.3). The highest peaks are saved in MAXPK for UPAGC. If FEP PICKER processing is being used, the peaks are from the selected echo data, which prevents a transient peak from lightning, etc., from decreasing the receiver gain. (See Section 13.8 for FEP PICKER processing.)

<u>QPCKR:</u> <u>Queue</u> <u>PICKER</u> (SPI28700-SPI31000)

QPCKR first tests P.EKO to see if PICKER is to be queued this SPC cycle. If not, but if an EKO transfer has just completed, the data are from the receiver calibration cycle or from a data overrun, and control goes to OUTP (below) to decrement NPIPE. If there is no new data and no queue for PICKER (NOP fill), control goes to GETBFR.

If P.EKO indicates that PICKER is to be queued, QPCKR gets the address of the S02 Q-block associated with the last EKO buffer allocated, from the bottom of S02LST (Section 2.6.4), leaving the stack intact. If the sounding is T- or W-class, the SCT.FTBL entry address (Section 7.4) in AFTQ (Section 2.6.4) is put into S02 + 8 for PICKER echo ranging. If the sounding is S-class, AFTQ contains zero. The S02 queue is then sent to PICKER.

At CKAGC, subroutine UPAGC (Section 2.9.1.3) updates the attenuation bytes for receiver channels under AGC, if it is time to do so. The pulse that entered the pipeline two cycles ago at STTSG now exits the pipeline, and NPIPE is decremented by one at OUTP.

GETBFR: Allocate PCT/EKO Buffer (SPI31100-SPI33800)

GETBFR first copies P.FEP into P.EKO. P.EKO will control routines WTEKO and QPCKR the next time through the SPC loop. GETBFR next tests P.FEP to see if PICKER is to be queued with the data currently in the FEP output memory awaiting transfer to SOUNDER memory. If not, there is no need to tie up one of SOUNDER's PCT/EKO buffers. The FEP DATA READY signal must however be cleared, so the first two bytes of the data will be transferred to DREGS, a convenient dumping area. (See the EKO Driver, Section 2.11.3.) Unless there is a data overrun (below), only the data from the receiver calibration cycle will be dumped in this manner.

If good data are to be transferred, GETBFR allocates a PCT/EKO buffer by popping the address of an S02 Q-block, containing the PCT and EKO buffer addresses, from the top of stack S02LST (Section 2.6.4). The same S02 Q-block address is pushed to the bottom of S02LST to circulate the list of S02 Q-block addresses. Thus, the buffers are used in round-robin fashion. The address of the associated staged PCT is popped from the top of STGLST (STTSG, above) and if PCT.TIK of this PCT indicates the first echo returns of a pulse set, a new SCT.FTBL address is popped from AFTLST and put into AFTQ, in case the sounding is T- or W-class. This is the value put into S02 + 8 by QPCKR, above.

If the PCT/EKO buffer just allocated has been released by PICKER (S02 status is 0), the staged PCT is copied to the PCT/EKO buffer, the buffer word count is zeroed, the S02 status is set to -1000 (hex), and control goes to STEKO, below. Otherwise, the data awaiting transfer from the FEP output memory are lost. The lost echo is counted in MD.ELST (Section 7.1), and to preserve the data already in the PCT/EKO buffer just allocated, the first two bytes of the data are transferred to DREGS, as are the receiver calibration data, above. P.EKO is set to 1: PICKER will not be queued, but the data must be cleared from the pipeline.

STEKO: Start the EKO Transfer (SPI34000-SPI35400)

STEKO first tests P.FEP to see if the TSG start at the beginning of the previous SPC cycle resulted in a FEP start. If the TSG was started with scenario 0, the FEP was not started, and no data can result. Control goes directly to WTTSG, below; the next time through QPCKR, PICKER will be queued with a good PCT, but the EKO buffer will be empty.

STEKO is entered with the buffer address for the EKO DMA data transfer in R14 and with one less than the transfer size in R15. From these values the SVC1 parameter block is set up for the transfer. Unless NOEKO (Section 2.6.4) is true, the status of the SVC1 parameter block is set to 1, and the EKO transfer is initiated. When the EKO transfer is complete, the EKO driver (Section 2.11.3) clears the parameter block status to Ø. If WTEKO, above, finds the status to be Ø, the time used to make the SVC1 wait call is conserved.

<u>WTTSG: Wait for TSG DONE</u> (SPI355ØØ-SPI373ØØ)

WTTSG checks the SVC1 parameter block status for the TSG start and if it is Ø, the TSG is already done; no time need be spent making the SVC1 wait TSG call. If the TSG times out (ERROR 8237), DEVERR (Section 2.6.2) is called to count the error and to flag it in MD.STAT (Section 7.1), and the TSG is flagged dead for the rest of the sounding or until AB is typed. (See NOTSG, Section 2.6.4.) A TSG timeout can be caused by the FEP not sending a DONE signal to the TSG. This condition can be tested with CALL task PIPLN, Section 1Ø.2.15.

Finally, WTTSG copies P.TSG into P.FEP and calls CLOCK (Section 2.1Ø.1) to update SOUNDER's time-keeping counters. P.FEP will dictate what happens in routines GETBFR and STEKO the next time through the SPC loop.

<u>HKADC: Read the House-keeping ADC</u> (SPI375ØØ-SPI391ØØ)

The last event in the SPC cycle during sounding is the reading of the House-keeping ADC channel in HKCHAN. Subroutine RDHSK (Section 2.6.2) is called to read the channel into ET.HKADC (Section 7.3). HKCHAN is incremented so that the HKADC channels are read in round-robin order, Ø-63. (MD.NHSK can be set by MAKEM, Section 2.1Ø.1, to read more than one HKADC channel per SPC loop.)

<u>End of SPC Loop</u> (SPI392ØØ-SPI4Ø5ØØ)

At the end of the SPC loop, PULSE is tested. If PULSE < Ø, the SPC loop is marking time between soundings or during a countdown to sounding; control goes to NXTSND. If NXTFIX > Ø (Section 2.6.4), a sounding is still in progress, and control goes to NXTPLSE. If NXTFIX = Ø, but NPIPE > Ø, the sounding has finished, but the data pipeline has not yet emptied;

control goes to NXTPLSE. Otherwise, all data from the sounding have been gotten from the FEP and passed to PICKER. NXTFIX is cleared to Ø to signal CLOCK (Section 2.1Ø.1) that a new sounding can be started when ET.TNXT = MD.CDN.

End of Sounding (SPI4Ø7ØØ-SPI479ØØ)

At the end of each sounding all 64 HKADC channels and DIO ports are read into the HKDIO buffer as described in Section 2.8.1. Although DIO Ø cannot be used, no special case is made of it here.

The DIO devices, as specified by the DST (Section 7.9), are set to their idle states, e.g., the RF and IF attenuators are normally set to zero gain with manual control enabled. (See DIOGEN, Section 1Ø.2.4.)

Next, the SØ6 Q-block is sent to PICKER and in turn to all the other down-stream tasks with the status, SØ6 + 2, initialized to -FFF. Subroutine WTPOST (Section 2.6.2) is called to mark time with DLYTSG until all post-sounding activity, such as MANAGER writing tape, is completed by the down-stream tasks, with the exception of the Scheduler (ANALYSER). The Scheduler is held in abeyance until the SØ6 status is set to -1ØØØ, below. After WTPOST returns, subroutine RUNEND (Section 2.1Ø.1) is called to log lost pulses and echoes, if required, and to log the sounding end message. The SØ6 status is now set to -1ØØØ (hex), the all-clear signal for the Scheduler to continue.

NEXT (Section 2.6.4) is now tested to see if an SPH series is in progress. If not (NEXT not Ø), control goes to ENTIMA, below. Otherwise, if a free run SPH series is active (SCT.SPH = Ø), ET.TNXT is set to MD.CDN + 1; the next sounding will begin as soon as possible, unless SN (Section 2.7.8) is satisfied. If SCT.SNO = SCT.SN, SCT.SN is reset to -1, and NEXT is set to 1 to disable the CONT command; control goes to ENTIMA. Otherwise, the SPH series continues; the device error bits in MD.STAT (Section 7.1) are cleared for the next sounding, and control goes to NXTSND, above. The SPC loop will mark time until the next sounding begins.

ENTIMA is entered at sounding end if no SPH series is in progress, or when SN is satisfied. ENTIMA calls GTGRAF (WTPOST, Section 2.6.2) to reinstate GRAPHER, in case an SPH series involving a Sounding-mode CALL task has ended, sets NXTACT to GETADV, and enters the IMA loop (Section 2.6.1).

2.9.1.1. SNDRPIPE

2.9.1.2. SNDRNXTF [DCW]

Module SNDRNXTF contains the next-frequency subroutine, NXTFRQ, which is responsible for sequencing the pulses that make up a sounding. NXTFRQ decodes SCT.PSEQ (Section 2.7.4) to sequence the pulses in a pulse set, getting appropriate data from the pulse palette, SCT.PD (Section 7.5). NXTFRQ also sets up the DST (Section 7.9) for the next TSG start from computed DIO data and from the palette extensions, DIO.PTP (Section 2.7.4). NXTFRQ calls routines in overlay Ø-ØØ (Section 2.7.1) to sequence the pulse sets, that is, to select base and ramp frequencies about which pulse sets are formed.

NXTFRQ Entry (SNXØ21ØØ-SNXØ38ØØ)

The NXTFRQ subroutine is entered from the SPC loop (Section 2.9.1.1) at label NXTFRQ. NXTFRQ first tests register GO. If GO = Ø, NXTPLSE in the SPC loop did not find not enough PCT/EKO buffers free to begin a pulse set. NXTFRQ returns to the SPC loop and will not set up for the next pulse until the pipeline clears.

If GO = -1, NXTFRQ returns to the SPC loop, because SOUNDER is between soundings or executing a countdown to sounding.

If GO = 8ØØØ (hex), the TSG NOP fill sequence specified by command NFIL (Section 2.7.7) is in progress. NFILL (Section 2.6.4) is decremented, and if it remains positive, the fill sequence is continued. Otherwise, GO is set to 1, and NXTFRQ proceeds as follows.

If GO = 1, the pipeline registers are saved in AREGS, and the NXTFRQ registers are gotten from FREGS. (See Table 2.4, Section 2.7.1 for the NXTFRQ register usage.) R15 now specifies which NXTFRQ routine is to be executed to set up the next pulse. NXTFRQ dispatches through R15 to one of the routines described below.

RXCAL (SNXØ39ØØ-SNXØ69ØØ)

NXTSND in SNDRPIPE (Section 2.9.1.1) sets R15 in FREGS to first execute RXCAL to set up for the receiver calibration cycle or pulse. The receiver calibration pulse is the first pulse of each sounding. RXCAL clears R8 to Ø, the RXCAL flag to subroutine STPDTA (below) and clears PCT.TIK in NXTPCT to Ø. The elapsed time for a sounding is thus initialized; CLOCK (Section 2.1Ø.1) updates this value each time through the SPC loop.

Subroutine STPDTA is called to set up the DST (Section 7.9) for the receiver calibration cycle. The devices listed in table CALDIO (below) will be set by DIOLOOP (Section 2.9.1.1) to enable the FEP to compute the DC offsets for the receiver channels.

The base frequency and frequency index (NXTFREQ and NXTFIX, Section 2.7.1) are initialized by sounding class. The first base frequency for an S-class sounding is set from MD.SFRQ (Section 2.7.7). The first base frequency for a T- or W-class sounding is set to the frequency in the first active SCT.FTBL entry, the address of which was put into A1STF by CKFTBL (Section 2.8.3) at sounding initiation by command RUN or CONT. This SCT.FTBL entry address (Section 7.4) is pushed to AFTLST (Section 2.6.4) to initialize this stack for GETBFR (Section 2.9.1.1). Unless the sounding was aborted during the countdown (NXTFIX = Ø), NXTFIX is computed from NXTFREQ by FRTON (Section 2.6.3). NRND, the number of passes through the base frequency selection, is initialized from MD.NSET. (See NXTBAS, Section 2.7.1, and command NSET, Section 2.7.7.) The pulse counters for UPAGC, AGCPUP (Section 2.9.1.3) are initialized from MD.PPU. (See command AGC, Section 2.7.9.) Finally, R15 is set for NXTFRQ to next execute RESDTA (below). The NXTFRQ registers are saved in FREGS, the SPC registers are recovered from AREGS, and NXTFRQ returns to the SPC loop to execute NXTPLSE.

<u>RESDTA</u> (SNXØ71ØØ-SNXØ94ØØ)

Routine RXCAL calls subroutine STPDTA, with R8 = Ø, to set up the DST (Section 7.9) for the receiver calibration cycle. After the receiver calibration cycle has been initiated by STTSG (Section 2.9.1.1), NXTFRQ is called to execute RESDTA. RESDTA calls STPDTA, with R8 = 1, to restore the DST to its sounding condition.

RESDTA also initializes the two ramp steps in RSTEP (Section 2.6.4). The two 32-bit B-ramp steps are simply set from MD.SUBS (Section 7.1). The Z-ramp steps are initialized according to (b.) under, <u>The Z-ramp Algorithm</u>, Section 2.7.1. Once the ramp steps and their backup values are initialized, RESDTA goes to NXTF1 in overlay Ø-ØØ (Section 2.7.1) to initialize for the first pulse of the sounding. Routines INRAMP and INSUBN in Ø-ØØ are executed, after which routine INPSET (below) is entered to set up the DST and the PCT for the first pulse of the first pulse set.

STPDTA (SNXØ95ØØ-SNX122ØØ)

Subroutine STPDTA is called by RXCAL (R8 = Ø) and by RESDTA (R8 = 1) to set up the DST for the receiver calibration cycle and to restore the DST to its sounding condition after the receiver calibration cycle. STPDTA modifies the DST.PDTA entries for only those dedicated DIO devices in table CALDIO (below), the object being to fully attenuate the receivers for the first cycle of each sounding to allow the FEP to compute the DC offsets of the receiver channels.

INPSET (SNX124ØØ-SNX133ØØ)

After the next base or ramp frequency is selected by routines in overlay Ø-ØØ (Section 2.7.1) and the TSG NOP fill sequence, if any, has executed, INPSET is entered from Ø-ØØ to initialize for the first pulse of a pulse set. PCT.TIK (Section 7.2) of NXTPCT (Section 2.6.4) is logically ORed with 8ØØØ (hex), turning on the first-of-set flag. R5 is initialized to index SCT.PREP, the PSEQ repeat counts, R6 is initialized to point to the top of SCT.PSEQ, and R3 is set to -1, meaning that no repeated pulse sequence is in progress. (See command PSEQ, Section 2.7.4, and the NXTFRQ registers in Table 2.4, Section 2.7.1.)

PSCONT (SNX134ØØ-SNX285ØØ)

PSCONT is the entry to the next pulse set up routine for all pulses of the pulse set except the first. PSCONT clears the first-of-set flag in PCT.TIK of NXTPCT and executes routine PSCON1, as does INPSET (above).

PSCON1 gets the SCT.PSEQ entry pointed to by R6, strips off any flags it may contain, and puts the resulting pulse palette index into PCT.PDIX in NXTPCT. The same palette index will later be used to index into the palette extensions in DIO.PTP.

The delta F factor (PD.DFAC) for the Pulse Description (PD) specified for the pulse is multiplied by the delta frequency (MD.DELF, set by command DELF), and depending on the sign of PD.DFAC, this product is either added to or subtracted from the base or ramp frequency of the pulse set. The frequency of the pulse thus computed is put into PCT.FRQ1 and PCT.FRQ2 of NXTPCT. The frequency index merged with the high order of frequency put into PCT.FRQ1 is that of the base or ramp frequency of the pulse set, unadjusted by the delta F computation. If the frequency of the pulse (the adjusted base or ramp frequency) is found to be protected or out of range by MAPFRQ (Section 2.6.2), the sign bit of PCT.FRQ1 is set, and one is added to the TSG command byte (PD.TSGC) specified for the pulse before it is put into TSGNXT for STTSG (Section 2.9.1.1). For all but W-class modes, each even-numbered scenario, n, has an associated NOP scenario, n + 1. For example, if scenario 2 was specified by command TSGC (Section 2.7.6) for the PD selected for the pulse being set up, and if the pulse frequency is protected, the SPC loop will keep pace for the pulse, but with the NOP associated with scenario 2, which is scenario 3. W-class soundings are prevented from selecting protected frequencies by allowing only null ramps and a delta F of Ø.

The pulse frequency computed above in Hz/1ØØ is converted to BCD for the frequency synthesizer by subroutine FTOBCD (Section 2.6.3), and the DST.PDTA entries (Section 7.9) for the three DIO ports dedicated to the synthesizer are set with calls to SETDST (Section 2.6.2).

For the first pulse of each pulse set, the calibration attenuation is set by frequency band, as specified by operator command, CALA (Section 2.7.7). The DST.PDTA entry for the DIO port dedicated to the calibration attenuator is set by SETDST, and the attenuator command is put into PCT.CALA of NXTPCT.

Also at the start of each pulse set, the RF and IF receiver attenuations in dB are converted to DIO command form for both channels, and the commands are put into the DST.PDTA entries for the DIO ports dedicated to the attenuators. (See STATTN, below.) The attenuation commands are also packed into PCT.RFIF of NXTPCT. For S-class soundings, the attenuations in dB come from SWATTN (Section 2.6.4). For T- and W-class soundings, AFTF has been set by NXTBAS (Section 2.7.1) to point to the appropriate SCT.FTBL entry for the pulse set in progress at the NXTFRQ level. The attenuations are from FR.ATTN of this SCT.FTBL entry (Section 7.4).

At PTPLP, the DST (Section 7.9) is scanned for DIO ports declared by the user to be extensions of the pulse palette. (See DIOGEN, Section 2.1Ø.4.) Such ports may be entirely user-defined or may be dedicated ports with normally unused bits defined by the user to be set pulse-to-pulse. In the latter case, a special case is made of dedicated devices for which data are computed by the routine being described. These devices are listed by their DIO.DED table offsets (Section 7.9) in table PCTDIO (below). For these devices, the computed data are merged with data from their DIO.PTP entry, as specified by the palette index for a given pulse.

Notice that making multiple use of a DIO port in this manner minimizes the number of ports written to by DIOLOOP (Section 2.9.1.1), which makes the SPC loop faster. However, the user must not intrude upon bits set by the dedicated DIO port routines. One must have a complete knowledge of how a given dedicated DIO port is commanded and set the pulse-to-pulse mask accordingly.

For ports that are entirely user-defined pulse palette extensions, the data byte from the appropriate DIO.PTP entry is also selected by the pulse palette index (PCT.PDIX from SCT.PSEQ) for the pulse in question. The data byte from DIO.PTP is logically ANDed with the inverted pulse-to-pulse mask (DST.PMSK XOR FF), and this result is ORed with the old data byte from DST.PDTA for the port in question. The final result is put back into DST.PDTA. Thus, the user-defined pulse palette extension may also be set up for multiple use: some bits may not change for an entire sounding, while others may change pulse-to-pulse.

At DPEX, the pulse selection control registers R3-R6 (Table 2.4) are updated. The next pulse in the pulse set will be selected the next time PSCONT is entered, or if the pulse set is finished, R15 in FREGS is set to execute NXTF2 in Ø-ØØ (Section 2.7.1) when NXTFRQ is called during the next pass through the SPC loop.

<u>STATTN</u> (SNX286ØØ-SNX297ØØ)

Subroutine STATTN is called to set the DST.PDTA entries for each of the RF and IF attenuators. STATTN converts the attenuation (dB) in R8 to an attenuator command and calls SETDST (Section 2.6.2) to set the DST.PDTA entry for the DIO.DED entry in R5. If the high order bit in PCT.FRQ1 of NXTPCT indicates that the sounding frequency is protected, R8 is cleared to Ø, the command for full attenuation, before the call to SETDST. The receivers are thus protected from a nearby transmitter, and the data from a protected frequency pulse do not affect the AGC. (See WTEKO, Section 2.9.1.1.)

CALDIO (SNX29900-SNX31500)

CALDIO is the receiver calibration DIO device table used by subroutine STPDTA to set up for and restore from the receiver calibration cycle, the first pulse of each sounding. Each CALDIO entry contains 3 bytes. Byte 0 is the DIO.DED table offset for the port (Section 7.9). Byte 1 is the DIO setting for the port for the receiver calibration cycle. Byte 2 is the save area for the DST.PDTA data byte during the receiver calibration cycle. The RESDTA call to STPDTA copies byte 2 back to the DST.PDTA entry for each CALDIO port. CALDIO must terminate with a byte of 80 (hex).

PCTDIO (SNX31600-SNX33100)

Table PCTDIO contains the DIO.DED table offsets (Section 7.9) for those dedicated DIO ports for which data are computed by subroutine NXTFRQ. These data are passed with the pulse echo data in the PCT associated with the pulse. These ports may, or may not, be defined for multiple, pulse-to-pulse use. PCTDIO must terminate with a byte of 80 (hex).

2.9.1.3. SNDRDSUB [DCW]

Module SNDRDSUB contains disc-resident, non-reentrant subroutines that, like NXTFRQ (module SNDRNXTF, Section 2.9.1.2), are called from, and as such are a part of, the SPC loop (Section 2.9.1.1).

RUNST (SDS02100-SDS07400)

Subroutine RUNST is called by NXTSND (Section 2.9.1.1) just before the receiver calibration cycle, the first pulse of each sounding, to do final checkout of the transmitter hardware and to set the DST entry (Section 7.9) for the DIO port dedicated to the transmitter attenuator.

After putting the DIO interface in the read only state by calling ROULM (Section 2.6.2), RUNST reads the transmitter control (DED.TXSA) flags. If the high voltage is on, the HV bit (mask 8000, hex) in MD.STAT (Section 7.1) is turned on, and MD.TXHP, which may be set by command HPAT (Section 2.9.2.2), is inverted and used to set the DST entry for the transmitter attenuator DIO port. If the HV is not on, the HV bit in MD.STAT is turned off, and MD.TXLP, from command LPAT, is used for the transmitter attenuation for the sounding. Subroutine SETDST (Section 2.6.2) is called to set the DST entry for DED.TXAT, the transmitter attenuator DIO port.

If the dummy load (DL) bit in the DED.TXSA flags is on, the antenna transmitter load has been selected by command DLOF (Section 2.9.2.2), and the DL/antenna bit (mask 4000) is turned on in MD.STAT. If the dummy transmitter load is selected, either by command DLON (Section 2.9.2.2) or by default at HF Radar start up, the MD.STAT DL/antenna bit is turned off, and the message, "DL ON", is logged. If the high voltage dump bit in the DED.TXSA flags is off, the message, "HV DUMP", is logged.

RUNST also reads the status of the synthesizer (DED.SYNS), and if the control source bit is on, the synthesizer is under remote (SOUNDER) control, and the synthesizer bit (mask 2000) in MD.STAT is turned on. If the synthesizer is under local (manual) control, this bit is turned off. CLOCK (Section 2.10.1) also reads the DED.SYNS flags when the countdown to sounding begins, and logs an error message, if any of the three low order flag bits is off.

<u>UPAGC</u> (SDS07600-SDS14100)

Subroutine UPAGC is called by QPCKR (Section 2.9.1.1) to update the attenuation settings for receiver channels under Automatic Gain Control (AGC), as specified by command AGC (Section 2.7.9). The attenuation bytes are updated in SWATTN (Section 2.6.4) for S-class soundings. For T- and W-class soundings, the attenuation bytes are updated in FR.ATTN of the SCT.FTBL entry (Section 7.4) pointed to by AFTQ (Section 2.6.4). MD.CTRL (Section 7.1) is examined to determine the sounding class.

The two receiver channels are independently controlled as follows. If the channel is under AGC, as specified by MD.AGC, UPAGC decrements its pulse counter in AGCPUP (Section 2.6.4). If its counter goes to zero, it is time to update the channel's AGC. The AGCPUP counter is reinitialized from MD.PPU (Pulses Per Update), and the maximum peak seen in the channel's data since the last AGC update is gotten from MAXPK. The MAXPK entry for the channel is then zeroed for the next AGC update cycle. (See WTEKO, Section 2.9.1.1.) The value in MAXPK is compared to the limits of the receiver channel's dynamic range, LOAGC and HIAGC (Section 2.6.4). The receiver gain is raised, lowered, or left as is, based on this comparison. The RF and IF attenuations are stepped between the minimum, or base, values in MD.BAG and the maximum 60 dB, according to Table 2.6.

RECEIVER GAIN	RF ATTENUATION	IF ATTENUATION
Ø	Ø	Ø
-4	Ø	-4
-8	Ø	-8
-12	Ø	-12
-16	Ø	-16
-2Ø	Ø	-2Ø
-24	-4	-2Ø
-28	-4	-24
-32	-8	-24
-36	-8	-28
-4Ø	-12	-28
-44	-12	-32
-48	-16	-32
-52	-16	-36
-56	-2Ø	-36
-6Ø	-2Ø	-4Ø
-64	-24	-4Ø
-68	-24	-44
-72	-28	-44
-76	-28	-48
-8Ø	-32	-48
-84	-32	-52
-88	-36	-52
-92	-36	-56
-96	-4Ø	-56
*-1ØØ	-4Ø	-6Ø
*-1Ø4	-44	-6Ø
*-1Ø8	-48	-6Ø
.	.	.
.	.	.
.	.	.
*-12Ø	-6Ø	-6Ø

Receiver Gain in dB relative to maximum.

* At these attenuations, overload would occur in the antenna RF preamplifiers supplied.

Table 2.6. ATTENUATION STEPS.

The base RF and IF attenuations are specified by command AGC (Section 2.7.9) with default values of Ø. The updated attenuation bytes are put into SWATTN or FR.ATTN of the appropriate SCT.FTBL entry, depending on sounding class (MD.CTRL). If neither channel is under AGC, or if it is not yet time to update the AGC (AGCPUP, above), UPAGC returns to QPCKR without altering SWATTN or FR.ATTN.

Notice that at the beginning of each S-class sounding, SWATTN is initialized from MD.BAG. T- and W-class FR.ATTN entries for each active SCT.FTBL entry are also initialized from MD.BAG, but only when the frequency in question is set or adjusted. (See command F, Section 2.7.5.) The T- and W-class attenuation settings for each active frequency are carried over from sounding to sounding and from session to session for a given mode.

Notice also that there is a delay of one pulse set time for the AGC to adjust. It remains to be seen how this will affect sounding modes like Z-mode that are very frequency-agile. (When the AGC command argument, Pn, is fully implemented, this delay can be minimized.)

<u>CPYPCT</u> (SDS142ØØ-SDS153ØØ)

Subroutine CPYPCT copies a 1Ø-byte PCT from the address in R3 to the address in R13. CPYPCT is called by STTSG (Section 2.9.1.1) to stage the PCT in NXTPCT (Section 2.6.4), then is called by GETBFR to copy the staged PCT to a PCT/EKO buffer.

<u>WRDIO</u> (SDS155ØØ-SDS199ØØ)

Subroutine WRDIO is called from the SPC loop (Section 2.9.1.1) by DIOLOOP to set the DIO devices for the next pulse, and is called at the end of each sounding to set the DIO devices to their idle states, as specified by DIOGEN (Section 1Ø.2.4). WRDIO is also called by SNDRINIT (Section 2.6.5) to idle the DIO devices at SOUNDER start up.

WRDIO is entered with the DIO data byte in R1, the port control flags (DST.CTRL) in R3, and the DST index for the DIO port in R15. (See Section 7.9 for the definition of the DST parameters.) The data byte is saved in case the DIO write is to be verified.

The data byte is written to the DIO port specified by R15, and if the device is remote (DIO.RDIO of DST.CTRL true), command 5 is sent to the central unit (DED.CREM).

If the write need not be verified (DIO.VRFY false), the DIO write is complete, and WRDIO returns to the caller. Otherwise, the DIO port is read back, and the data byte read is ANDed with the data direction mask, DST.DMSK, for the port. This result is then compared to the data byte written, which has already been ANDed with DST.DMSK before the WRDIO call. If the read does not verify the write, DEVERR (Section 2.6.2) is called to count the DIO error and to flag it in MD.STAT (Section 7.1). No other action is taken. A write-verify of a remote DIO port requires a delay of over 300 microseconds.

Notice, that for want of a proper DIO exerciser, DIOGEN (Section 10.2.4) can be used to force verification of suspect DIO ports. The DIO error count is displayed by command SYS (Section 2.9.3.1).

2.9.2. Overlay 2-01 [DCW]

Overlay 2-01 contains two modules. Module SNDRBOOT (Section 2.9.2.1) contains routines to load the FEP program memory (BOOT FEP), to pass parameters to the FEP program (FEP n), to bootstrap the TSG and to load TSG programs (BOOT TSG), and to read the DIO control tables into SOUNDER memory (BOOT DIO). SNDRBOOT also contains the routine that times the TSG paces for command TICK (Section 2.7.9).

Module SNDRSTUP (Section 2.9.2.2) contains command routines for the control of the transmitter, the routine that reads the protected frequencies maps from the disc, and routines that perform initialization for mode switching after command RUN or CONT is given.

2.9.2.1. SNDRBOOT [DCW]

Module SNDRBOOT, as part of overlay 2-01, is loaded by routine BOOT in SNDRADVC (Section 2.6.1) when any form of the BOOT command is given. Also, any form of the FEP command and command TICK cause 2-01 to be loaded. Because mode switching may invoke new DIO control tables or FEP and TSG programs, 2-01 is loaded by

Ø-Ø1 (Section 2.7.2) after the RUN or CONT command is given.

PASSF (SBOØ21ØØ-SBOØ53ØØ)

PASSF is the entry for the FEP command, given to pass parameters to FEP program locations 2-5. If the FEP command has a single argument, it is taken as the FEP thresholding level to be passed to FEP program location 5 and to be saved in MD.FEPT (Section 7.1). Subroutine FEPASS (below) is called to pass the thresholding level to FEP location 5.

If more than one argument is given to the FEP command, the first argument must be 2, 3, or 4, or the FEP command is rejected with ERR 1. If the location specified is valid, the second argument is taken as a count, which is masked to 1Ø bits, limiting the magnitude of the count to 1Ø23 (decimal). The third argument is a constant, also masked to 1Ø bits of magnitude. MD.FEPP (Section 7.1) is a 12-byte field in the SCT for the parameters for FEP locations 2-4, with 4 bytes reserved for each location. The count is saved in bytes Ø and 1 of a given location's 4-byte field, and the constant is saved in bytes 2 and 3. The parameters are passed to the FEP by subroutine FEPASS, below.

SETFEP (SBOØ54ØØ-SBOØ69ØØ)

Subroutine SETFEP is called by FEPASS and LDFEP (below) to set up for a FEP load of either a single program location or an entire program. SETFEP calls RESFEP (Section 2.6.2) to reset and initialize the FEP, then sets up a table which contains the FEP program (DED.FEPA) and control (DED.FEPB) DIO addresses associated with control bytes sent to the FEP to effect the load. (See FEPLOC, below.)

FEPASS (SBOØ71ØØ-SBO112ØØ)

Subroutine FEPASS is called by PASSF (above), by LDFEP (below), and by the mode switching routine BTALL (Section 2.9.2.2), to pass a count, a FEP counter number, and a constant to the FEP program location (2-5) specified in R12. This is a generalized mechanism for passing variable information to the FEP, devised by J. R. Winkelman. It is necessary to pass parameters in this manner, because only the FEP program memory can be written to from the Interdata via the DIO bus. Once the FEP program memory has been loaded, no other information other than the program starting address can be sent to the FEP.

FEPASS constructs an actual FEP program step from MD.FEPT for location 5, using FEP counter C1. The program steps for locations 2, 3, and 4 are constructed from the counts and constants in MD.FEPP (above), using counters C2, C3, and C2, respectively. The counters are chosen with consideration given to the standard HF Radar FEP programs. See command FEP (Chapter 8) for details about how the passed FEP parameters are currently used.

When the FEP program step is thus constructed, FEPASS calls SETFEP (above) to set up for the load, calls FEPLOC (below) to write the program step to the location specified by R12, and then calls RESFEP (Section 2.6.2) to reset the FEP after the load.

LDFEP (SBO11400-SBO15300)

Subroutine LDFEP is called by routine BOOT (Section 2.6.1) in response to the BOOT FEP[,xxx] command. LDFEP is also called during mode switching by BTALL (Section 2.9.2.2), if the new sounding mode requires a different FEP program than was required by the old mode.

LDFEP is entered with the address of the specified FEPMEM file extension ("xxx") in R3. If none was specified, R3 points to "TWO", the default FEPMEM extension. LDFEP attempts to assign LU6 to the FEPMEM file on SYS1, and if it cannot, the BOOT FEP, MGET, RUN, or CONT command that initiated the FEP load is rejected with ERR 20. Otherwise, the FEPMEM file extension ("xxx" or "TWO") is saved in MD.FEPX (Section 7.1), specifying the FEP program for the current sounding mode. If RUN or CONT called LDFEP, MD.FEPX is merely copied back to itself.

LDFEP then calls SETFEP (above) to set up for the FEP program load and reads successive 12-byte records from the FEPMEM disc file until EOF is sensed. Each 12-byte record is written to the FEP location specified in the record with a call to FEPLOC, below.

After the FEP program has been thus loaded, LDFEP calls FEPASS (above) to write the MD.FEPP specifications to locations 2-4 and the thresholding level (MD.FEPT) to location 5. Notice that the FEP command specifications override any program steps 2-5 from the FEPMEM program. This is a price paid for automatic mode switching.

Only after the program load has succeeded does LDFEP copy the FEPMEM extension to FEPX (Section 2.6.4), indicating the program currently in the FEP. Thus if a direct BOOT FEP,xxx fails, the load will be tried again at run time if MD.FEPX does not match FEPX. (See BTALL, Section 2.9.2.2.)

FEPLOC (SBO155ØØ-SBO2Ø4ØØ)

Subroutine FEPLOC is called by both FEPASS and LDFEP (above) to write a single FEP location to FEP program memory via the DIO bus. The Interdata memory address of the FEP location is in R11 when FEPLOC is entered.

The program location is sent to the FEP, byte by byte, in reverse order to how the bytes appear in the location image. After each byte is sent, the FEP is sent control bytes that cause the FEP to load the data byte into an internal 88-bit register and to shift the register. When the 88-bit register in the FEP has the full program step, or location, the FEP location number is sent with control that causes the FEP to put the program step to the program memory.

CALL task FEPEKO (Section 1Ø.2.16) uses an identical procedure for loading the FEP and for passing variable information to the FEP.

FEP-related Tables (SBO2Ø6ØØ-SBO256ØØ)

This section contains tables, etc., related to the FEP loading procedures.

LOCN is the template of the program step sent to FEP loactions 2-4 by the FEP command. Bytes Ø, 3, 4, 9, and 1Ø are set up by FEPASS before LOCN is sent to the FEP.

LOC5 is used like LOCN, but for location 5 (FEP thresholding) only. Bytes 9 and 1Ø are set up by FEPASS before LOC5 is sent to the FEP.

CNMBR is the table of FEP counter numbers by location. The FEP counters associated with locations 2-5 were chosen to accomodate the current standard HF Radar FEP programs.

SCTRL through FDIOA are control tables used in loading a FEP program location. Refer to the listings for details about how they are used.

ASFMEM is the SVC7 parameter block for assigning LU6 to SYS1:FEPMEM.xxx, and RDFPB is the SVC1 parameter block for reading FEPMEM.xxx. BUFR is the 12-byte buffer used by RDFPB.

BTTSG (SBO25800-SBO30200)

Subroutine BTTSG is called by the BOOT routine (Section 2.6.1) in response to the BOOT TSG command. BTTSG is also called during mode switching by BTALL (Section 2.9.2.2), if TSGX (Section 2.6.4), the extension of the TSGMEM disc file last written to the TSG, does not match MD.TSGX of the new sounding mode.

BTTSG is entered with the address of the specified TSGMEM file extension in R3, and like LDFEP, the extension used is "TWO", if none was specified by the BOOT command. BTTSG attempts to assign LU6 to the TSGMEM file on SYS1, and if the assignment fails, the BOOT TSG, MGET, RUN, or CONT command that invoked BTTSG is rejected with ERR 21. If the TSGMEM file is found, the file extension is saved in MD.TSGX, specifying the TSG program for the current sounding mode. BTTSG then calls RSTSG (below) to reset the TSG, then reads a byte from DIO.TSGA, in case the installed TSG PROM contains the self-test routines. (See Appendix A, dated 1983 November.) The error condition byte from DIO.TSGA is saved for possible later examination. BTTSG then loads overlay 3-02 (Section 2.10.3) for subroutine TDLAYS. Overlay 3-02 also contains the TSGMEM read buffer.

BTTSG makes calls to TSGPG (below) to read records 0, 2, and 3 from the TSGMEM disc file and to write these page images to the TSG via the DIO bus. Page 1 of the TSG contains stacks and other storage internal to the TSG operation and is not written to the TSG. Record 1 of the TSGMEM file is likewise unused, but the TSGMEM file is made 4 records long for programming convenience.

After TSGPG has been called for page 0, byte 255 of the zero-page is checked. If this byte is 1, the error condition byte, read earlier from DIO.TSGA, is examined. Otherwise, the byte is ignored. If the error condition byte is examined and found to be non-zero, BTTSG logs the message, ERROR ØØØn AT C4B2. The error, n, is one of the following:

1 = ROM checksum error.
2 = Page Ø checkerboard pattern write-read failure.
3 = Page Ø addressing failure.
4 = Page 1-3 pattern failure.
5 = Page 1-3 addressing failure.

The self-tests are made in 1-5 order. The first error detected stops the self-testing procedure, and only that error is reported.

Subroutine TDLAYS (Section 2.1Ø.3) is called to compute the TSG delays for table TSGDLY for scenarios 1-7, and to establish tables ZPCTRL and TSGREP. (See TSG-related Tables, Section 2.6.4.) Tables TSGDLY and ZPCTRL are used by TSGASS (Section 2.8.2) to set up the pulse palette; TSGREP is consulted by CLOCK (Section 2.1Ø.1) for time-keeping.

After the TSG has been successfully loaded, BTTSG copies the TSGMEM file extension to TSGX (Section 2.6.4) to identify the last TSG program written to the TSG. TSGX is checked against MD.TSGX by BTALL (Section 2.9.2.2) during mode switching. BTTSG then loads overlay 1-Ø1 and calls TSGASS (Section 2.8.2) to set up the TSG-associated values in the pulse palette.

TSGPG (SBO3Ø4ØØ-SBO392ØØ)

Subroutine TSGPG is called by BTTSG (above) with the TSG page number (Ø, 2, or 3) in R7; this number is also the record number of the TSGMEM disc file that contains the binary images of the TSG pages. R5 at entry to TSGPG is Ø, if the command BOOT TSG was given. If the command form was BOOT TSG,xxx, R5 is 1. If R5 = Ø at TSGPG entry (default boot), the TSGMEM record (R7) is neither read, nor is the TSG page (R7) written. Instead, TSGPG reads the page specified by R7 into the buffer in 3-Ø2, verifying the read with an exclusive OR (XOR) ckecksumming procedure, described below.

If R5 = 1, the TSGMEM record (R7) is read from disc and then written to the TSG page specified by R7. The page is then read back to verify the write. Both the TSG page write and read-back are executed with handshaking between the Interdata and TSG computers. Both computers compute an XOR checksum byte which figures into the verification of the data transfer.

If the page write handshaking times out, error 8224 is logged; if the page read handshaking times out, error 8225 is logged. If a byte read back from the TSG is not the same as the corresponding byte when written, an error message is logged that displays the byte as written in the high order byte and the byte as read back in the low order byte. As a 256-byte page is read back from the TSG, TSGPG computes a running XOR checksum. The last byte from the TSG is its version of the same checksum for the page. The XOR of this byte and the running XOR checksum of the page should result in Ø. If it does not, an error message is logged displaying the final, erroneous checksum. For any of the above errors, DEVERR (Section 2.6.2) is called to count the TSG error and to flag it in MD.STAT (Section 7.1). Also, if any error is detected, RSTSG (below) is called to reset the TSG.

<u>RSTSG</u> (SBO394ØØ-SBO41ØØØ)

Subroutine RSTSG is called by BTTSG to reset the TSG before loading pages Ø, 2, and 3, and is called by TSGPG if any loading error occurs.

RSTSG sets the RESET bit (LSB) of DED.TSGB (Section 7.9) high (1), then after 1ØØ ms sets it low (Ø). RSTSG then waits for 8ØØ ms for the TSG to be loaded from its PROM and to run its self-test routines, if any. The reset thus loads the zero page with the default timing scenarios.

<u>RWTSG</u> (SBO411ØØ-SBO429ØØ)

Subroutine RWTSG is called by TSGPG to issue a write-page then a read-page command to the TSG. A read/write command is recognized by the TSG when the sign bit of the command is Ø. A read command is flagged with hex 4Ø; the page number is in the low order 6 bits, right-justified.

RWTSG is called with the read/write command byte in R7. This command byte is sent as part of the TSG Non-Maskable Interrupt (NMI) sequence: the command byte is sent to DIO adddress DED.TSGA (Section 7.9). The DED.TSGB NMI bit (2nd least significant bit) is taken high (1), then after about 1Ø micro-seconds, the NMI bit is taken low (Ø). This causes the NMI in the TSG, which then reads the command byte via one of its Peripheral Interface Adapter (PIA) ports.

WT25 (SBO43100-SBO45300)

Subroutine WT25 conducts the handshaking with the TSG for page reads and writes via the 2nd least significant bit of DIO address DED.TSGB (Section 7.9). This bit is expected to be alternately low and high to signal DATA READY and DATA SEEN. WT25 waits for the expected state and exits, skipping two bytes, if the wait times out after about 10 ms. The skip exit results in error 8224 or 8225, above.

PACE (SBO46500-SBO52800)

Subroutine PACE times the four strappable TSG paces for command TICK in overlay 0-08 (Section 2.7.9). The four TSG paces are normally strapped on the TSG board to 10, 20, 100, and 1000 milliseconds, but there are other alernatives. If other than standard paces are selected, or if the standard paces are selected in non-standard order, command TICK should be issued after SOUNDER start up, before sounding begins, in order for SOUNDER's time-keeping to be correct.

PACE times each of the four TSG paces by running the TSG at each pace for one second, as determined by a counter, CLKCNT, used by the MT2 OS precision interval timer. Because the Interdata and the TSG clocks are asynchronous, PACE allows for an uncertainty of ± 1 in the number of TSG DONE interrupts per second. A pace of less than 5 Hz is assumed to be 1 Hz.

The TSG paces are saved in ms, as measured by PACE, in table TPACE, and are saved in 10-ms intervals per pace in table TENMS (Section 2.6.4) for CLOCK (Section 2.10.1). The SYS command display (Section 2.9.3.1) displays the TPACE entries as the "HARD TICK" and the TENMS entries, times ten, as the "SOFT TICK". The software tick is the hardware tick rounded up to the next 10 ms, if necessary, because SOUNDER's time-keeping is in 10-ms intervals.

GTDIO (SBO53000-SBO58500)

Subroutine GTDIO, the DIO equivalent of LDFEP and BTTSG, is called by the BOOT routine (Section 2.6.1) in response to the BOOT DIO command, and is also called by BTALL (Section 2.9.2.2) during mode switching, if the extension of the currently loaded DIO disc file (DIOX, Section 2.6.4) does not match MD.DIOX (Section 7.1) of the newly installed sounding mode.

Like LDFEP and BTTSG (above), GTDIO is entered with the address of the specified SYS1:DIO disc file extension in R3. The extension "TWO" is used if none was specified by the BOOT DIO command. GTDIO tries to assign LU6 to SYS1:DIO.xxx, and if it fails to do so, the BOOT DIO, MGET, RUN, or CONT command that invoked GTDIO is rejected with ERR 22. If the assignment succeeds, the file extension is saved in MD.DIOX, associating the DIO control tables from SYS1:DIO.xxx with the current sounding mode.

The DIO disc file, written by DIOGEN (Section 1Ø.2.4), is then read into SOUNDER table, DIO (Section 2.6.4). Although DIOGEN writes sectors Ø-8 of a DIO file, only sectors 6-8 are read by GTDIO. (See Section 7.9 for details.) After the DIO control tables are thus established, the DIO file extension is copied to DIOX (Section 2.6.4). If MD.DIOX of the next sounding mode does not match DIOX, the DIO control tables are reestablished.

GTDIO next sets flags NOTSG and NOADC (Section 2.6.4) to Ø, if the TSG (DED.TSGA) and the HKADC (DED.HKAD) are configured into the DIO system just established. If either of these devices is not known to the DIO control tables, their device-is-dead flag is set to 1, allowing debugging runs without the devices being used. These flags do not change when the IMA loop is entered at GETADV (Section 2.6.1) as they do if set to -1 by command NO (Section 2.7.8).

Finally, GTDIO sizes the DIO.PTP area and flags the size word with 9ØØØ (hex). The DIO.PTP area, which contains the pulse palette extensions, is associated with each sounding data set by MANAGER.

2.9.2.2. SNDRSTUP [DCW]

Module SNDRSTUP contains the routines for transmitter control commands TXON, HVON, HVOF, DLON, DLOF, LPAT, and HPAT. (The TXOF command routine is memory resident in SNDRADVC, Section 2.6.1.) Subroutine RESTX resets the transmitter, and GTMAP reads a protected frequencies map from the disc. SETUP and subroutine BTALL set up for a sounding after command RUN or CONT is given, and finish the sounding mode switching begun in overlay Ø-Ø1 (Section 2.7.2).

TXON (SSTØ25ØØ-SSTØ42ØØ)

Command TXON is given to turn on the transmitter filaments, beginning the 9Ø second warm up period, after which the high voltage (HV) can be turned on.

The TXON sequence begins with a call to RESTX (below) which resets the transmitter, clearing any non-current error conditions. The current state of the dummy load (DL) is saved. TXON then fully attenuates the transmitter, selects the dummy transmitter load, and turns off the filaments and HV. The transmitter is again reset, and after a delay of 9Ø ms (DLYMT2, Section 2.6.2), the filaments are turned on. Finally, the DL is returned to its original state.

<u>HVON</u> (SSTØ43ØØ-SSTØ77ØØ)

The HVON command is given to turn on the 1Ø kW transmitter high voltage, enabling sounding at full power. If the TXON command has not been given, or if the 9Ø second warm up period has not elapsed (HV HOLD true), the HVON command is rejected with ERR 8.

After resetting the transmitter with a call to RESTX (below) and saving the current state of the DL, HVON selects the dummy transmitter load and fully attenuates the transmitter. Then, after a delay of 9Ø ms, the transmitter HV is turned on. After another delay of 1 second, the SET bit of DED.TXCT (Section 7.9) is taken low (Ø), then after a delay of 1Ø ms, SET is taken high (1). With the HV thus turned on, the DL is returned to its original state.

<u>HVOF</u> (SSTØ78ØØ-SSTØ84ØØ)

Command HVOF is given to turn off the transmitter HV during periods of inactivity. When the transmitter HV is turned off, the states of the filaments and the DL are preserved. The 9Ø second warm up period is not required before a subsequent HVON command.

<u>DLON</u> <u>and</u> <u>DLOF</u> (SSTØ85ØØ-SST115ØØ)

Commands DLON and DLOF are given to select the dummy and the antenna transmitter loads, respectively. The default state at SOUNDER start up is DLON. If a sounding is run before the DLOF command is given, the message, "DL ON", is logged. For either command, the states of the filaments and the HV are unchanged.

TXATLP and TXATHP (SST11700-SST13300)

Commands LPAT and HPAT are given to specify the transmitter attenuation for low and high power sounding, respectively. The default low power attenuation (MD.TXLP) is 6 dB; the default high power attenuation (MD.TXHP) is 9 dB. The transmitter attenuations can be set from 3-31 dB, in 1 dB steps.

The DST entry (Section 7.9) for the transmitter attenuator (DED.TXAT) is set at sounding start by RUNST (Section 2.9.1.3).

RESTX (SST13500-SST17000)

Subroutine RESTX is called by command routines TXON and HVON (above) as part of the filaments and high voltage turn on sequences. RESTX is also called by SNDRINIT (Section 2.6.5) to initialize the transmitter at SOUNDER start up.

As part of the turn on sequences, RESTX takes the SET bit of DED.TXCT (Section 7.9) high (1) and the RESET bit low (0). This resets the transmitter control, clearing any non-current fault conditions. After a 10 ms delay, the RESET bit is taken high (SET remains high), ending the transmitter reset sequence.

RESTX leaves the states of the filaments, HV, and DL unchanged.

GTMAP (SST17200-SST21900)

Subroutine GTMAP is called by subroutine GETMAP (Section 2.6.1) to get the protected frequencies map for the current sounding mode. GETMAP is called whenever the protected frequencies map needs to be loaded or checked.

GTMAP first sets FRQMAP (Section 2.6.4) to protect all frequencies, in case the map load fails, and then attempts to assign LU6 to SYS1:xPROT.TWO. (The "x" is from MD.MAP of the current SCT, Section 7.1.) If the assignment fails, the MGET, RUN, or CONT command that invoked GTMAP is rejected with ERR 23, leaving all frequencies protected.

Otherwise, GTMAP reads the xPROT.TWO disc file into FRQMAP, then calls CKMAP (Section 2.6.2) to verify that the map as read from disc is intact. After converting the frequency bands from 32-bit floating point to 32-bit fixed Hz/1ØØ values, GTMAP again calls CKMAP to recompute the map check bytes. The protected frequencies map is thus readied for frequency selection and sounding.

The xPROT.TWO disc files are created by CALL task PROTF (Section 1Ø.2.3).

SETUP (SST221ØØ-SST3Ø5ØØ)

Routine SETUP is entered from overlay Ø-Ø1 (Section 2.7.3) after overlay 2-Ø1 is loaded as a result of commands RUN or CONT. SETUP performs all pre-sounding initialization that can be done before the SPC overlays are loaded by SNDOV (Section 2.6.1). The initialization done here is required only after a RUN or CONT command. Pre-sounding initialization required before every sounding of an SPH series is done by NXTSND (Section 2.9.1.1).

SETUP first checks MD.PPS (Section 2.7.9), and if it is Ø, a PSEQ command error was left uncorrected (Section 2.7.4); the RUN or CONT command is rejected with ERR 16. If MD.PPS is not Ø, MD.OPOB, the offset to the echo data in PICKER's output, is computed as 2 + (MD.PPS * PCT size). MD.N is set (redundantly) to MD.PPS * 2. For complementary coded soundings and for soundings that have TSG NOPs embedded within the pulse sets, PICKER will reset MD.OPOB to its MAX.OPOB (Section 3.1), in order that the recorded data may be properly unpacked.

SETUP next scans the RUN command argument list for a specified run time of the form, mm, or mm:ss. (See RUN, Chapter 8.) If the time format is wrong, the RUN command is rejected with ERR 1. Otherwise, the seconds to sounding start in R5 will be checked by SNDOV, after the TSG clock has been read. If no run time is specified, R5 is set to MD.CDN.

SETUP next calls subroutine BTALL (below) to guarantee that the DIO control tables, FEP, TSG, protected frequencies map, and PICKER task are set up as specified by the current SCT.

Next, SETUP checks if MD.PPU was specified by the AGC command (Section 2.7.9). If not, MD.PPU is set to the default which is MD.PPS for each receiver channel under AGC; the AGC will be updated every pulse set.

SETUP then converts the mV values in MD.PKLO and MD.PKHI to ADC counts for LOAGC and HIAGC, respectively, the receiver dynamic range limits for the AGC system. (See UPAGC, Section 2.9.1.3.)

The sounding frequency limits, MD.SFRQ and MD.EFRQ, are copied to SCT.SFRQ and SCT.EFRQ, respectively. It was planned to set SCT.EFRQ to the actual ending frequency, in case of sounding abortion. However, this would lead to some confusion if MD.NSET > 1 for S-class soundings.

TSGBFR, the TSG command byte, is initialized to CØ, in anticipation of executing the countdown with the 1ØØ ms TSG NOP start. (See STTSG, Section 2.9.1.1.) Notice that TSGBFR is also used by PACE, Section 2.9.2.1.

Finally, SETUP checks MD.CALL, and if its first byte is blank (hex 2Ø), no Sounding-mode CALL task has been specified for run time. Otherwise, overlay 1-Ø3 (Section 2.8.4) is loaded, and GTCALL is called to swap GRAPHER for the specified CALL task. If the CALL task cannot be found, the RUN or CONT command is rejected with ERR 9. Otherwise, SETUP waits until the CALL task sends its TFF request queue to SOUNDER. If bit 13 of TFF + 4 is set, the sounding is conducted using all of SOUNDER's PCT/EKO buffers. (See GTCALL, ANSCALL, and ENDCALL, Section 2.8.4.)

All initialization for sounding that can be done before loading the SPC overlays and reading the TSG clock is now complete; control goes to SNDOV (Section 2.6.1), where sounding set up is completed.

<u>CMPEXT</u> (SST3Ø7ØØ-SST321ØØ)

Subroutine CMPEXT is called by BTALL (below) to compare 3-character disc file extensions during sounding mode switching to see if the DIO, FEP, and TSG are set up as specified by the new sounding mode. If a file extension in the SCT (R3) matches that saved for the previous mode in SNDRTBLS (R13), CMPEXT exits normally; otherwise, CMPEXT exits skipping two bytes.

<u>BTALL</u> (SST322ØØ-SST396ØØ)

Subroutine BTALL is called by SETUP (above), after a RUN or CONT command is given; by LOADM (Section 2.7.2), when command MGET is given; and at SOUNDER start up (Section 2.6.5), to set up the current sounding mode.

BTALL first calls GETMAP (Section 2.6.1), because neither frequency selection nor sounding can continue unless the protected frequencies map is intact.

BTALL next calls CMPEXT (above) to compare MD.DIOX (Section 7.1) and DIOX (Section 2.6.4). If the new sounding mode requires DIO control tables different from the old mode, BTALL calls GTDIO (Section 2.9.2.1) to get the new DIO tables specified by MD.DIOX. With the DIO control tables established, BTALL copies the receiver 1 and 2 input specifications in the pulse palette (PD.1RIN and PD.2RIN), set by command ANT (Section 2.7.6), to the DIO.PTP pulse palette extensions for DED.1RIN and DED.2RIN (Section 7.9). The ANT command specifications thus override the DIOGEN specifications for the receiver inputs. (See DIOGEN, Section 1Ø.2.4.)

Notice, that because the DIO system is entirely driven by the DIO control tables, SOUNDER table DIO must be established before the FEP or TSG can be set up.

At this time the low power RF filters specification to the BDW command (Section 2.7.1Ø) takes effect. The filters bit (mask 8) is extracted from MD.BDW, positioned, and ORed with the two bandwidth commands. These values are used to set the DST.PDTA entries for DED.1BDW and DED.2BDW (Section 7.9). The filters bit is also used to set the DED.SYNS DST entry.

Next, CMPEXT is called to see if a different FEP program is needed for the new sounding mode. If so, LDFEP (Section 2.9.2.1) is called with R3 pointing to MD.FEPX. If the FEP is not reloaded, FEPASS (Section 2.9.2.1) is called to pass the FEP command parameters to FEP program locations 2-5.

Likewise, the TSG is booted by BTTSG (Section 2.9.2.1), if the new sounding mode demands; MD.REPS is set from table TSGREP, and BTALL returns to the caller.

Finally, the PICKER task specified by MD.PIKT is loaded, unless it already resides in PICKER's memory partition.

2.9.3. Overlay 2-Ø2 [DCW]

Overlay 2-Ø2 contains two modules. Module SNDRSYST (Section 2.9.3.1) contains the SYS command routine and MCON3, the final segment of the MGET command display.

Module SNDRDISP (Section 2.9.3.2) contains the two display subroutines, WRCON and PAGE.

2.9.3.1. SNDRSYST [DCW]

Module SNDRSYST contains nothing but display routines for commands SYS and MGET. Devoting this much memory to display routines is a luxury affordable only because this code shares the large overlay area 2 with the SPC code and the boot routines.

SDISP (SSYØ18ØØ)

The size of overlay area 2 is established by module SDNRSYST. Label SDISP is equated to the size of module SNDRDISP. SNDRSYST is padded so that the total size of SNDRSYST and SNDRDISP is 256Ø bytes, or exactly 1Ø disc sectors.

SYST (SSYØ2ØØØ-SSYØ25ØØ)

Routine SYST is entered when command SYS is given. SYST calls subroutine SYSTAT (below) to display the HF Radar system status. COUNSEL invokes SYST and the SYS display at HF Radar start up.

SYSTAT (SSYØ26ØØ-SSY154ØØ)

Subroutine SYSTAT is called by the SYS command routine (above) and by the TICK command routine (Section 2.7.9). SYSTAT displays the overall status of the HF Radar software and hardware systems.

The date and time displayed are from the MT2 OS clock, which is set from the TSG clock at HF Radar start up. The site code is that set by CALL task LOCAL (Section 1Ø.2.2). The task revisions are from the SCT; each task sets its own revision when the task is first started. The minor revision is displayed in hexadecimal.

The soundings per hour, set by command SPH, is Ø if inactive or set to free run. Soundings to run (SN) will be 65535 (-1) if no limit has been set or if SN has been satisfied. The frequency generation algorithm (FGA) and DIO generation algorithm (DGA) are currently always 2.

The hardware and software ticks are explained in Section 2.9.2.1, under PACE. If the hard and soft ticks are not identical, the true value of the hard tick should be supplied to off-line programs, possibly as one of the user-defined values in SCT.USER. (See CALL task LOCAL, Section 1Ø.2.2.)

The hardware status is not honest until a sounding has been run, because the conditions of the HV, DL, and synthesizer are from MD.STAT (Section 7.1), set by RUNST (Section 2.9.1.3) at sounding start. The receiver DC offsets are computed by the FEP during the receiver calibration cycle and should not drift significantly. (See WTEKO, Section 2.9.1.1.)

The hardware error counters (ERRxxx, Section 2.6.4) are zeroed at HF Radar start up and are maintained for the duration of the sounding session, that is, until CANHF is typed. An error detected during a sounding is not only counted, but the appropriate bit is set in MD.STAT. (See DEVERR, Section 2.6.2.) MD.STAT should be examined by off-line programs to determine if the sounding was error free. (See MD.STAT, Section 7.1, for more details about the hardware error counters.)

<u>MCON3</u> (SSY156ØØ-SSY235ØØ)

MCON3 is entered from MCON2 in overlay 1-Ø5 (Section 2.8.6) and is the final segment of the MGET command display.

MCON3 displays everything from the BOOT TSG line to the end of the display. The last sounding status is not honest for any of the permanent modes unless the mode has been run during the current sounding session, because a sounding session begins with the backup copies of permanent modes. (See SNDRINIT, Section 2.6.5.)

2.9.3.2. SNDRDISP [DCW]

Module SNDRDISP contains the two display subroutines, WRCON and PAGE. If either of these subroutines is modified, SDISP in module SNDRSYST (Section 2.9.3.1) must be updated, so that the size of overlay area 2 is correctly established.

<u>WRCON</u> (SDIØ18ØØ-SDI424ØØ)

Subroutine WRCON is called to format lines of text and to write these formatted lines to LU1, normally the console (CON:). This subroutine is a revision of the WRCON loaded with each of the CALL tasks. Functionally the two subroutines are identical, and Section 1Ø.5 describes this subroutine exactly, except for error handling. This subroutine calls

ERROR (Section 2.6.2), instead of making an SVC3 call to the MT2 OS, in case of a formatting or I/O error.

PAGE (SDI42600-SDI43800)

Subroutine PAGE is called to clear the TEKTRONIX 4012 screen and to home the cursor. If LU1 is assigned to the TELETYPE Model 40 line printer (LPR:), PAGE effects a form feed, or page eject. PAGE calls WRCON to write hex bytes 1B and 0C to LU1, then delays 1 second for the LU1 device to respond.

2.10. Overlay Area 3 [DCW]

Overlay area 3, like areas 0 and 1, is 512 bytes, or 2 disc sectors long. SOUNDER has two command scan lists, one available during sounding, the other between soundings. Both lists are in area 3 overlays and have the same memory address. The three area 3 overlays are described in order in Sections 2.10.1 through 2.10.3.

2.10.1. Overlay 3-00 (SNDRO300) [DCW]

Overlay 3-00 is loaded by SNDOV (Section 2.6.1) following a RUN or CONT command, and remains in memory for a single sounding or for an SPH series of soundings.

Overlay 3-00 contains SOUNDER's SPC loop command scan list and the CLOCK subroutine, called from the SPC loop only.

OV3 (S3002500)

Label OV3 establishes the address of overlay area 3 for the overlay loader (Section 2.6.1).

SPC Commands (S3002600-S3004600)

Labels ADADV and SMNEM establish the addresses of SOUNDER's command routine address table and command mnemonic, or scan, list, respectively, for both the SPC command set in 3-00 and the IMA command set in 3-01. COUNSEL is given the address of SMNEM at SOUNDER start up and is indifferent to the fact that SOUNDER's commands are overlayed, provided the command scan lists always start at the same location, SMNEM.

ADADV for both 3-00 and 3-01 is at OV3 + 0. ADVICE (Section 2.6.1) uses the contents of ADADV as the base address of the command routine address table, whichever command overlay occupies area 3. The table structure is described in Section 2.1Ø.2.

The SPC command set contains only AB, TX, RF, and IF. The routines that execute these commands are all memory resident in SNDRADVC (Section 2.6.1). Jim Winkelman points out that, on the worst of days, it is possible that an IMA command could be given at a time that would cause COUNSEL to scan SOUNDER's command list while the IMA command overlay is in the process of being read from disc. This can probably be avoided by not typing any IMA commands until the end-of-sounding message has been logged to the console.

CLOCK (S3ØØ48ØØ-S3Ø169ØØ)

Subroutine CLOCK is called each time through the SPC loop by WTTSG (Section 2.9.1.1) to maintain SOUNDER's software clock. CLOCK is also called by DLYTSG (Section 2.6.2), the SPC delay subroutine.

SOUNDER's time-keeping counters are initialized when the TSG clock is read by RDTIME (Section 2.8.1). Some or all of these counters are incremented or decremented for each TSG start. The time-value of each TSG start is determined by the TSG command byte in TSGBFR (Section 2.6.4). Bits 1 and 2 (Ø = MSB) of the command byte are the index into table TENMS, and bits 3 and 4 are the index into table TSGREP. (See TSG-related Tables, Section 2.6.4.) The time-value of any TSG start is the product of the TENMS and TSGREP values specified by the TSG command byte. A TSGREP entry of Ø is taken as a repeat of 256. This product is added to PCT.TIK of NXTPCT, preservimg the sign bit, or the first-of-set flag. (See INPSET, Section 2.9.1.2.) Thus, PCT.TIK of the PCT associated with each pulse contains the number of 1Ø ms intervals elapsed from sounding start, modulo 32768. (PCT.TIK wraps around every 5 minutes, 27.68 seconds.)

The time-value of each TSG start is also added to FRSEC, the fractional second in 1Ø ms intervals. At CKSEC, after updating FRSEC, CLOCK subtracts 1ØØ, or one second's worth of 1Ø ms intervals, from FRSEC to determine if at least one full second has elapsed. If not, CLOCK exits through CLKEX, below. Otherwise, one is added to ET.SEC, the number of seconds elapsed since midnight, 1 January 1977, and to HRSEC, the number of seconds into the current hour. If HRSEC is now 36ØØ, an hour has elapsed; one is added to HOUR, and HRSEC is zeroed. If HOUR is now 24, a day has passed, and HOUR is reset to Ø.

When a second has elapsed, one is subtracted from ET.TNXT, the number of seconds until the next sounding is to begin. If after ET.TNXT has been decremented, ET.TNXT = ET.CDN, it is time for the countdown to sounding to begin. Otherwise, another pass is made through CKSEC (above), in case the time-value of the last TSG start was greater than one second.

If ET.TNXT = ET.CDN, CLOCK checks NXTFIX (Section 2.6.4). If NXTFIX < Ø, no sounding is currently in progress, a new sounding can therefore begin, and control goes to CKSYNS, below. Otherwise, CLOCK checks SCT.SPH (Section 2.7.8). If a free run SPH series is in progress (SCT.SPH = Ø), one is added to ET.TNXT, deferring the countdown for one second. Otherwise, the operator has specified an impossible SPH schedule. A message is logged displaying the current value of NXTFIX, or how far the current sounding progressed before the next one was scheduled to begin; SCT.SPH is zeroed, selecting free run; and the countdown is deferred by one second, as described above.

At CKSYNS, it has been determined that the countdown to sounding can begin. The status of the synthesizer (DED.SYNS, Section 7.9) is read, and unless the three synthesizer flags are all high (1), an error message is logged displaying the status as read. If the operator has inadvertantly switched the synthesizer to local (thumbwheel) control instead of remote (SOUNDER) control, error 5 is logged. The operator has the time of the countdown to abort, unless the sounding was intended to be under local synthesizer control.

The sounding start time (SCT.STS) is now set to ET.SEC + MD.CDN, SCT.STM is zeroed, indicating a start on the even second, and the sounding is counted by adding one to SCT.SNO.

CLOCK's final task at countdown start is to send the SØ4 Q-block to PICKER (or a Sounding-mode CALL task) with SØ4 + 4 set to two times the sounding mode flag (SCT.MODE, Section 7.1). The SØ4 status, SØ4 + 2, is initialized to -1ØØØ (hex), and will be checked by NXTSND (Section 2.9.1.1) at the end of the countdown to confirm that all down-stream tasks have acknowledged the SØ4 Q-block.

At CNTDN, if a countdown is in progress, the value in ET.CDN is decremented and displayed in the hex panel lights. NXTSND monitors ET.CDN, and when ET.CDN = Ø, the countdown is finished, and the sounding begins.

The CLOCK subroutine always exits through CLKEX. Between soundings, PULSE is kept at -1, but during sounding, PULSE is incremented each time through CLOCK and kept modulo 32768, or not < Ø. PULSE is rewound to 2, because PULSE = 1 has special meaning to SETDIO (Section 2.9.1.1).

RUNEND (S3Ø171ØØ-S3Ø209ØØ)

Subroutine RUNEND is called at sounding end after all post-sounding activity has finished (Section 2.9.1.1). RUNEND logs the number of lost pulses (MD.PLST) and lost echoes (MD.ELST), if these values are greater than Ø, and logs the sounding end message.

2.1Ø.2. Overlay 3-Ø1 (SNDRO3Ø1) [DCW]

Overlay 3-Ø1 contains the tables for the IMA loop command set. Overlay 3-Ø1 is in overlay area 3 whenever ADVICE is called by IMALOOP (Section 2.6.1).

The first 1/2-word of 3-Ø1 contains the address of the command routine address table, ADV1. The address of this 1/2-word is established for ADVICE by ADADV in overlay 3-ØØ (Section 2.1Ø.1). The start of the command mnemonics is similarly established by SMNEM in 3-ØØ.

Each command mnemonic terminates with a null (hex Ø) byte. Required characters are flagged with hex 8Ø. The command mnemonic list terminates with two consecutive null bytes.

The command routine address table (ADV1) entries correspond one-to-one with the command mnemonics. Each ADV1 table entry contains 4 bytes. Bytes Ø and 1 contain the address of the routine that executes the command. This routine may be either memory resident or in an overlay in areas Ø, 1, or 2. If the routine is memory resident, bytes 2 and 3 of the ADV1 table entry are -1. Otherwise, byte 2 contains the overlay area number, and byte 3 contains the overlay number.

2.1Ø.3. Overlay 3-Ø2 (SNDRO3Ø2) [DCW]

Overlay 3-Ø2 is called only by subroutine BTTSG (Section 2.9.2.1). Overlay 3-Ø2 contains the subroutine that computes the TSG delays; 3-Ø2 also contains the TSG page buffer used by BTTSG.

TDLAYS (S32Ø15ØØ-S32Ø63ØØ)

Subroutine TDLAYS is called by BTTSG with the TSG zero page in BUFR, below. TDLAYS first copies the first 26 bytes of the zero page to contiguous tables ZPCTRL and TSGREP (Section 2.6.4). Values from ZPCTRL are copied to the pulse palette by TSGASS (Section 2.8.2); subroutine CLOCK (Section 2.1Ø.1) uses TSGREP to compute the time-value of each TSG start.

TDLAYS next computes the TSG delays for timing scenarios 1 through 7 for table TSGDLY (Section 2.6.4). The TSG delay for the NOP scenario Ø is always Ø. The TSG delay for a given scenario is measured in microseconds as the time from the center of TX KEY active to the start of the FEP WRITE ENABLE data acquisition window. (See a scenario displayed with SNRIO, Section 1Ø.2.1Ø.) The width of any FEP WRITE ENABLE window that preceeds the TX KEY is subtracted from the TSG delay. If a scenario does not key the transmitter, its TSGDLY entry is Ø.

BUFR (S32Ø65ØØ)

BUFR is the buffer used by BTTSG (Section 2.9.2.1) for reading a 256-byte page from a TSGMEM disc file and for reading a page from the TSG with its checksum byte.

2.11. SOUNDER's Drivers [DCW]

SVC1 calls within SOUNDER's SPC loop (Section 2.9.1.1) that start and wait for the TSG, FEP, and EKO devices, invoke drivers at the MT2 OS level. Because these control non-standard devices, they were written at SEL and have been incorporated into the MT2 Executive (EXEC) following conventions described in detail in Interdata's OS/16 MT2 Guide to Writing Drivers. (Although the FEP is no longer under control of SOUNDER, the FEP driver remains a part of the EXEC and is invoked by PIPLN, Section 1Ø.2.15.)

A Device Control Block (DCB) is associated with each driver. The DCB contains flags, routine addresses, and register storage required by the driver to operate the device. It also contains short blocks of code executed when the interrupt service routines

(ISR) are entered and exited.

Each of the three drivers described below contains four sections. The first section (U4xDVR) is the initialization routine that is invoked by the SVC1 call to start the device. It sets up registers to be used later by the driver and saves R2-R15 in the DCB. U4xDVR executes with external interrupts enabled.

The second section, which is the first ISR, is invoked by U4xDVR simulating an interrupt (SINT) to the device. The second section (ISR1xxx), entered with interrupts disabled, starts the device, sets the timeout count, and enables and arms the device interrupt. ISR1xxx exits through the DCB to the instruction immediately following the SINT in U4xDVR, with external interrupts again enabled. U4xDVR in turn exits to the task scheduler through EXEC routine IOTWAT, which updates the I/O wait thread. The scheduler eventually returns control to the instruction immediately following the SVC1 start call, and SOUNDER continues to execute.

The third section (ISR2xxx) is entered either as a result of a device interrupt, transmission complete, or a timeout count, DCB.TOUT, which has been decremented to Ø. ISR2xxx determines how it was entered and takes appropriate action. In either case ISR2xxx adds the address of the DCB + 1 to the top of the system I/O termination queue, LIOTRM, and the system is informed that the SVC1 device has finished. ISR2xxx exits through the DCB and the scheduler eventually returns control to the point of interruption. If the SVC1 wait call is made before ISR2xxx is entered, SOUNDER (or PIPLN) is not scheduled to run until the system sees the appropriate entry on LIOTRM. If the device has already finished when the SVC1 wait call is made, the system determines from the DCB that the driver is inactive and returns to SOUNDER directly.

The fourth section (U4xTRM) is entered as a result of command CANHF (Chapter 8), to which SOUNDER responds by executing an SVC3 call to cancel itself, or as a result of a CANCEL command to the Command Processor. In the latter case, the system executes an SVC6 call to cancel, or delete, the task. In either case, part of the task-canceling process is to terminate any ongoing I/O, and this is done at the driver level by the termination routine, U4xTRM.

The addresses of the first and fourth sections, or routines, are preset in the DCB's as a result of the SYSGEN process (Section 9.5). U4xDVR and ISR1xxx set R15 to A(ISR1xxx) and A(ISR2xxx), respectively. U4xDVR saves R2-R15 in the DCB explicitly, and ISR1xxx exits through DCB.LEAV, which saves R8-R15 in the DCB. Thus the system is informed of which driver section, or routine, is to be executed as the result of the next external event (SVC1 call, interrupt, or cancel) requiring action at the driver level.

The TSG, FEP, and EKO drivers are described below.

2.11.1. The TSG Driver [DCW]

U4ØDVR is the initialization routine for the TSG driver. U4ØDVR sets R15 to A(ISR1TSG), the routine invoked when the SINT is executed; R14 to A(DCB) from R1; R13 to A(TSG start parameter block) from R3; R12 to 37 (hex), the TSG device address from R6; and R1Ø to SVC1.SAD, the address of the TSG command byte, from R13. R1-R7 are set up on entry to U4ØDVR by EXEC. (See OS/16 MT2 Guide to Writing Drivers.) U4ØDVR saves R2-R15 in DCB.RSAV, as required by the OS, then executes a SINT to device 37, invoking ISR1TSG. After ISR1TSG executes, U4ØDVR exits to EXEC routine IOTWAT.

ISR1TSG gets the TSG command byte from the buffer pointed to by R1Ø and sends the command byte to the TSG via DIO 24. After a delay of 2-3 microseconds, depending on the Interdata cycle time, ISR1TSG starts the NMI signal to the TSG (Section 2.9.2.1) by writing 6 to DIO 25. The NMI signal must last at least 1Ø microseconds, and in effecting the necessary delay, ISR1TSG does the following: sets DCB.TOUT, the timeout count, to 3 seconds; sets R15 to A(ISR2TSG), the routine to be executed on receipt of the TSG DONE interrupt; clears R9 to Ø on the assumption that the TSG start will not time out; adds 1 to R1Ø so that EXEC routine IODONE can compute the length of the data transfer; and sets R11 to A(DCB) + 1 from R14, to be added to the top of LIOTRM by ISR2TSG (below). ISR1TSG can now end the NMI and does so by writing 2 to DIO 25. Finally, ISR1TSG outputs command 41 (hex) to device 37 and exits through DCB.LEAV. Command 41 enables and arms the device 37 interrupt and leaves the DIO interface in the read/write state.

ISR2TSG loads DCB.TOUT into R8, zeroes DCB.TOUT to terminate the timeout countdown by the clock ISR, and tests R8. If R8 is non-zero, the interrupt was TSG DONE, and control goes to IOTSAV. Otherwise, the TSG has timed out, and R8 is set to 82ØØ (hex). At IOTSAV, R9 (Ø or 82ØØ) is put into DCB.RSAV + 1Ø and becomes the exit value of R7 (device independent status) for EXEC routine IODONE. R11, A(DCB) + 1, is added to the top of LIOTRM, R15 is set to DCB.NOPI so that unsolicited interrupts will be ignored, and ISR2TSG exits through DCB.LEAV.

U40TRM disables and disarms the device 37 interrupt and leaves the DIO interface in the read/write state with command C1 (hex). R9 is set to 8400 (hex), the error code for an unrecoverable error, and control goes to IOTSAV in ISR2TSG (above).

2.11.2. The FEP Driver [DCW]

U41DVR is identical to U40DVR (Section 2.11.1), except that R12 is set to 77 (hex), the FEP/EKO device address. The SINT invokes ISR1FEP.

ISR1FEP senses the device 77 status, but has no need to examine it further. Because the SVC1 call to start the FEP only sets up the OS to wait for the FEP DATA READY interrupt, there is no actual FEP start in the driver; the FEP is started by the TSG. ISR1FEP simply sets DCB.TOUT to 3 and enables and arms the device 77 FEP DATA READY interrupt with command 58 (hex). Note that command 58 includes command bits 11 and 12 in the high (1) state. (See SNDRINIT, Section 2.6.5, and RESFEP, Section 2.6.2.) ISR1FEP exits through DCB.LEAV with R15 = A(ISR2FEP).

ISR2FEP clears R9 to 0 on the assumption that the FEP has not timed out, loads R8 with DCB.TOUT, then zeroes DCB.TOUT. If R8, the entry value of DCB.TOUT, is non-zero, ISR2FEP has been entered as a result of the FEP DATA READY interrupt; otherwise, the FEP has timed out, and R9 is set to 8200 (hex). At IOTSAV, DCB.RSAV + 10 is set to R9 for IODONE. (See ISR2TSG, Section 2.11.1.) The A(DCB) + 1 is added to LIOTRM, R15 is set to DCB.NOPI, and ISR2FEP exits through DCB.LEAV.

U41TRM sets R9 to 8400 (hex), and since ongoing I/O is not in question, control simply goes to IOTSAV in ISR2FEP (above).

2.11.3. The EKO Driver [DCW]

U43DVR is identical to U40DVR (Section 2.11.1), except that R12 is set to the FEP/EKO device address, 77 (hex), R11 is set to the driver device address, that of SELCH F1, and the SINT invokes ISR1EKO.

ISR1EKO stops the SELCH with command 8, then writes SVC1.SAD, the EKO buffer starting address, and SVC1.EAD, the EKO buffer ending address, to the SELCH. These are the addresses put in AEKOBFR by STEKO (Section 2.9.1.1). The SELCH is an I/O device that can be shared among a number of devices for data transfers. For this reason, and even though SELCH F1 is dedicated to the EKO device, ISR1EKO associates the EKO device with SELCH F1 by reading the device 77 status. Finally, ISR1EKO sets DCB.TOUT to 3, starts the SELCH with command 3Ø (hex), and exits through DCB.LEAV with R15 = A(ISR2EKO). Although the SELCH is started here, the presence of the FEP DATA READY signal, which is tied to the FEP interrupt (Section 2.11.2), actually starts the data transfer from the FEP to one of SOUNDER's EKO buffers.

ISR2EKO sets R9 to Ø or 82ØØ depending on DCB.TOUT and zeroes DCB.TOUT. At NOTOUT, ISR2EKO reads the SELCH address register into R1Ø to get the address of the last data transferred, and unless R1Ø = SVC1.SAD, increments R1Ø by 1 for IODONE's data transfer length computation. If R1Ø = SVC1.SAD, no data were transferred, and IODONE will compute the transfer to be Ø (SVC1.SAD - R1Ø). At ARM77, the FEP DATA READY signal is cleared with command 8 in case the data transfer was incomplete. (See note, below.) Also, because the EKO DONE interrupt disables the FEP interrupt (FEP and EKO share device address 77), the device 77 interrupt in re-armed and re-enabled with command 58 (ISR1FEP, Section 2.11.2). At IOTSAV, DCB.RSAV + 1Ø is set to R9 for IODONE, A(DCB) + 1 is added to LIOTRM, R15 is set to DCB.NOPI, and ISR2EKO exits through DCB.LEAV.

Note: Two constraints are placed upon the length of the EKO data transfer. First, the first 16 bits of the FEP output memeory contain the number of 1/2-words to be transferred. This count is decremented by 1 for each 1/2-word transferred, and when the count goes to Ø, the transfer is considered complete, and the EKO DONE interrupt is generated. Second, the SELCH is given the input buffer limits (ISR1EKO, above), and if during the data transfer, the buffer end address is reached, the SELCH terminates the transfer and issues the EKO DONE interrupt, even through the 1/2-word count remains positive. However, the FEP DATA READY signal is automatically cleared only when the 1/2-word count goes to Ø, and the FEP DATA READY signal must be cleared before the TSG will issue a new start to the FEP. Therefore, in case the EKO data transfer is incomplete from the FEP's standpoint, the FEP DATA READY signal must be explicitly cleared after the EKO DONE interrupt is sensed so that the data pipeline can function.

U43TRM sets R9 to 84ØØ for IOTERM's R7, stops the SELCH with command 8, and exits through IOTSAV (ISR2EKO, above).

3. PICKER

James R. Winkelman

3.1. Introduction

Chapter 3 discusses various configurations of PICKER revision 2.Ø1. All configurations of PICKER 2.Ø1 log in dated Ø8/Ø4/83. For details about changes made to PICKER 2.ØØ, refer to Appendix A, dated 1983 November.

Task PICKER selects range-coincident echoes from the data supplied by SOUNDER and sends the selected data on to MANAGER. It acknowledges the receipt of sounding start and stop Q-blocks from SOUNDER and start-up from COUNSEL, which also generates a logging message.

PICKER uses two definitions from SNDR in PCB.CAL. SNDR is a block of definitions for SOUNDER parameters created at assembly time. These definitions are edited into PCB.CAL and entered into PICKER with the COPY SNDR command during assembly (see SNDR, Section 7.7).

EKOSIZE = 1Ø4Ø is the length of the EKO buffer in bytes.

Q = 4 specifies the use of SVC4 (rather than SVC6) and SVC8.

Several more definitions are available to match installation needs. An installation may require several configurations of PICKER for various studies. Four such configurations are described below. Assembly parameters for the PICKER task are defined from PIKØ21ØØ to PIKØ38ØØ.

SV6 = Q Q is defined in SNDR. If Q is 6, SVC 6,- and SVC 9,- are executed normally. If Q is 4, SVC 4,- is used to queue another task. SVC 4,- is much faster than SVC 6,- for queueing tasks.

SV9 = Q/2+6 If Q is 4, the SVC 9,- is replaced by a much faster SVC 8,-. SVC SV9,- is used to exit from tasks to wait for another input.

PICKFIV = Ø PICKFIV is used to specify two different PICKER modes. The version described here is used at NOAA to alternate between a full set of options, and small set which uses large buffers. The two options require the same size partition. The user is encouraged to select a set better tailored to their installation.

PIK.TEST = Ø PIK.TEST is non-zero to include additional tests and buffer usage counts. MLOST is incremented each time that the previous PØ2 buffer is still in use when the next buffer is packed. MLOST is related to the amount of time available to lower priority tasks. NLOST is incremented when both buffers are still in use and PICKER must wait for GRAPHER or MANAGER to release one of the buffers (catch up). If necessary, both PICKER and SOUNDER can wait until all tasks catch up. SOUNDER will log the 'LOST PULSES' message if the wait results in unscheduled TSG NOP fills.

PIK.MEAN = 1 A mean and maximum amplitude are calculated by the FEP for each channel. These mean, max pairs are saved and output at range -1 if PIK.MEAN is non-zero. The mean was used to calculate the threshold, and both give an indication of signal to noise.

The following five flags support programs in the FEP. Substantial amounts of memory can be saved by deleting support for some modes. Unsupported FEP modes are ignored.

PIK.STD = 1-PICKFIV A non-zero value supports the original FEP and PICKER processing, in which PICKER did a lot of computing. It also requires a lot of memory.

PIK.EQR = 1-PICKFIV If non-zero, the equal reference processing is supported. PIK.FEP does much the same process in the FEP, which saves both time and memory.

PIK.FEP = 1 If non-zero, the range coincidence table which was generated in the FEP is used. The FEP program does complete EQUALREF type processing.

PIK.PAR = 1 If non-zero, the partial reflection mode is supported. This mode is also useful in looking at raw data or weak echoes. The mode does not require much memory, but is normally used with very large PØ2 buffers which do require memory.

PIK.PCT = 8 If non-zero, SØ2 Q-blocks for mode PIK.PCT or greater are passed directly to MANAGER.

3.1. Introduction

A number of general parameters are required to define sizes and to provide fine control of some processes.

EQRM = 5Ø Ranges within EQRM microseconds of each other are considered coincident in PIK EQR processing.

EQRK = 6Ø The data block saved by the FEP EQR process is 6Ø microseconds wide, 3Ø microseconds before and after the peak (seven double pairs).

NPTYPE = 4 There are normally four pulses per pulse set. The code has been checked out, however, for five and six pulses per pulse set. (See PICK5PPS and PICK6PPS, below.) The maximum value for NPTYPE is 8, unless the code is modified.

MAX.EPF = PICKFIV*36+16 The program can pack up to 16 echo ranges into the output buffer, 52 for partial reflection versions, although SOUNDER starts with only eight. PICKFIV = Ø or 1 sets MAX.EPF to 16 or 52. The SOUNDER command EPF n can be used to change the value of EPF used, although PICKER will not save more ranges than MAX.EPF. For example, the 'mean, max', calibration, and six echoes would fill a buffer with EPF = 8. EPF stands for echoes per frequency.

MAX.HPE = NPTYPE*4+1 = 17 MAX.HPE is used to check for valid HPE n values. If n is too large, MAX.HPE is used, but no error message is generated. HPE stands for half words per echo and can be changed by the SOUNDER command HPE. (See Section 2.7.7 for details about using HPE for complementary coded soundings.)

MAX.OPOB = 42 This value is used to limit the number of bytes in the PCT header. If the value of MD.OPOB as read from the SCT is greater than MAX.OPOB, MD.OPOB is set to MAX.OPOB. This is the case for complementary coded soundings for which SOUNDER gives 8 starts to the TSG per pulse set, but only 4 of which produce data.

NECHO = 28 PICKER keeps a table of peaks and their ranges, for testing coincidence. There is room in that table for NECHO ranges. The table can be expanded to insert new ranges until it is full. Ranges which do not match a range in a full table are ignored.

MICRO = 6Ø Ranges within MICRO microseconds of each other are considered coincident in STD processing.

BASE = 14 BASE is the length of a FEP buffer header.

STRT = 6 STRT is twice the number of samples retained before threshold is exceeded in a STD process FEP buffer.

ENDB = 6 ENDB is twice the number of samples retained after the signal drops below threshold in a STD process. The current STD, SNDR, FEP process inserts three X,Y pairs before and after the data above threshold.

The 64K bytes of memory standardly available in the Interdata computers do not always allow for all of PICKER to be in memory at one time. If all types of processing are selected for a given PICKER configuration, echoes/frequency and pulses/pulse set (1/2-words/echo) must be limited. If a minimum of processing types is tolerable, echoes/frequencies and 1/2-words/echo can be increased. The decisions about these trade offs are made at assembly time from the parameters described above, and many combinations are possible. Currently, four configurations of revision 2.Ø1 of PICKER have been tested at SEL.

1. PICKER.TSK (source, PICKER.CAL) is the standard configuration from which all others derive. PICKER.TSK is the default for I-, B-, K-, V-, and Z-modes and supports the four standard types of FEP data processing: SNDR or FEPORIG, EQR or EQUALREF, PAR or PARTIAL, and PIK or FEP PICKER. (See Sections 3.1Ø through 3.13.) All processing is limited to 4 pulses/set (8 for some complementary coding schemes), 17 1/2-words/echo, and 16 echoes/frequency. (For PARTIAL processing, used by G-, P-, and W-modes, echoes/frequency translates to X,Y pairs/pulse.)

2. PICK4PPS.TSK (source, PICK4PPS.CAL) supports FEP PICKER processing at 4 pulses/set and PARTIAL processing at 1 to 4 pulses/set, all at up to 52 echoes/frequency. PICK4PPS does not support SNDR or EQR processing; if SNDR or EQR processing is invoked, PICK4PPS passes no data on to MANAGER. PICK4PPS is derived from PICKER, above, by setting PICKFIV = 1 and leaving NPTYPE = 4.

3. PICK5PPS.TSK (source, PICK5PPS.CAL) supports FEP PICKER processing at 4 pulses/set and PARTIAL processing at 1 to 5 pulses/set and up to 52 echoes/frequency. PICK5PPS does not support SNDR or EQR processing. PICK5PPS, the default for G-

and P-modes, is derived from PICKER by setting PICKFIV = 1 and NPTYPE = 5.

4. PICK6PPS.TSK (source, PICK6PPS.CAL) supports only PARTIAL processing at 1-6 pulses/set and 52 echoes/frequency. PICKFIV = 1, PIK.FEP = Ø, and NPTYPE = 6. PICK6PPS.TSK is the default for W-class soundings.

3.2. Input

PICKER keeps ranges with at least M echoes in them, where M is in location MD.M (Section 7.1) and may be changed with command KEEP (Chapter 8). KEEP is ignored during PARTIAL processing.

The CØ2 Q-block (from command RANGE) from COUNSEL defines range limits on echo locating which can be used to speed processing (Section 3.6).

Q-blocks SØ4 and SØ6 from SOUNDER start and stop a sounding and are passed on to MANAGER (Section 4.2).

The SØ2 Q-block (Section 3.9) contains the address of the Pulse Configuration Table (PCT, Section 7.2) and the EKO buffer.

3.3. Output

PICKER answers COUNSEL and sends a logging message during start-up.

SØ4 and SØ6 Q-blocks are sent to MANAGER; the SØ4 immediately upon receipt, and the SØ6 after the last data block has been sent.

MD.OPOB in the SCT is set to MAX.OPOB under certain conditions. (See Section 3.7 for details.)

MD.ECHO in the SCT is set, and MD.EKOF and MD.EKOS are set, if specified by command ECHO 1. (See Sections 3.6 through 3.8.)

PICKER sends a PØ2 Q-block to MANAGER giving the location of a buffer containing:

Length in bytes, 1 word.
4 PCT blocks, 5 words each.
MD.EPF (typically 8) ECHO blocks, 17 words each.

3.3. Output

The ECHO blocks are a range in microseconds and 8 X,Y pairs.

Data Record

A typical Data Record takes up 42 + 34*n bytes, or 21 + 17*n words, where n is the number of ECHO BLOCKS in the record; n may be Ø. The fixed 21 words at the start of each block contain the record length and four PCT's.

A Data Record contains the following words in default sequence (see commands ATN and DFAC, Chapter 8):

WORD	DESCRIPTION
1	Byte count for the Record.
2-6	PCT, 1st antenna pair, frequency F.
7-11	PCT, 2nd antenna pair, frequency F + delta F.
12-16	PCT, 1st antenna pair, frequency F + delta F.
17-21	PCT, 2nd antenna pair, frequency F.
22	First word of ECHO BLOCK1, 17 words: R-BAR = average range value (echo time) in microseconds of all echoes in BLOCK1.
23	X-amplitude of first echo.
24	Y-amplitude of first echo.
25-26	X- and Y-amplitudes of second echo.
. . .	. . .
37-38	X- and Y-amplitudes of eighth echo.
39	ECHO BLOCK2, R-BAR.
4Ø-55	ECHO BLOCK2, 8 X,Y amplitude pairs.
. . .	. . .
9Ø-1Ø6	ECHO BLOCK5.
. . .	. . .
22+17*(n-1)	ECHO BLOCKn, first word.
. . .	. . .
22+17*n-1	ECHO BLOCKn, last word.

3.4. PACK

PIK51660 to PIK58700. This subroutine packs into a P02 buffer the PCT's and data in TABLE that satisfy the M-out-of-N criterion.

Input MD.M = 8 or set by command KEEP.

TABLE, used by PACK to form a P02 Q-block (Section 3.3). TABLE has NECHO (a PICKER parameter, currently 28) blocks of 18 words each. These descriptions are byte addresses. Missing data are zero.

TABLE + 0 contains the number of ranges included in this average range.

TABLE + 2 contains the average range in microseconds corrected for antenna, receiver, and TSG delays (MD.ADLY, PD.TDLY).

TABLE + 4 + 4*n contains the nth X value.

TABLE + 6 + 4*n contains the nth Y value.

Output P02 Q-block.

Registers:
0: 02 'P'
2: status = '-1000'
4: PICKRn, n = 1 or 2

The total count of pairs is accumulated during packing into MD.EKOF. The count of pairs actually packed into P02 buffers is accumulated in MD.EKOS.

3.5. PKEXIT and QUEUE

All routines exit through QUEUE to wait for the next Q-block. PKEXIT clears the Q-block status before going to QUEUE.

PIK06940 to PIK07800. The Q-block is checked and a branch is made to the appropriate process.

3.6. Q.C

PIK10400 to PIK14200. Q.C processes Q-block C02 Q-blocks from COUNSEL to PICKER.

Input — RANGE [n], MEAN [n], or ECHO [n] typed by the operator (see below).

RANGE [n] — If n is zero or missing, no range testing is done. Otherwise, the ranges set up for frequency n in SCT.FTBL (T-class frequency table) (see command RNGE, Chapter 8) are used to speed processing by ignoring echoes outside the assigned range. Range intervals should be somewhat wider than actually required to prevent loss of echoes near the limits. Normal time jitter may put some of the echo returns just outside the range so that fewer returns are chosen for coincidence.

MEAN [n] — If n is zero or missing, MEAN and PEAK (MAX) are not included in the P02 data block. If n is not zero, MEAN and MAX for each pulse and channel are inserted at a range of minus one. MEAN is used to set the signal threshold for processing. If MEAN is large, there was probably an interfering signal. The ratio of MAX to MEAN is an indication of signal to noise.

ECHO [n] — An n of zero or missing, cancels the ECHO mode (default). If n is not zero the message: mmmmm ECHOS OUT OF nnnnn is logged after each sounding. MD.EKOS is the value of mmmmm, and is the total number of X,Y pairs saved in the P02 data block. MD.EKOF is the value of nnnnn, and is the total number of X,Y pairs from TABLE considered in packing the data block. EKOS plus the number of pairs which did not meet the KEEP criterion, is EKOF. Both values are modulo 65536. This ratio is larger with the FEP PICKER because all entries into TABLE have already met a KEEP of four, that is, data from two pulses were above threshold. The mean,max data, if present, are not counted, while the calibration data, if present, are counted. If the KEEP is 8 of 8, MD.EKOS should be a multiple of 8. If the KEEP is 6 of 8, MD.EKOS will be 6 * N1 + 7 * N2 + 8 * N3, where N1 is the number of echoes saved that had exactly sixfold coincidence, N2 is the number of sevenfold coincidence, etc. Once MD.EPF echoes have been saved for the pulse set, no more echoes will be counted in MD.EKOS or in MD.EKOF for that pulse set. The accumulation to both variables occur during the same packing scan in PICKER. Range gating may also decrease MD.EKOF and MD.EKOS.

3.7. S.Ø4

An S.Ø4 Q-block signals the start of a sounding. The Q-block is passed to MANAGER, and the program expects data within seconds.

PIKØ54ØØ to PIKØ692Ø. The program isolates the S- vs. T- or W-class control bit for later use in testing ranges.

MD.OPOB, MD.EPF, MD.PPS, and MD.HPE are tested. Legal values for each are saved for use during the sounding. Also, if MD.OPOB > MAX.OPOB, MD.OPOB in the SCT is set to MAX.OPOB on the assumption that a complementary coded sounding is in progress, or that TSG NOPs are embedded in the pulse sets. PCTs are discarded if the word count in the EKO buffers is less than BASE, currently 14.

3.8. S.Ø6

An S.Ø6 Q-block signals the end of a sounding. The process finishes everything connected with this sounding.

PIKØ85ØØ to PIK1Ø3ØØ. The last of the data in TABLE is packed and sent to MANAGER.

The echoes found, echoes saved and their ratio is stored in the SCT. MD.EKOS*256/MD.EKOF is stored in MD.ECHO at the end of a sounding and an echo count message of MD.EKOS and MD.EKOF is logged if an ECHO 1 was given.

MD.ECHO is an indication of the noise rejection by PICKER. If MD.EKOF overflows, MD.ECHO is zero until MD.EKOS overflows, after which MD.ECHO can be almost anything. Overflows occur when 65536 pairs are counted.

MD.ECHO is not required for processing, so its questionable validity at very large counts is not considered serious.

The S.Ø6 Q-block is sent to MANAGER after PICKER has finished.

3.9. S.Ø2

The S.Ø2 Q-block passes the EKO buffer data from SOUNDER to PICKER. The buffer format is almost the same for all data types. In some types, the second channel data is included with the first channel data, so that there is no address R2, channel 2 data. The buffer types are given by words 14 and 15, Ro and Po. The branch destinations and buffer type (Ro, Po) are SNDR (Ø,-), PAR (-,Ø), EQR (-,-), or FEP (Ø,Ø).

PIK144ØØ to PIK17ØØØ. The PCT data block is saved, buffer locations decoded, the range gate information retreived (see RANGE), and a four way branch taken. Short buffers and data processing types which are not supported by the assembly parameters are ignored.

SØ2 Queue block:

```
Ø  'Ø2', 'S' (bytes Ø and 1)
2  Status, '-1ØØØ' in, 'ØØØØ' out.
4  Address of EKO buffer (BUFFER).
6  Address of PCT (Section 7.2).
8  Address of SCT.FTBL entry (Section 7.4).
```

BUFFER contains data from channels 1 and 2 in the format given below. The first two blocks contain derived data generated by the FEP for each channel. Xo and Yo are zero-offsets related to calibration; they are subtracted from the data. The mean is at present unused. The peak enters into the maximum peak computed by SOUNDER for comparison with PKHI and PKLO (Chapter 8) to accomplish automatic gain control. R is the range for each set of data; Ro corresponds to the calibration pulse. P points to the next R.

```
Ø  length
1  start of channel 2 data, address B2

2  Xo   _|
3  Yo    |   channel 1 derived data
4  Mean  |
5  Peak_ |

6  Xo   _|
7  Yo    |   channel 2 derived data
8  Mean  |
9  Peak_ |
```

```
   1Ø   Max1 ¯|
   11   Max2  |   see Chapter 13, FEP programs
   12   Cala  |
   13   Date_|

   14   Ro   ¯|
   15   Po    |
   16   X     |   channel 1 data
   17   Y     |
    -   ---  _|

 B2     Ro   ¯|
 B2+1   Po    |
 B2+2   X     |   channel 2 data
 B2+3   Y     |
    -   ---  _|
```

3.1Ø. SNDR or FEPORIG

SNDR is a modification of the original process done by PICKER in the HF sounder. SNDR is recognized by a zero word 14, Ro, and a non-zero word 15, Po.

PIK18Ø2Ø to PIK2Ø9ØØ. Two register sets (all computer registers are saved and restored), SCAN1 and SCAN2, are constructed corresponding to the data from the two channels. Each channel is scanned for a peak worth saving. These two ranges, one from each channel, are compared for coincidence and the data is stored. A range which was too large for coincidence is not scanned again until its data has been saved in TABLE. If the range from one channel is much larger than the range from the other channel, it remains unchanged until the smaller channel catches up.

3.1Ø.1. PKLOOP

PIK211ØØ to PIK317ØØ. PKLOOP finds a range in the TABLE defined in PICKIT.

Input	EKO buffer from SOUNDER (BUFFER). Registers 1-3 from SCAN1 and SCAN2.
Output	TABLE as defined in Section 3.3.
Registers	PKLOOP makes the following internal use of registers:

3.10.1. PKLOOP

R0 always zero.
R1 0: SCAN1 range is smaller.
   1: SCAN2 range is smaller.
   2: both ranges are within MICRO (currently 60) microseconds of each other.
R2 range from SCAN2.
R3 range from SCAN1, then count of items averaged into new range.
R5 range in TABLE.
R6 current table location.
RA-RF multiple register move.

3.10.2. SCAN

Input SCANn table, n = 1 or 2.

EKO buffer from SOUNDER (BUFFER).

Registers: all 16 registers are loaded from SCANn before SCAN is called. Details about register use appear below.

SCAN1 defines the processing of the data from channel 1; SCAN2 does the same for channel 2.

Output SCANn table updated.

All registers except R0 are saved after exit from SCAN. Many of these registers are set by PICKIT to define the two buffers.

Method PIK31900 to PIK41940.

A single pass is made through the buffer data. PEAKR is called for any peak established and then a return is made to PKLOOP.

During set-up for processing a new echo, if the echo does not overlap the range specified by the RANGE command, the echo is not processed. A second check is made within PEAKR so that SCAN will not return a value outside the requested range.

For T- and W-class soundings the range is always tested against the frequency table given in S02 + 8. For an S-class sounding, the range will be tested only if the operator has given a RANGE command. The same range limits will be used for all frequencies.

SCAN finds a local peak using amplitudes from JABS. The amplitude is calculated as the sum of the absolute values of X and Y. This amplitude estimate is fully adequate to locate peaks because the peak is expected to have constant phase across it.

If the second point after the local peak is higher, the search continues for another local peak. If the second point is not higher, the local peak index is saved and SCAN looks for a second local peak. If SCAN moves four points past the first peak without detecting a second peak, the first peak is established. If a second peak is found, the higher of the two peaks is established. Peaks with less than three points between them (40 microseconds) are not output as two separate peaks.

Registers The following describes register usage in detail. R0 through R2 are used as temporary registers within SCAN.

- R0 Defined only on input as the start of the amplitude buffer. It is not saved on exit (which preserves the input value).
- R1 X-BAR out, changed only if a new range is computed. X-BAR is the average of the peak X and the next X.
- R2 Y-BAR out, changed only if a new range is computed. Y-BAR is the average of the peak Y and the next Y.
- R3 The calculated range, which is the essential output of this subroutine. SCAN interprets the input values of R3 as follows. R3 = -1 for the first call to SCAN within a buffer (to do initialization). R3 = 0 if the previous range was used and a new range is to be calculated. R3 > 0 if no new range is needed, the range has not been used, (immediate exit). R3 is calculated by a five-point, least squares quadratic through the points surrounding the peak amplitude (see PEAKR method, Section 3.10.3). If there are no more echoes, the range is set to 32000 microseconds, recognized as too large to be real.
- R4 Xo, the zero-offset in X from BUFFER (Section 3.9).
- R5 Yo, the zero-offset in Y from BUFFER (Section 3.9).
- R6 Current amplitude for comparison with the new amplitude.

3.1Ø.2. SCAN

R7 Base range for the echo. The range within the echo is added to this range.

R8 Subroutine link.

```
     LM   RØ,SCANn
     BAL  R8,SCAN
     STM  R1,SCANn + 2
```

R9 The end of the buffer, meaning that there are no more echoes past this index.

RA The index of the peak, offset by PX (internal variable, -4 bytes).

RB Index for sum of absolute values of X and Y, an approximate amplitude. These amplitudes overlap the data in the buffer. They start at the value given in RØ for each echo.

RC Index of the last X,Y pair in the echo set. It is used to test for done. It is offset from the true end of echo by 2*ENDB (12 bytes).

RD This is the index for a suspected or verified peak. It is offset from the true peak by PXY (internal variable, currently -8).

RE Index of the current X,Y pair being examined.

RF The location of BUFFER from SOUNDER. This is the base address for all pointers.

3.1Ø.3. PEAKR

PEAKR interpolates for the range of an established peak.

Output R1: X-BAR.
R2: Y-BAR.
R3: range in microseconds.

Method A through E are five successive amplitudes with C the peak value.

R1 = [X(peak) + X(next)]/2 - Xo

R2 = [Y(peak) + Y(next)]/2 - Yo

new range = 7*[(A - D/2) + (B/2 - E)]/
[(A - D/2) - C - (B/2 - E)]

R3(range) = 5*[RANGE + new range] - MD.ADLY + PD.TDLY

Where:

RANGE = R7(base range) + R1(peak index) -
RØ(defined at input)

If RANGE exceeds MAXIMUM, then R = 32ØØØ. If RANGE is less than MINIMUM, the last pair is tested to search for a new range. MAXIMUM and MINIMUM are defined by the command RANGE described in Section 3.6 under Q.C, or by the SØ2 Q-block during T- or W-class soundings.

3.11. PAR or PARTIAL

PIK515Ø8 to PIK51652.

PAR processing is selected by a non-zero Ro and a zero Po. The pointer is not needed because all echoes contain six words: range, zero, X,Y (channel 1), X,Y (channel 2). PAR is used to look at several consecutive ranges, without requiring the echoes to exceed threshold.

The program assumes that all buffers within a pulse set contain the same range block. Data are stored in successive echo blocks without checking for a range match.

The number of ranges is often large enough that the use of an intermediate TABLE would use too much memory. Data are packed directly to the next PØ2 buffer. PAR is the only process to recognize and properly process 2Ø microsecond sampling, as opposed to the usual 1Ø microsecond sampling.

3.12. EQR or EQUALREF

PIK17592 to PIK18ØØØ.

EQR processing is selected by having both Ro and Po, non-zero. The data are in echo blocks of thirty words: range, pointer, and seven sets of double pairs, X,Y (channel 1) and X,Y (channel 2). The range is that of the first double pair. The middle (fourth) double pair is the peak. This peak is the peak of the sum of amplitudes in the two channels.

The EQR process uses two EKO buffers (BUFFER) at a time. The location of the first EKO buffer is saved and the routine goes to QUEUE to wait for the next pulse buffer (PIK1752Ø to PIK1754Ø).

The ranges from the two consecutive pulses are compared for a match within fifty microseconds. The two double pairs closest to the average range are saved. If these two pulses were at the same frequency or frequency plus delta F, the data should all have the same delays and phase rotation.

The last two pulses are processed the same as the first two pulses. If there is a range match within fifty microseconds, all four pulses are combined. If TABLE is not yet full and there is no range match, TABLE is expanded to make room for this new range.

The EQR output from the FEP is always in pairs of channels, coincidence of two. Ranges in TABLE require a match of two pulses, a coincidence of four. This results in TABLE containing only coincidences levels of four or eight.

The coding is all straight forward, except for the address truncation to address the proper range pairs corresponding to a range nearest the average range of the two pulses. Both buffers (Q-blocks) are released at the end of processing of the buffer pair.

3.13. FEP PICKER

PIK17248 to PIK17480.

The FEP PICKER is EQR type processing where all coincidence checking is done in the Front End Processor (FEP). The FEP does coincidence checking on all four pulses at once, without the requirement on EQR processing to return buffers to SOUNDER, so that SOUNDER can continue sounding.

The first three buffers from the FEP are short, but SOUNDER needs the MAX values for gain control. PICKER needs the PCT, which accompanies those short buffers, to tell the user about those pulses. Short buffers are ignored, after the PCT is saved, at PIK17221. During complementary pulse coding, the first pulse of a complementary pair must be completely ignored during processing. PIK17180 to PIK17215 check for very short buffers and remove the PCT which corresponds to this 'forgettable' buffer.

The FEP program requires that the range appear in at least two pulses (KEEP 4). In contrast to EQR processing, a range can contain three pulses (KEEP 6) or all four pulses (KEEP 8). All ranges saved in a set have peaks which are within fifty microseconds of each other. The data from all four pulses are at the same range.

FEP PICKER processing is selected by two zeros in Ro and Po. These zeros are followed by a data block, starting in word 16, in the same format as TABLE. This block is used by the PACK subroutine in combination with the current KEEP. The ranges in the data buffer have been converted to microseconds and adjusted for analog and TSG delays. A range gate (see RANGE and RNGE, Chapter 8) may require the data block to be contracted to remove ranges smaller than desired. The end of buffer pointer is changed to remove ranges larger than desired.

3.14. Range Limitations

The number of ranges, from any one pulse set, which are written to tape depends on several factors. These are generally controlled by the amount of memory the program and user have agreed to use.

The buffer which is transferred to SOUNDER from the FEP is limited to 52Ø words by mutual agreement of the authors. This buffer contains a maximum of about ten distinct ranges in the SNDR mode. Ten ranges are enough under most circumstances. The EQR buffers contain fifteen ranges in a fixed format. They are under 512 words long to allow the FEP PICKER program to keep two of them in the same memory. PARTIAL buffers could hold 81 ranges in the 52Ø words, but PICKER doesn't have room for the P.Ø2 buffer.

The FEP PICKER program compares the ranges in all four pulse buffers at the same time and saves the first 28 ranges which meet a KEEP of four. These 28 ranges, by no coincidence, just fill a 52Ø word buffer. The first two ranges are the mean,max and the calibration. Ranges greater than the 28th range are lost, even if they have more pulses coincident than an earlier saved range.

The EQR buffers are processed in pairs by PICKER and must have range coincidence between the pair of buffers to save the range. The EQR buffers contain fifteen ranges including the calibration, so no more than fifteen new ranges can be stored into TABLE during the processing of a buffer pair. The assembly variable NECHO, currently 28, is the maximum number of ranges in TABLE. Since the calibration pulses are coincident, only one range match in the remaining fourteen pairs will cause no ranges to be ignored.

The difference between EQR and FEP PICKER processing is that EQR compares the buffers in pairs, then matches the ranges found in the first pair of buffers with the ranges found in the second pair. A range must appear in both of the buffers in a pair to be saved. The FEP PICKER compares the ranges in all four buffers at once, and saves the range if it appears in any two of the buffers.

The SNDR processing treats each channel in each pulse independently. The ten ranges in each channel of the first pulse could use twenty of the 28 ranges in TABLE. During processing of the second pulse, new ranges can be inserted into TABLE only until it is full. New ranges, after TABLE is full, are lost. Ranges which match a range already in TABLE are included with that range, so all pulses which meet an eight out of eight will not be lost. Very few pulses metting a six out of eight KEEP are lost, except under extreme circumstances.

The PACK subroutine gets the 28 range coincidence table, either TABLE or the FEP PICKER buffer. PACK saves the mean,max values and the calibration values as its first two ranges. The remaining ranges in TABLE are scanned and the first ones to meet the KEEP criterion are saved. The EPF command, in combination with MAX.EPF (echoes per frequency, ranges per frequency in this context) define the maximum number of ranges in the output buffer, P.Ø2.

EPF starts as eight, but can be set as high as sixteen in the space currently allocated to PICKER. This means that only the first fourteen ranges plus mean,max and calibration can be saved, out of the 28 intermediate ranges in TABLE. Two counts of pairs are kept during this process. One is the count of all pairs considered, and the other, those pairs saved. For example, four ranges are in TABLE with pair counts of eight, three, five, and six. For a KEEP of six, fourteen pairs out of 22 were saved. These counts do not include any pairs after EPF ranges have been packed.

4. MANAGER

James R. Winkelman

4.1. Introduction

This chapter describes MANAGER revision 2.Ø1, which logs in dated Ø8/19/83. For details about changes made to MANAGER 2.ØØ, refer to Appendix A, dated 1983 November.

MANAGER tranfers data from PICKER to the disc and, at the end of a sounding, from the disc to magnetic tape. MANAGER supplies GRAPHER and ANALYSER with the location of PICKER output. The data stored on disc are available to stand-alone programs, GRAPHER, ANALYSER, PICKER, and CALL tasks. Previous soundings can be replotted or plotted in different formats.

4.2. Input

Q-blocks SØ4 and SØ6 occur at the start and end of a sounding and are passed on to ANALYSER and GRAPHER by PASS.ON. Q-blocks AØ2, AØ4, GØ2, GØ4, PØ6, and PØ4 get data for previous soundings from disc.

CØ2 Q-blocks from COUNSEL are initiated by the commands TAPE [n], FORMAT [n], SORT [n], CLOSE, DROP [m], SAVE [m], COPY [n], TNEW, NOQUEUE [n], NTAPE, FFILE, and BFILE. These commands are described in Section 4.7.

The PØ2 buffer contains the actual sounding data.

4.3. Tape and Disc Output

MANAGER writes a sounding on disc and the same sounding onto tape if TAPE was requested.

All tapes are 9 track, 16ØØ bpi, phase encoded. There are no ANSI label records. The tape will usually have tape marks (end-of-files) interspersed with data. The logical end of tape (EOM) is a double tape mark. The 'normal' mode from Product One is now called 'expanded'. Under some versions of MANAGER, as determined by an assembly parameter, the 'expanded' format is not available. This format is wasteful of tape and may not be worth the longer MANAGER.

4.3. Tape and Disc Output

The expanded version writes every logical record as a physical record, whose length is that of the longest record of the type. All PØ2 records are written as if all eight, or EPF, echoes are present. The first word contains the length of the good data.

All physical records on the tape were originally written as multiples of 256 bytes, or 128 16-bit words. If the tapes have been copied on other computers, a few extra bytes may have been appended during copying on computers with word lengths that are not a divisor of 256 bytes. The records are of variable length, with from 1 to 16 blocks of 256 bytes in each physical record. The 16 block maximum length is an assembly parameter in MANAGER and is considered to be a reasonable compromise between packing efficiency on the tape, loss of data in long records with parity errors, and memory requirements while originally writing the tape.

There are two types of data sets on the tape, local data and sounding data. A data set starts at the beginning of a physical record and can be identified by the top four bits, including the sign bit set. The local data set starts with a hex FFØØ and is 15 blocks in length. The sounding data set starts with a hex 82ØØ and has a SCT (Sounding Configuration Table) that is 512 bytes long, exactly two blocks. Masking off the four most significant bits gives a hex FØØ, 3840 bytes for the local data and a hex 2ØØ, 512 bytes for the sounding data. For the local data, this is the physical record length, as well as the logical record length. For the sounding data, the length is that of the logical record (SCT), with more data following in the physical record. One of these two types of record with the first bit set should follow the beginning of tape (BOT) and each tape mark (EOF), except the penultimate EOF.

The sounding data set consists of the data taken during one run. A run usually takes from ten seconds to two minutes, although it may be shorter or much longer. The sounding data set is composed of an SCT, followed by a number of pulse data sets (PØ2 blocks), House-Keeping/DIO data (HKDIO), the pulse palette extension (DIO.PTP), and a terminator. These are packed together into one or more physical records. A pulse data set typically consists of data from four pulses, the four antennas at f and the four antennas at f plus delta f; together with the pulse specific settings for each pulse.

The SCT fills the first two blocks of the first record of the set. After the SCT, the rest of the logical records in the sounding are packed together, with the first word of each logical record being the length of that logical record in bytes. The word immediately following the last word of the logical record is then normally the first word (length) of the next logical record, and so on. Assembly of several logical records continues across 256-byte blocks and physical records. This format reflects the organization of the data as it is saved on the disc during sounding. The SCT is saved in the first two sectors of this sounding, then rewritten at the end of the sounding. Data blocks are packed into blocks 256 bytes long, which become disc sectors. These sectors are grouped into physical records, beginning with the SCT, with no changes to the disc blocks during this copying process. Each of these data blocks, after the SCT, consists of a two byte pointer and 254 bytes of data. The two byte pointer points to the start of the next logical record. It is actually the number of bytes remaining in the previous logical record. A logical record with a length of zero is the data terminator. Between the last pulse data block and the terminator are two informational records. The HKDIO record starts with a hex length Axxx, the DIO.PTP record starts with a length of 9xxx, where xxx is the actual length of each record.

The two words which are crucial to proper unpacking of the data blocks, are the pointer and length. The length is correct as read when it is positive, and can have a range of 4 through 32766 bytes. When the length is negative, the top four bits identify the block type, and the lower 12 bits give the actual length in the range of 4 to 4Ø94 bytes. The pointer is not to be copied into the current logical record, nor is it to be counted in the word count of the logical record. This pointer is at the start of every 256-byte block, including at the beginning of every tape record except the first block of each data set. It is used to continue the unpacking of data after a tape error.

An assembly parameter in MANAGER will write consecutive numbers Ø,1,2,3,... as the first word of each block. These are easier to check for continuity, but can't be used to restart after a bad record.

4.4. Method

MANAGER writes data to disc according to the data's frequency index (SORT), or in the order they were taken. At the end of a sounding, the data are copied from the disc to tape, skipping missing data (zero sectors). The two sector SCT is written at the beginning of the file and sounding. This SCT is overwritten at the end of the sounding.

The disc on the 7/16 is made by CDC and uses a Xebec controller. The 8/16 has a Wang disc and Interdata controller. The removable discs are physically interchangeable, but are not formated the same by the controllers. This formatting puts some synchronization bit streams, and the track and sector addresses on the disc. The 7/16 discs can be formatted with the PREAMBLE command in DUTIL. The 8/16 disc is formatted using the disc diagnostic format program. Only a new disc should require formatting. The disc should have been running for twenty minutes to get completely temperature stabilized.

Both types of disc use the same logical organization, and the same calls. The drivers are different, see SYSGEN. A disc sector holds 256 bytes. All 256 bytes are modified by a write, but less than 256 bytes can be read. Twenty-four sectors make a track, and the track on the other side of the disc can be read or written without moving the head. These forty-eight sectors have consecutive addresses. The disc spins at 40 revolutions per second, 25 milliseconds per revolution or 24 sectors. The 8/16 can read a number of sectors in the same read and revolution. The 7/16 reads one sector at a time, and there is not enough time between sectors to restart the disc when under the operating system (the formatter can do it, but works very hard to do so). The 7/16 disc driver reads or writes every other sector of the disc request, and then goes back to read or write the missed sectors. If all sectors are on the same track, this can be done in about 50 milliseconds.

MANAGER writes sectors one at a time, packing the new data into the same buffer that was written previously. Normal eight-echo buffers have 314 bytes, and require five sectors for four buffers. One buffer per revolution would require 125 milliseconds for four buffers, less than 32 milliseconds per buffer, and well within the needs for standard sounding. MANAGER actually writes four sectors per revolution, and does not overwork the disc for normal soundings.

There are special modes, such as partial reflection modes with fifty echoes, which require seven sectors per buffer. These modes cannot be run normally with 40 milliseconds per buffer, written in 43 milliseconds or more. The SORT command has been modified to write the seven sectors in one block. These sectors can be written in seven milliseconds plus a partial revolution on the 8/16, but require fifty milliseconds on a 7/16.

4.5. Reads and Writes

Reads and writes are done by storing the six-word command in DISK.WR and executing it.

Registers — The registers contain the following information:

RA Command, function in first byte, logical unit (LU) in second byte. LU = 2 for disc, 3 for tape, and 4 for a save file on disc. The commands are:

3C02	write disc starting at given random sector
3C04	write save file at given random sector
5C02	read disc starting at a given random sector
5C04	read save file at given random sector
3803	write tape
8803	write end of file (tape)
8203	backspace file
8403	forward space one file
9003	forward space one record
0803	wait for tape done

RB Status on output (unused on input).

RC First byte address in the buffer.

RD Last byte address in the buffer.

RE Zero as upper half word of a random address for random sector calls.

RF Random sector address.

4.6. QUEUE

The QUEUE processing runs from MAN14700 to MAN19300. The program exits from the previous process at LEAVE (MAN15900) which clears the previous Q-block and waits. When a new Q-block is received by MANAGER, the program continues at QUEUE.

The program gets the initial of the task which sent the Q-block, and branches to the process indicated by the number in the Q-block heading. These labels are all similar to Q.C.2; a type 2 queue from COUNSEL (C02).

4.7. Q.C.2

Q.C.2 processes operator commands and runs from MAN42900 to MAN53500. The optional constants in the commands are converted to register R0 for all commands. R0 is zero for a missing constant. MAN42900 to MAN53500, the command type from COUNSEL is used to select the proper process from a branch table. Each command appears below with its options and actions.

TAPE [n] — If the tape is not ready or the write ring is missing, MANAGER responds with "CHECK TAPE". The 15 sector file LOCAL (LU5) is written after the tape is positioned. Either a parity error or no write ring (the program can't determine which) will leave the tape closed with a "CHECK TAPE" message. If the end-of-reel marker in sensed during the post-sounding tape write, the message is also logged, and a double EOF is written after the last complete file on tape before closing the tape. In all cases of tape error, MANAGER also sends a queue to the Scheduler (ANALYSER) directing it to abort any schedule in progress (SABORT).

n = 0 or missing: open tape. The tape is backspaced over any EOF's and then given a forward file command, except at the beginning of tape, to leave it properly positioned.

n = 1: open tape, skip to double EOF, and backspace over the second EOF.

Further tape control uses the following three commands.

FORMAT [n] n = Ø or missing: expanded format (default). An assembly parameter, MAN.FORM not zero, eliminates the expanded format to save memory space.

n > Ø: compact format with EOF's every n soundings.

SORT [n] n = Ø or missing: no sort (default).

n = 1: sort soundings into ascending frequency order; requires LU2 IONOSOND. LU2 is zeroed by the SORT command to eliminate the sectors for protected frequencies. During sounding, the data in PØ2 buffers are written to the proper sectors of LU2 (see Q.P.2, Sect. 4.1Ø), thus sorting them. After sounding, the non-zero sectors are packed to LU4, as if they had been recorded in frequency order. If the starting frequency, STEP, EPF, or HPE are not changed after a SORT, further zeroing is not needed. If they are changed, SORT should be called to make certain that the proper sectors are zero.

The assembly parameter, MAN.SORT not zero, deletes all SORT code from MANAGER, including the SORT command. The assembly parameter, MAN.PAR not zero, changes the SORT process. The MAN.PAR version writes the data blocks to LU2 with none of the packing required for LU4. LU2 is written as if it were a sequential file with records the right number of sectors long. This mode is much faster on an 8/16 and allows seven sector data blocks to be recorded every 4Ø milliseconds. This version is slower than DISK.INC = 6 on a 7/16.

BFILE Backspace the tape, one file. The tape must be on-line, but does not have to be selected.

FFILE Move the tape forward one file. The tape is positioned after the first EOF found. BFILE and FFILE are equivalent to the operating system commands, but require no argument.

CLOSE Close tape, write double end of file (EOF), and backspace. No tape will be written until a COPY or TAPE command; n for the TAPE command must be missing or Ø, if not, the tape may run away because a double EOF cannot be found.

4.7. Q.C.2

SAVE [m]
DROP [m]

Previous soundings may be saved on disc. The most recent sounding is always available to GRAPHER or ANALYSER. If MANAGER is in the 'DROP' mode, the next sounding is written over the current sounding, replacing it. If MANAGER is in the 'SAVE' mode, the next sounding is written after the current sounding. The data are packed into the LU4 buffer, normally IONOSOND.SAV, in the compact tape format discussed in Section 4.12.

MANAGER starts in the SAVE mode and remains in that mode until a SAVE or DROP is executed.

DROP [m]: MANAGER remains in the DROP mode for m soundings, replacing the current sounding and the next m - 1 soundings, then reverts to the SAVE mode.

SAVE [m]: MANAGER remains in the SAVE mode for m soundings and then reverts to the DROP mode.

If m = Ø or is missing, it is defined as 1.

If m equals or exceeds 9999, MANAGER remains in the mode without changing m.

NTAPE

When NTAPE is given before a sounding, the sounding will not be copied to tape, regardless of the status of TAPE. NTAPE is often given when running short soundings to test conditions and settings. The data are saved on LU4, unless DROPped, in which case it will be overwritten by the next sounding.

COPY [n]

Copy the n most recent files from LU4 to tape in compact format.

If n is absent, COPY writes all soundings saved since the last COPY or start-up. Less than n soundings are written if earlier files have been overwritten by wrap-around on LU4. A CLOSE command is executed at the end of this copying. No TAPE command is needed except to position the tape if required.

COPY might be used for writing tape after a sequence of soundings at maximum rate or for making all recording to a tape at once so as not to leave the tape on-line.

ABort should precede COPY because the time to write a tape may be long and COPY does not recognize SØ4 start Q-blocks.

TNEW — Rewind LU 4 and restart at sector 1.

The LU4 index block records the locations of the last 124 soundings, if they were short enough to all fit on LU4. The index block is only cleared by a TNEW, or when MANAGER is started if the first two words do not match what is expected, probably a newly allocated file or a change in the disc increment.

NOQUEUE [n] — n = Ø or missing: (default) ignore the ANALYSER task (only SØ4 and SØ6 Q-blocks are passed from SOUNDER). This saves the millisecond it takes to send a PØ2 Q-block and schedule ANALYSER to ignore it.

n = 1: pass Q-blocks to both GRAPHER and ANALYSER. MAN195ØØ to MAN217ØØ. PASS.ON and PASS.SE are used to pass the Q-blocks.

n > 1: send PØ2 Q-blocks to no one. This mode can be used with the GRAPHER command [PE to save the time spent queueing GRAPHER, and generating and refreshing the display.

4.8. Q.S.4

Input — SØ4 Q-block from SOUNDER, indicating the start of a sounding.

Some file size information from the SCT.

Output — R3: length of data record in sectors.

R6: highest sector used.

R7: length of data record in bytes.

Method — MAN266ØØ to MAN275ØØ. MD.HPE (half-words per echo) is multipled by MD.EPF (echoes per frequency) and doubled to give bytes per frequency. MD.OPOB (bytes in the PCT header) is added to give bytes per record.

4.8. Q.S.4

MAN27600 to MAN28700. Bytes per record is divided by 256 (bytes per sector) and rounded up to give the value in R3.

4.9. Q.S.6

Input — S06 Q-block from SOUNDER indicating that the sounding is complete.

The sounding on disc.

Output — One file of data on the tape if the operator has requested tape output.

Method — MAN28900 to MAN31000. If SORT is selected, sequential records from LU2 are read. If the second word is zero (this frequency was not sounded), the record is appended to LU4. After the last sounding data record has been recorded, the HKDIO and DIO.PTP data are appended to LU4.

MAN36600 to MAN39500. The appropriate sectors of LU4 are copied to tape. The count of files since the last EOF is incremented and if it matches the 'n' in FORMAT n, a double end of file (EOF) is written and backspaced one file. The next write will overwrite the second EOF.

Registers — R3: length of a record in sectors.

R6: highest sector used.

R7: length of a record in bytes.

4.10. Q.P.2 and Q.S.2

Input — P02 or S02 Q-block from PICKER.

R3, R6, and R7 from earlier calls to both Q.P.2 and Q.S.4.

Output — Record on disc, either on LU2 or compacted onto LU4.

Updated R6.

Method

MAN253ØØ to MAN264ØØ. For Q.S.2 insert MEAN, MAX, and PCT into the header and save on LU4. Ignore short records, less than MAN.SHRT = 16.

MAN244ØØ to MAN251ØØ. In the absence of SORT, only the writing of compacted data on LU4 is done.

MAN234ØØ to MAN243ØØ. The buffer from PØ2 is written onto disc at a computed sector location. MD.SFIX (starting frequency index) is subtracted from the frequency index for this buffer. The result is divided by MD.STEP (the frequency index spacing of the soundings) giving a quotient which is an integer number: Ø for the lowest frequency, 1 for the next frequency used, etc.

The quotient is multipled by R3 (the number of sectors per data record) and the product is the sector address used in the write. If the sector used is larger than R6, then R6 is set to that sector.

The actual length of the PØ2 block is used in the write so that only one sector may be written.

MAN2432Ø to MAN248ØØ. When SORT is selected for a mode larger than view mode (partial reflection mode), the data are written at sector R6 and R6 is incremented by R3 (sequential writes).

4.11. Q.G.2 and Q.G.4

Input

Requests for data from previous soundings.

Q.G.2, Q.A.2, and Q.P.6 request the SCT, and wait until the request from another task has finished.

Q.G.4, Q.A.4, and Q.P.4 get the next data block, and are rejected if they come from a different task than the one which selected the SCT.

Output

Data in the SCT array supplied by the SCT request.

A data record is returned by an MØ2 Q-block which contains a record pointer or zero (end of data).

4.11. Q.G.2 and Q.G.4

Method The LU4 file, while being useful for writing the compact tape format and for saving several soundings on disc to be written all at once, was designed to provide data from previous soundings to GRAPHER and ANALYSER. Either of these routines sends a GØ2 or AØ2 Q-block to MANAGER to identify the sounding and to get an SCT. They then send GØ4 or AØ4 Q-blocks for each data record. This process ends when an EOF is reached, an SØ4 (start of new sounding) is received, or a new GØ2 or AØ2 is given.

The discussion will refer only to GØ2 and GØ4 because AØ2 and AØ4 serve the same purposes for ANALYSER and discussing only one simplifies the discussion. A PØ6 can be used instead of an AØ2; and a PØ4 instead of an AØ4; to have the data sent to PICKER.

When GRAPHER wants data from a previous sounding, it sends a GØ2 Q-block to MANAGER in the following format:

Ø2 'G'	
STATUS	
A(SCT)	Address for new SCT
SNUM	Sounding number

MAN537ØØ to MAN577ØØ. The sounding number is zero for the current sounding, 1 for the previous sounding, 2 for the one before that, etc. This number is used to index into TAB to get the pointer to the sector containing the SCT. TAB recognizes soundings which are so old that they have been overwritten, by the absence of an SCT.

If the GØ2 comes while MANAGER is still working on an SØ6 (end of sounding) or on an AØ2 and AØ4 request, the GØ2 is stacked for later use. When the data are available, an MØ2 Q-block is sent to GRAPHER with the address of the SCT as the second word. If the data are not available this word is set to zero.

MAN578ØØ to MAN589ØØ. Data records are requested by a GØ4 block from GRAPHER to MANAGER. No information beyond the GØ4 is used from this Q-block. MANAGER gets a data record and returns the address as the second word in an MØ2 Q-block. The data record is in one of SOUNDER's data buffers, which might get zeroed as soon as an SØ4 is sent. This second word is set to zero for an EOF on the selected data block. A new GØ2 is required to either continue to the next sounding or to reread the previous one. A GØ2 requesting file number 999 (obviously illegal) will end the data request before an EOF is reached.

MAN39900 to MAN41200. Routine LEAVER is used as the exit from S06 and occasionally G04 and A04. Its purpose is to allow G02 and A02 Q-block calls to be stacked, especially after a sounding. For example: at the end of a sounding GRAPHER sees the S06, knows that the sounding is done, and sends a G02 to MANAGER; MANAGER is still busy writing tape so it stacks the G02 request. When the tape is done, an exit through LEAVER takes the G02 off the stack and processes it instead of exiting.

4.12. Disc and Tape Formats

Two different files, each with its own format, are kept on disc. Two tape formats are also available. All formats are based on recording an SCT block followed by as many P02 blocks as necessary. The P02 blocks are recorded unchanged and are generally called data records.

Logical Unit 2 (LU2) is usually assigned to the contiguous file IONOSOND. This file name is in a specific ASSIGN statement in the CSS file used to load and start a sounding. LU2 is only used if the SORT command has been given.

If SORT is non-zero, MANAGER assumes that frequencies are being sounded in a pseudo-random order and sorts the data records onto disc using LU2; or that LU2 is being used on an 8/16 to record data faster than it can be recorded on LU4.

No SCT is saved on LU2. A fixed number of sectors sufficient to hold the largest possible P02 block is allotted to each data record. In the typical format using 17 words per echo, two sectors are allotted because 8 echoes require 2 sectors. If the maximum number of echoes per P02 block were 6 or less, only one sector would be needed. If more than 13 echoes were saved, a third sector would be needed. Seven sectors per data block are required for 52 echo sets.

Using the current two sectors per record for an example, LU2 is formatted as follows:

SECTOR	CONTENTS
0, 1	P02(MD.SFIX)
2, 3	P02(MD.SFIX + MD.STEP)
4, 5	P02(MD.SFIX + 2*MD.STEP)
. . .	. . .
LAST-1, LAST	P02(MD.EFIX)

where: LAST = 2*(MD.EFIX - MD.SFIX)/MD.STEP + 1

It follows from this that the length of the file LU2 must be as great as LAST, which may limit the choices of MD.SFRQ, MD.EFRQ, and STEP. MD.SFRQ and MD.EFRQ are converted to frequency indexes MD.SFIX and MD.EFIX.

At the end of the sounding, LU2 is copied to LU4 for copying to tape and later reference.

Logic Unit 4 (LU4) is normally file IONOSOND.SAV and can be on either disc. This file should be very large so that it can contain several soundings for later reference or copying. If a sounding generates more data than LU4 can hold, the first-recorded data are overwritten. In such a case, the message, "DISC OVFLOW", is logged, MANAGER directs the Scheduler to abort any active schedule, and does not copy the data to tape at sounding end. MANAGER does not close the tape or alter the TAPE or FORMAT specifications for subsequent soundings.

Sector Ø of LU4 contains the following pointers:

Ø	256*DISK.LEN + LENGTH Hex 67C
1	maximum sector in file (modified length of file)
2	first unused sector
3	first sector of current sounding
4	first sector of previous sounding
...	...

S.LENG is the number of soundings which can be stored. It is normally 124, but there are no easy tests for overlapping soundings, other than a missing SCT.

The first unused sector will be used for the next sounding. Sector numbers start at 1 and are incremented by DISK.INC (6) until the sector number minus word 1 (maximum sector in file) is positive, non-zero. This small sector number is then the next sector number. See MAN738ØØ to MAN744ØØ for the coding.

Note that the word before the pointer to the desired sector contains the pointer to the next sector after the desired one.

The pointers in TAB within MANAGER are the same as recorded in sector Ø of LU4. These pointers point to a sector containing the SCT. The "compact tape format" is simply the copying of an integral number of 256-byte sectors (not more than 16) to a tape record and the writing of as many records as necessary. The last record in a sounding and the last record in the file (before wrap-around) may be shorter than the others.

A program reading the first record of a compact tape file would read exactly the same data and format as if that program read 4Ø96 bytes (16 sectors) from LU4 starting at the sector being pointed to, and stepping the sector address by DISK.INC for each sector read.

First two sectors: SCT block of 512 bytes.

Remaining sectors: the first two bytes are a bytes-left number starting with zero. The data are treated as 254-byte data blocks, ignoring the bytes-left numbers. The data format completely ignores sector boundaries.

These data are standard PØ2 Q-blocks in which the first word is the length of the data block. The next PØ2 block starts immediately after this number of bytes in the present block.

For example:

SECTOR BYTES	FILE BYTES	CONTENTS
Ø - 255	Ø - 255	SCT block
Ø - 255	256 - 511	rest of SCT block
Ø - 1	512 - 513	Ø, the first data sector
2 - 3	514 - 515	178, size of block with 4 echoes
4 - 179	516 - 691	remainder of the data block
18Ø - 181	692 - 693	144, size of block with 3 echoes
182 - 255	694 - 767	74 more bytes of the PØ2 block
Ø - 1	768 - 769	68, bytes left in the data block
2 - 69	77Ø - 837	last 68 bytes of the data block
7Ø -	838 -	the next PØ2 block

Additional data records are essentially copies of this.

A zero length of PØ2 block ends the data. The data blocks break across tape records in the same way they break across sectors.

4.13. Assembly Constants

MANAGER.CAL contains a set of information and conditional assembly flags. These flags are on the first page of the listing, and may be changed by the user.

MAN.SORT = Ø If non-zero, the code to sort an ionosond sounding is included. SOUNDER does not have a random frequency algorithm, however.

MAN.PAR = 1 If non-zero, multiple sector recording will be done faster on an 8/16 if the command SORT is given (see Section 4.4.).

MAN.BUF = Ø If non-zero, the disk sectors will contain successive integers as the first word, instead of containing the number of bytes left in the previous buffer.

MAN.FORM = 1 If non-zero, all tapes will be in the compact format and the FORM command will only control the insertion of file marks.

MAN.SHRT = 16 MAN.SHRT is the minimum length of S.Ø2 buffers which will be saved. Shorter buffers are ignored, and their PCT's removed from the stack.

DISK.INC = 6 DISK.INC is the sector increment to be used in reading or writing the disc and is the same for both the 7/16 and 8/16. The only usable values are: 24, 12, 8, 6, or 4 (4 is probably too small).

SECTORS = 16 A compact record will not contain more than SECTORS disc sectors of 256 bytes. It may contain less if there is neither enough data or enough room in memory to format the record.

HEADER = 256*15 HEADER is the length of the HEADER file, currently 15 contiguous sectors.

5. GRAPHER

Lorne D. Matheson

5.1. Introduction

Chapter 5 describes GRAPHER revision 2.01, which logs in dated 83 Jul 31. For details about changes made to GRAPHER 2.00, refer to Appendix A, dated 1983 November.

Task GRAPHER is responsible for displaying selected portions of the data in suitable formats on the refresh graphics device. It does analysis to select the data to be plotted and converts that data into plotter commands. GRAPHER also handles the display refresh. It has no responsibility for permanent archiving of any digital data. In the event that a camera is permanently attached to the refresh display, GRAPHER would be responsible for the camera control.

In future systems, still under the same basic organization, GRAPHER would handle the light-pen inputs from the refresh CRT. Graphics portions of ANALYSER tasks might be handled either by GRAPHER or ANALYSER.

Sections 5.1 through 5.13 contain all of the information necessary for normal operation of GRAPHER and may be considered an operator's manual. All of the GRAPHER commands are described in these sections. Section 5.14 is a brief description of the organization of the tasks, while Section 5.17 describes the plotter driver and hardware. Section 5.18 contains the software flowcharts. These latter sections document various phases of the processing and are needed only for fuller understanding, program modification, or debugging.

Fundamental to a thorough understanding of the present version of GRAPHER is an understanding of the refresh graphics hardware and the I/O driver, PLTDVR, written for it and resident in the system. PLTDVR is called by an SVC 1 call (see Section 1.7). The driver normally does an I/O initiate-and-proceed, using the SELCH for output without CPU intervention. At the termination of I/O, GRAPHER receives a Q-block and must re-initiate the I/O for the refreshing of the display. A fuller explanation of the driver, PLTDVR, and the hardware is in Section 5.17.

Almost all of the information required by GRAPHER is contained in the Q-block sent automatically by the other tasks, so very few inputs to GRAPHER are required from the console. These inputs will be discussed under the separate program overlays and subtasks to which they apply.

There are currently 25 commands accepted by GRAPHER. Below is a list of these commands and their form. These commands are discussed more fully in chapter 8 and in other sections of this chapter.

GAB	Section 5.13
GCAL	5.3
GHMAX hhh	5.4
GHMIN hhh	5.4
GMOV	5.3, 5.4
GNOCA	5.3, 5.4
GNOMV	5.2, 5.3
GNORM nn	5.9
GNPT[-]nnn	5.3
GONO	5.2, 5.3
GONX	5.2, 5.3
GOVER n	5.13
GSCALE x1,x2,y1,y2,c,a	5.7, 5.8, 5.9
GSCASK f1,f2,h1,h2,s	5.6
GPION nnn	5.13
GPSKY nnn	5.6, 5.13
GTHIN n	5.3, 5.4, 5.10
PLOTAMP n,hhh	5.7, 5.8, 5.9
PLOTPH n,hhh	5.7, 5.8, 5.9
[PA	5.2
[PB	5.2
[PC	5.2
[PE	5.2
[PS	5.2
[PZ	5.2

As an aid to understanding the task, you may consider the task to be in one of several states or subtasks that continue execution until GRAPHER switches to another subtask. These subtasks are listed below. The subtask concept was developed before the overlays were added. Subtasks 1, 3 and 4 could logically be combined.

Subtask 0 — Provides test patterns and plotter control; no other subtask has yet executed, or a [PC has just been executed. The display shows a test pattern or the grid for a sample ionogram. See Section 5.2.

Subtask 1 — Sweep-class sounding (I-, B- or Z-mode). The default display is a conventional ionogram, echoes on a frequency vs. virtual height plot. See Section 5.4.

Subtask 2 — Put Table-class frequency markers on the last ionogram plotted. The last frequency changed is specifically marked, if present. See Section 5.4.

Subtask 3 — View Table-class frequency for a short time. The data are not archived and are for only one frequency. See Section 5.5.

Subtask 4 — T- or W-class sounding (K-, P-, G-, or W-mode). The default display for K-mode is overlay 2, GRKIN. The default display for P-, G-, and W-mode is overlay 4, GRAMPH.

Subtask 5 — Ask for a previous SCT from disc in preparation for display of previous data.

Subtask 6 — Get the previous pulse data from the disc for display.

GRAPHER consists of three programs which are in the computer at all times. There is one overlay region. The resident programs consist of program task control, common subroutines and the plot buffers. The overlay resident depends on the type of display required. There are presently nine overlays. If the desired overlay is not resident, it is loaded just before the processing of the data occurs, not when the desired overlay is specified. GRAPHER is terminated to run CALL tasks, and is restored when the CALL task is ended.

5.2. Subtask Ø Test Patterns and Plotter Control

It may be important to verify the operation of the plotting software and hardware independently of the rest of the system's hardware and software. When the task is initially started, a standard test pattern is displayed consisting only of points and vectors. This pattern tests several features of the plotter controller and the refresh logic, but it does not test the character generator. Several long diagonal lines are plotted as a test of linearity.

5.2. Subtask Ø Test Patterns and Plotter Control

Several command sequences that can be used to plot lines and points, but which are not used by the normal ionogram plotting sequences, are used for testing. An alternate display is a sample ionogram grid, which does check the character generator. Several commands from the console that begin with the characters '[P' may be used to check and control the CRT plotting unit. These commands appear in Chapter 8 alphabetized without regard to the initial left bracket ([). Several commands beginning with G, whose primary functions are in other subtasks, also change the display.

[PA — Displays only the points from the test pattern. This may be used as a preliminary check on the point plotter, the fundamental block of the plotting hardware. The point test pattern checks quite a few modes of plotting points, including several not used by the normal ionogram plotting.

[PB — Displays only the vectors from the test pattern. Again, several modes are tested that are not used in the normal plotting.

[PZ — Displays the points and vectors in the test pattern, the same display that comes up when the task is initiated.

Once the task has started an ionogram, the plot buffer may overwrite the test pattern, so the above three instructions are treated as NO-OP's. These three commands are ignored after the first sounding and only are effective during start up.

[PE — Ends the refreshing of the display. This only affects the refresh logic, and does not change the generation of the plot buffer. If it is suspected that during periods that the transmitter is pulsing, there is too much to do in the allotted time, or the I/O requests are somehow interfering with each other, this command may give some relief. The SOUNDER command NOQueue 2 will save even more time. NOQ 2 does not disable the GPION, GPSKY commands.

[PS — Starts the refresh logic again. This rescinds [PE or may be used to restart the display if somehow an I/O interrupt was missed.

5.2. Subtask Ø Test Patterns and Plotter Control

[PE and [PS are always available, regardless of the subtask that GRAPHER is executing.

[PC — Generates an ionogram grid and labels from the SCT (Sounding Configuration Table, Section 7.1) stored in the task. A sample SCT is in the task at initiation time; it is changed to the last SCT seen by GRAPHER in Subtasks 1 or 5 later in the processing. This command is primarily used for diagnostics and checking task operation.

The display comes up with a test pattern of points, points and vectors, vectors alone, and points and vectors in a continuous cycle. This is the normal operational mode of the display. Alternation of points and vectors may be terminated by the command GNOMV and restarted by GMOV. GONX and GONO select points or vectors. These commands may be issued during a sounding or data replay from disc and take effect immediately. GONX and GONO have the same effect as [PA and [PB although the mechanism is quite different.

GONO — Suppress the last segment of a multi-segment buffer. Usually, plot only the ordinary wave.

GONX — Suppress the penultimate segment of a multi-segment buffer. Usually, plot only the extraordinary wave.

GNOMV — Suppress movement or change of the buffer segment alternations, plotting all simultaneously.

GMOV — Restore movement or alternation of multi-segment buffers.

5.3. Buffer Management and Control

All of the current nine overlays for data display use three segments in the plot buffer. The first contains the labeling and plot frame. Two additional segments are used for displaying O,X or two different data channels. The total buffer is of a fixed length and different options are available for utilizing the buffer in different ways.

Assume that a conventional ionogram has been put into the plot buffer and that there are reasonable numbers of both ordinary (O) and extraordinary (X) echoes. In this case, all of the O echoes are in the second buffer segment and all of the X echoes are in the third plotting buffer segment. The four commands GONO, GONX, GNOMV, and GMOV determine which segments reach the display device. In all cases, the first segment with the frame and labels is plotted. GONO (Graph ONly O) produces a steady picture with the first and second segments being plotted and the third segment with the X echoes being skipped. GONX (Graph ONly X) produces a steady plot with the first and third segments reaching the plotter and the second one being skipped. GNOMV (Grapher NO MoVement) produces a steady plot with all three segments reaching the plotter. GMOV (Grapher MOVement) always plots the frame, but slowly cycles through the second, second plus third, third and second plus third. GMOV presents the maximum amount of information to the user, but it is hard to photograph. The apparent motion bothers some experimenters, so they use one of the three commands resulting in a steady display.

At some point during a sounding or replay from disc, the buffer segments containing the O and X echoes may fill up. The management of this problem has three parts, using the GNPT and GTHIN commands. Some of this code was developed by Mike Jones. The user may only wish to keep the first points and study them. This alternative is selected by the optional minus sign in the GNPT command. In this case, O and X echoes are entered into the buffer segments until all of the available space has been used. Additional data is simply ignored, and the first points continue to be displayed. An alternative is to replace the oldest points by the most recent ones. This option is selected by not using the minus sign in the GNPT command. At least one quarter of the available space is allocated to the O returns and one quarter to the X returns, even if those returns never occur. The remaining half is split up according to the mix of O,X points after one or both of the dedicated quarters is full. After the space has been allocated during a sounding or replay, the ratio of O buffer space to X buffer space does not change for the rest of the sounding. In a conventional ionogram display with this option, the low frequency points will start disappearing after one of the segments is full. Typically, the lowest frequency displayed is not the same for O and X after the segments are full and cycling.

The user may wish to control the number of points displayed. nnn from the GNPT command sets the number of available points in buffer segments two and three. nnn cannot be less than 1ØØ nor more than is available at assembly time. This limitation occurs after the frame and labels have been plotted, so the user does not have to estimate how much of the total buffer space has been used by the frame and labels.

GNPT[-]nnn Set the plotting buffer overflow control and length.

Some of the display overlays accept n from the GTHIN command. For example, the conventional ionogram display, overlay one, accepts the GTHIN command. If n is 5, then only every fifth pulse set gets to the plotting logic. Suppose that an ionogram has a small STEP and that there are multiple echoes and spread conditions. Because of the limited buffer size, in this ionogram, only ten or fifteen per cent of the data might end up being displayed. In this case, GTHIN 5 would have the effect of displaying a larger portion of the frequency interval and the traces would be less dense. GTHIN has no effect on the amount of data recorded on the disc or tape. Not all overlays use the command, nor is the usage necessarily consistent from overlay to overlay. See specific overlay descriptions to see if that overlay accepts the GTHIN parameter.

GTHIN n Plotting overlay dependent point density control.

5.4. Overlay 1 Ionogram Mode

Overlay 1 displays a conventional ionogram, that is, virtual height as a function of frequency. The code for this overlay is in the program GRION. This overlay is the default display for I-, B- and Z-modes. The frequency axis is logarithmic and displays a minimum frequency range. The frequency axis will not start above 7 MHz. This range is automatically expanded to include all sounded frequencies in a run. If the displayed range does not start or end on a tick mark, that range is 150 kHz, 1.5 MHz or 15 MHz. The philosophy is to include at least one frequency marker that is labeled, and enough additional frequency tick marks to include all of the sounded frequencies. Points off the bottom or top of the virtual height axis are not plotted. For compatibility with Subtask 5, a sounding display from a previous sounding set, the SCT is copied into the task.

This overlay display is initiated by a S04 SOUNDER start Q-block with an I-, B- or Z-mode sounding and default overlays or a previous GOVER 1 command. At this time the SCT is copied into the task and the grid for a new ionogram calculated and displayed. The points are added to the plot buffer and displayed as each set is received from PICKER. The end-of-sounding Q-block, S06, causes the plot buffer to be stored on the disc, in GRAPHER.PLT, for possible later use by subtask 2, below. PICKER Q-blocks before S04 or after S06 cause error messages to appear on the console.

5.4. Overlay 1 Ionogram Mode

The PØ2 buffers normally contain a calibration pulse that, when plotted, occurs at a very low height; it is not usually plotted because it contains little useful information, when presented visually in this format. This overlay recognizes GNOCA, which causes echoes below 45 kilometers to be ignored. The plotting of calibration pulses is controlled by the following two commands.

<u>GCAL</u>	Allow calibration pulses to be plotted.
<u>GNOCA</u>	Plot no calibration pulses. Calibration pulses are those below 45 kilometers.

The above commands may be issued during a sounding or a replay from disc and take effect immediately. Commands issued during a sounding, particularly at a 1Ø ms pulse interval, may lead to timing problems due to the time required to decode the message.

The minimum and maximum heights on the ionogram plot may be changed. This only changes the display, not the data recorded. Any change takes place at SØ4 time before the next ionogram. The following commands specify heights that are truncated to 5Ø's of kilometers of virtual height; at least two labels must appear on the height scale. GTHIN n plots only every nth pulse set.

<u>GHMIN</u> hhh	Set the minimum height to hhh km.
<u>GHMAX</u> hhh	Set the maximum height to hhh km.
<u>GTHIN</u> n	Plot every nth pulse set.

The plot normally alternates display of the ordinary (O) and extraordinary (X) echoes. The display may be altered by the following commands.

<u>GNOMV</u>	Suppress alternation of the O and X echoes.
<u>GMOV</u>	Restore alternation of the O and X echoes.
<u>GONO</u>	Plot the O echoes without the X echoes.
<u>GONX</u>	Plot the X echoes without the O echoes.

The differentiation of O and X echoes on the plot requires a means of distinguishing between them in the data. Since this discrimination is needed for each echo but only influences the display and does not change the original data, the computation should be fast and sometimes wrong, rather than slow and always correct. The same algorithm is used for all display modes. The algorithm would like values from orthogonal, physically coincident antennas at the same time. Using our current four-antenna array and sounding sequence, and averaging the N-S antennas and the E-W antennas, the orthogonal, and physically coincident criteria are satisfied, but one set of data is one pulse time later. Therefore, a large doppler shift may confuse the O-X determination. Other antenna sequences may be confused by off-vertical returns, etc.

The method compares the phase of a return on one antenna with the phase on the orthogonal antenna. Ideally, these would differ by 90 degrees with one sense of rotation being O and the other X. The method actually used corresponds to rotating both returns such that the first lies on the plus X axis. The second return is then in the upper or lower half plane, corresponding to O or X. Whether the upper half plane is O or X is best determined by experiment with real data. This test is now in a SOUNDER reentrant subroutine (XORO, Section 2.6.3). The test used is the sign of

$$-X_B*Y_A + Y_B*X_A$$

where

X_A is the first component of the first antenna;
Y_A is the second component of the first antenna;
X_B is the first component of the orthogonal antenna;
Y_B is the second component of the orthogonal antenna.

Subtask 2
T- or W-Class Frequencies on Ionogram Plot

This subtask is used to compare frequencies selected for a T- or W-class sounding mode with the last ionogram plotted. The code is in program GRION, overlay 1. Execution of this subtask is initiated by a S08 Q-block. Initiation of this subtask forces the loading of overlay 1, regardless of the last GOVER command. If an ionogram plot buffer is not present, one is read from the disc file GRAPHER.PLT. The frequency index markers are added to the plot. If the frequency indicated in S08 is not zero, that marker is specifically identified by the character "S".

5.4. Overlay 1 Ionogram Mode

SØ8 Q-blocks are initiated by the commands F and RNGE, as well as MGET on a T- or W-class mode, if at least one frequency is set.

Because the plot buffer rather than raw data is read from the disc, GHMIN, GHMAX, GCAL, and GNOCA have no effect until the next ionogram or kinesonde sounding. GMOV, GNOMV, GONO, AND GONX work properly. When the plot buffer is read in, many of the constants set by GRAPHER are restored to the values present at the time the buffer was written. This file is also read and the constant values restored after a CALL task termination and GRAPHER restoration.

The file of plot commands for the last ionogram is written when an I-, B- or Z-mode sounding occurs with overlay 1 resident. The file is also written when a GPION Ø or GPSKY Ø is executed on an I-, B- or Z-mode with overlay 1 resident. As a part of the same write, many of the constants set by GRAPHER commands are written as part of the file.

5.5. Overlay 2 Kinesonde Run

This overlay is initiated by an SØ4 Q-block that results from a RUN command on a K-mode sounding with default overlays. The code is in program GRKIN. A K-mode run contains from 1 to 1Ø frequencies. The main purpose of the display is to allow a subjective evaluation of the chosen frequencies during the run. The basic display is the plot of the X-Y components of the return.

A run with one frequency has a display identical to the Kinesonde View display. In a single frequency display, two plots are displayed, the left one at frequency F and the right one at F + delta F. When more than one frequency is present, just the basic frequency is plotted, with no plotting of the F + delta F return. Two- and three-frequency plots have the same size graphs as the one-frequency plots, with four or more frequencies having the basic graph one-half the size. The frequencies are plotted in order of their occurrence in table SCT.FTBL (Section 7.4). The first valid frequency plots in position one, the second in position two, etc. See Figure 5.1 for the sequencing of frequencies on the display. GCAL, GNOCA, GMOV, GONO and GONX all operate immediately. The normal alternation of points is between the ordinary and extraordinary returns.

Subtask 3 Kinesonde View

This subtask is initiated by an SØ4 Q-block with the third word being 4, which results from a VUF n command. The nth frequency is sounded repeatedly, eventually followed by an SØ6 Q-block. The X-Y components of the returns are plotted. Two plots are displayed: the left one at frequency F and the right one at F + delta F. This plot is identical to a one frequency kinesonde sounding. GCAL, GNOCA, GMOV, GNOMV, GONO, and GONX all operate immediately. The normal alternation is between ordinary and extraordinary returns. The center of a short vector indicates the center of the two coordinate systems. The scales of each plot presently correspond to plus or minus 3 volts at the input to the FEP ADC's. Full scale for the ADCs is 1Ø volts. Command VUF is not implemented in SOUNDER, although V-mode will invoke subtask 3. (See V-mode under MAKEM, Section 1Ø.2.1.)

5.6. Overlay 3 Skymap Display

This display shows the apparent point of reflection with the zenith being the center and the scales being kilometers from the zenith, assuming the reflection takes place at the virtual height. North is up with east to the right. The inner marks are at one-third the distance of the outer marks. The default range is 6Ø kilometers from the center to the outer marks. The code is in program GRSKY.

There are two sets of conditional assembly in the code. The variables CHAN1 and ROTA select the options. If CHAN1 is not zero, a constant correction of fifteen degrees is applied to all receiver one data. As supplied, CHAN1 is set to zero, so the correction is not made.

The display is rotated by 45 degrees if the assembly parameter ROTA is not equal zero. This is necessary if the receiver one, antenna one, antenna is actually NE of the center of the array, rather than N. As supplied, ROTA is set to one to rotate the data display and correctly align north. It is not obvious how to insure that all of the hardware phasings are correct so that the formula works correctly and is in the right sense. The computations are critically dependent on correct phasing of all of the receivers. Incorrect phasing may confuse O-X separation, change the sense of rotation in the kinesonde data, or confuse N-S deviation from the zenith, but the basic display looks correct.

5.6. Overlay 3 Skymap Display

The Skymap display may be generated by a GPSKY nnn command for data off the disc or a GOVER 3 command to force GRAPHER overlay 3 to be resident for real-time display during sounding runs. GCAL, GNOCA, GMOV, GNOMV, GONO, and GONX all operate immediately. SFMAX and SFMIN in GRAFBUF are set by the GSCASK fields f1,f2 and are the minimum and and maximum frequencies in kHz to be plotted on the display. The parameters, h1 and h2, are the minimum and maximum heights, in kilometers, to be plotted on the display. The distance from the edge of the display to the center is s, in kilometers. The default is 6Ø km.

GSCASK f1,f2,h1,h2,s   Skymap window and scaling parameters.

5.7. Overlay 4 Height vs. Amplitude or Phase.

This overlay was developed by Mike Jones for his partial reflection work and plots amplitude or phase of all the echoes, typically in a partial reflection sounding (G-, P-, or W-mode). Additional features were added by USU. The program name is GRAMPH. All of the data is plotted over a specified height range. The plots are properly labeled. Either 8Ø dB of amplitude range or 36Ø degrees of phase range is plotted. An optional offset may be added to the second amplitude to separate the traces. Two channels of data are plotted in two different plot buffer segments. The user has control over the data used for the display. An optional conversion from linear components to circular polarization may occur. If X1,Y1 are the x and y components for the first channel, X2,Y2 the orthogonal channel, and x1p,y1p are the first channel of circular polarization then the conversion is:

```
x1p=X1-Y2        y1p=Y1+X2
x2p=X2-Y1        y2p=Y2+X1
```

The n from PLOTAMP or PLOTPH is from 1 to 8 and determines the pulse, receiver channel data to used as X1,Y1. The parameter, a, from GSCALE is the number of bytes later to use for the second data set, X2,Y2. A value of 4 will pick up the second receiver for that pulse, if the first receiver was selected as X1,Y1. For the normal ionogram sounding sequence, n = 3 and a = 8 will pick up the E and N antennas at f + delta F. The value of c from GSCALE determines if linear to circular polarization conversion takes place or not. Ø causes conversion, 1 means no conversion. y1,y2 from GSCALE in kilometers determines the range of heights plotted. The PLOTAMP and PLOTPH commands set a variable which is used by overlays 4 and 5 (GRAMPH and GAMPHT) to determine if phase or log amplitude is to be plotted when the overlay is executed. The most recent of these commands takes precedence. Default is amplitude plots.

The horizontal scaling is set by internal variables at assembly time. These are at the end of the program to help in locating them for possible changes. MINPH and MAXPH set the default phase limits, with Ø to 16Ø8 being Ø to 36Ø degrees. MINAM and MAXAM are the log amplitude limits with Ø to 2Ø48 being a range of 8Ø dB. In amplitude no correction is made for attenuation changes. Ø dB is one FEP ADC count. In some cases, it may be desirable to add a constant offset to the second amplitude to distinguish the channels from each other more clearly. A variable in the GRAFBUF save area, AM2OF, is added to the second channel. AM2OF is presently set to zero, but a value of 512 would offset the second channel by 2Ø dB.

The vertical scaling is set by y1,y2 from GSCALE. y1,y2 are in kilometers. Echoes with zero or negative delays are ignored. All other echoes are plotted, either on scale or at the top or bottom.

<u>GSCALE</u> x1,x2,y1,y2,c,a   Set scaling parameters and data source.

<u>PLOTAMP</u> n,hhh   Plot amplitudes, select data source.

<u>PLOTPH</u> n,hhh   Plot phase, select data source.

5.8. Overlay 5 Phase or Amplitude vs. Time

This display was developed by Mike Jones for his partial reflection work and plots a time-series of phase or amplitude of a selected echo. The program name is GAMPHT. Much of the code and control is similar to that in GRAMPH, overlay 4. Two channels of data are plotted as a function of time in two segments of the plot buffer.

The two channels of data are selected by the pulse number, n, in PLOTAMP or PLOTPH, as well as the antenna increment, a, from GSCALE. The conversion parameter, c, in GSCALE determines if linear to O,X conversion occurs. The last PLOTAMP or PLOTPH determines if log amplitude or phase is plotted. So far, this is similar to the operation of GRAMPH, overlay 4.

Only one echo per channel is plotted. It is the first one at or above hhh kilometers from the last PLOTAMP or PLOTPH command. In a partial reflection mode, the display plots data from a constant height. In a Sweep-class sounding, if hhh were 12Ø kilometers, then the plot would be the minimum height of 2 hop E, or the F region, O or X. If no echoes are present meeting the height criteria, then no points are plotted. GSCALE x1,x2 scales the horizontal time axis.

The horizontal axis is time, running from x1 to x2. The x plotted is the pulse set number times 4. GSCALE Ø,1Ø24 would plot points from the first 256 pulse sets of the sounding. The vertical axis has tick marks and labels for 8Ø dB or 36Ø degrees as in GRAMPH. The default ranges are in MINAM, MAXAM, MINPH and MAXPH as in overlay 4, GRAMPH.

5.9. Overlay 6 Power Spectra Display

This display was developed by Mike Jones for his partial reflection work. The program name is GRAFFT. This overlay does a Fourier transform on a complex time series. The amplitude of the resultant transform is plotted on a logarithmic scale. The same operations are performed on a second data set and the results put in the other buffer segment. As this overlay uses space in other tasks for temporary storage, THIS OVERLAY MUST NOT BE RUN FROM A SOUNDING, ONLY FROM A GPION OR GRSKY COMMAND. (GRAFFT has not been thoroughly debugged at this writting, and does not function properly in Product Two.)

The last n from a PLOTAMP or PLOTPH determines which pulse, receiver channel is in the first data set. n is in the range of 1 to 8. 1 is pulse 1, receiver 1; 2 is pulse 1, receiver 2; and 8 is pulse 4, receiver 2. The GSCALE parameter, 'a', determines the offset in bytes to the second data set and should be a multiple of 4. The parameter c from GSCALE determines if linear to O,X conversion occurs. If c equals zero, then conversion occurs and if c equals 1, then no conversion occurs.

The horizontal axis of the plot is frequency and runs from minus the Nyquist frequency to plus the Nyquist frequency. The Nyquist frequency is the reciprocal of twice the time interval between data samples. If there were one pulse every 2Ø milliseconds with an additional 2Ø millisecond interval between pulse sets, then the sampling interval is 1ØØ ms. The Nyquist frequency is then 1/.2ØØ or 5 hertz. The plot would run from -5 to +5 hertz. GNORM nnn controls an internal shift to prevent overflow.

GSCALE 8,1024,1200,2900 and GNORM 3 are reasonable starting values. A range of 2048 in y is 80 dB.

GNORM nn — Set spectra normalization shift. Allowed values are 1 to 16. Overlay 6 automatically increases n to avoid overflow, but will not decrease it. To display the current value of n, type GNORM.

5.10. Overlay 7 Plot Signal and Background.

This overlay plots the log of the mean-peak data vs. frequency. The data source normally is a Sweep-class sounding, I-, B- or Z-mode. The plot may be useful for picking Table-class frequencies. Strong bands of interference show up if present and it is relatively easy to pick frequencies that have strong returns and little interference. The code is in the program GRASIN.

The data from pulse 1, receiver 1 is plotted with all attenuation included. GTHIN is the only command influencing the plot, with every n-th point being plotted.

The log amplitude axis in labeled in dB, with 0 dB being an FEP ADC amplitude of one unit with no attenuation in the RF or IF. The dB range plotted is set by the variables HMINX and HMAXX, which are indexed in various internal arrays. The horizontal frequency axis is scaled from information in the SCT in the same manner as in overlay 1, GRION, the ionogram display.

5.11. Overlay 8 Height vs. N-S Location.

This overlay, written by USU, plots selected echo heights corrected by angle of arrival vs. their N-S deviation from the zenith. Much of the code came from the skymap code in GRSKY (overlay 3). The overlay code is in program GRHVT.

Echoes are selected in a frequency window and, optionally, above the calibration height. The N-S and E-W deviations are calculated as in skymap and used to correct the total delay to corrected height. The corrected height is plotted vs. the N-S deviation. O,X discrimination determines the point plot buffer segment.

The parameters f1, f2, in kilohertz from the command GSCASK determine the frequency acceptance window. GCAL or GNOCA set the usage of calibration pulse data. The GSCASK parameter, s, in kilometers, determines the horizontal range.

5.11. Overlay 8 Height vs. N-S Location.

The horizontal axis is minus to plus s kilometers. Tick marks are also at Ø and s/2 kilometers. The vertical axis is set at Ø to 8ØØ kilometers. Tick marks are at Ø, 1ØØ, 3ØØ, 5ØØ, 7ØØ and 8ØØ km.

5.12. Overlay 9 E-W or N-S Deviation vs. Time

This overlay, written by USU, plots selected echoes as E-W or N-S deviations vs. time. Echoes are selected by a frequency window and, optionally, above the calibration pulse. Code similar to that in the skymap overlay, GRSKY, calculates the N-S and E-W deviations. Standard O-X discrimination determines into which plotting buffer segment the point is put. The selection of N-S or E-W depends on the pulse number to be plotted. The source code is in program GRXYT.

The commands GCAL or GNOCA determine the lowest virutal height accepted. GNOCA suppresses all echoes below 45 kilometers. f1,f2 in kHz from GSCASK determine the frequency range to be plotted. PLOTAMP n or PLOTPH n plots N-S for n=1, E-W for n=2. s in kilometers from GSCASK determines vertical scaling. GSCALE x1,x2 determines the horizontal scaling.

The x1 and x2 from GSCALE are the minimum and maximum x plotted. The pulse number is multiplied by four to get an x. There are tick marks at the left and right edges. The vertical axis runs from minus s kilometers at the bottom to plus s at the top. A horizontal line runs through the zenith.

5.13. Subtasks 5 and 6 Display Prior Data Set

These tasks are initiated by the GPION and GPSKY commands described below. GRAPHER asks MANAGER for the proper data set. The plotting options are those currently in use for the particular type of plot. Since the data are reprocessed, scaling may be changed, calibrations added or deleted from the plots, a different type of display selected, etc.

GPION nnn — Get the previous sounding data set number nnn, where nnn is Ø for the last set stored and counts back in time.

GPSKY nnn — Get the previous data set as in GPION and display in SKYMAP mode.

GAB — Abort a GPION or GPSKY sequence in order to examine a specific portion of a sounding or to start a new GPION or GPSKY sequence or a new sounding.

GOVER n — Force GRAPHER overlay n (n = 1 to 9) for processing and display. GOVER Ø restores automatic selection of overlay. OVERLAY 6, GRAFFT, MUST NOT BE RUN WHILE IN THE SOUNDING MODE, ONLY DURING DATA REPLAY FROM THE DISC. See Section 5.9.

5.14. GRAPHER Task Organization

All changes in subtask state are caused by SØ4 Q-blocks from SOUNDER, or the GPION nnn or GPSKY nnn commands and their MANAGER responses.

At present, I/O consists of refreshing the plotter, and writing the last ionogram plot buffer to the disc and retrieving it when necessary. The refresh is handled by a standard SVC 1, initiate and procede, under SELCH control with a queue message when the I/O is finished. The task must then re-initiate the I/O for refresh graphics to continue. After the first ionogram has started, the test pattern may have been overwritten and is considered no longer available. The CRT plotter is assigned to logical unit 9 and the file for the plotter buffer, SYS1:GRAPHER.PLT, is assigned to unit 8. Unit 7 is assigned as GRAPHER.LMO, the overlay file. Units 7 and 8 are programmed as I/O and wait until complete.

Presently nine overlays are used for processing different types of displays. They are on file GRAPHER.LMO and are named GROV1 through GROV9. All of the overlays can plot both ionogram and kinesonde data in some fashion. Each overlay has at least three parts, one to do the frame and setup, the second to process the PICKER pulse set, and the third to do postprocessing, if necessary.

Rather than having large tables of overlay entries, all overlays have a standardized entry system, so that the name of the overlay 1 initialize and draw-the-frame entry, DRAWIF, is at the same location as that function in all of the other eight overlays. Each overlay starts with a word containing the overlay number, 1 to 9. The next word is the entry for postprocessing and this area is two words long. The next two words are the initialize and draw-the-frame entry, followed by the process-a-PØ2-buffer entry. All other entries are either just for aid in locating parts of the program or are overlay specific. Any changes to the existing overlays or any additional overlays must follow these standardized entries.

GROV1 plots S-class data, GROV2 K-mode data, and GROV3 makes a SKYMAP plot of any data. Overlays are read by the SVC 5 command. Overlaying is rather primitive and requires the calling task to keep track of the position in the overlay file. The overlay file may only be read forward or rewound and read forward. The rewind and read generates an I/O queue entry upon completion of the rewind. GRAPHER is programmed to ignore this queue.

Three problems existed previously in GRAPHER and its interactions with the system. Two of these have been corrected by code. An additional Q-block, MØ4, was added to stop the refresh CRT during tape writing and to start it again upon completion of the tape writing. This is necessary to consistently write tape records of the correct length. An intermittant error exists in the hardware or software, or both, that only shows up during simultaneous tape write and CRT refresh. We hope this problem will be solved in the future and the display will continue to operate while the tape is being written. Having the display off during the tape write speeds up the tape motion considerably.

An additional modification to GRAPHER was necessary to prevent program hangup if the display is missing or bad. If more than five consecutive I/O calls to the plotter are bad, the refresh logic is turned off just as by a [PE. I/O may be tried again by the [PS for at least five times. GRAPHER also now operates without hanging up even if the MT2 OS contains no plotter (PLT:) driver.

A problem, not corrected by software, exists if the plot buffers take too little time. Since GRAPHER has a higher priority than ANALYSER, ANALYSER may be locked out; [PE allows ANALYSER to be scheduled.

5.15. Task Establishment

The GRAPHER task is established by running TETGRAF.CSS. This CSS file presently requires twelve compiled programs and generates three output files. The three output files are the .TSK file containing the three resident programs, the .LMO file containing the nine overlays, and the .MAP file containing the task map. TETGRAF requires three arguments; the output file name, the bias for the task, and the priority. A typical request might be

```
TETGRAF GRAPHER,6CØØ,9Ø
```

This would generate the files GRAPHER.TSK, GRAPHER.LMO and GRAPHER.MAP. The task would be established at hex address 6CØØ with a priority of 9Ø. The execution of the CSS file lists all of the required programs. The CSS file should be used as an example if the overlay structure is changed. The total length of the task is hex 2396. Below is a list of the programs and some of their specifications. As small changes are made, the numbers may change slightly, but the list gives a feeling for the size of the various programs.

Program	Code lines	Hex length	Description
GRAFPRO	9Ø9	9DC	Basic driver and I/O
GRAFRAM	35Ø	56C	General purpose routines
GRAFBUF	529	EØØ	Plot buffers and constants
GRION	537	672	Ionogram display overlay
GRAKI	252	27E	Kinesonde display
GRSKY	533	51Ø	Skymap
GRAMPH	454	558	Phase or amp vs height
GAMPHT	439	4EA	Phase or amp vs time
GRAFFT	533	582	Power spectra
GRSIN	4Ø3	4EC	Peak, mean vs freq
GRHVL	549	552	Height vs latitude
GRXYT	533	52Ø	NS or EW dev vs time

A CALL task, GXREF, is useful for looking at the EXTRN, ENTRY references and linkages between the twelve programs. It needs all twelve of the .OBJ programs available. If the names of the programs change or additions or deletions are made, edit GXREF.CAL, recompile and reTET. TETRUN GXREF,6CØØ,65 will establish the task. GXREF can run standalone. RUN GXREF initiates the task. There are six commands available.

5.15. Task Establishment

BUILD	Build labels table (redundant)
LIST <PROG>	List program <PROG> to CON:
LIST	List all programs to LPR:
UNREF	List all unreferenced ENTRYs to CON:
<label>	List all references to ENTRY labeled <label> to CON:
END	End task.

5.16. Plotter Buffer Management

Critical to the ability to add overlays or make changes in the display is a knowledge of the workings of the plot buffer and how it is filled and updated. Section 5.3 describes how the buffer is controlled and how it appears to the user. This section will explain the present workings of the plot buffer in more detail. All of the nine display overlays handle the buffers in the same way. The present software contains only a fixed frame, continuously displayed, and two sets of points, one or both of which are displayed, possibly alternating. Other configurations are possible, but the only possible change that has any code written provides for the possibility of alternating labels or a portion of each of the two point segments that do not cycle.

A set of common subroutines in GRAFRAM and GRAFBUF supports the buffer management. None of the display overlays puts words in the plot buffer directly. All plotting words are left in registers and a subroutine called or are generated and put in by a called subroutine. Typical subroutines for this purpose are GSTUFF, GSTUFT, GSTUF2 and GSTUT2, all located in program GRAFBUF. All of the buffer insertion subroutines work with multiple pointers. When the second and third buffer segements are being used, there are ten pointers controlling the insertion of plotter commands.

We will now follow the process of plotting a typical overlay, either from the disc or from a sounding. The process starts with the Sounding Configuration Table (SCT) being available, either at SØ4 time or after the MANAGER response back to GRAPHER after a GPION or GPSKY command. The first thing that happens at this time is that the proper overlay is loaded, if it was not already resident. The program is now at label SO.4D in program GRAFPRO.

The buffer control is set to one buffer by calling GST1BUF. This call also puts several commands in the plot buffer such as reset the plotter, reset the light pen, and set intensities. The time and station line at the top of the plot is added by calling DRAWST in program GRAFRAM. The rest of the frame is drawn by calling the portion of the overlay that is called at SØ4 time. The entry is DRAWIF in overlay 1 and the rest of the overlays have the same standardized entry system. GST3BUF is now called, setting up three buffer segment plotting, pointers and controls. If the machine is sounding at this time, nothing is added to the plot buffers until the countdown sequence is complete and pulse sounding starts.

Most of the plotting overlays put in one or more points for each PØ2 buffer, once the sounding starts, unless GTHIN is ignoring points. GRAFFT only adds selected points to an internal buffer and does not add plotted points until postprocessing time. After a SØ6 or the equivalent from disc retrieval, the postprocessing takes place. GRAFFT is the only display overlay presently doing extensive postprocessing.

There are ten pointers in use during the process of adding plotted points to the second and third buffer segments. They are listed roughly in order.

PLBFB	Bottom of the second segment
PLBFB1	Bottom of the cyclic portion of the 2nd seg
PLBFPO	Pointer for next addition to buffer
PLBFPL	Top of cyclic portion of 2nd segment
PLBBOT	Possible top of second segment
PLBTOP	Possible bottom of third segment
PLBLS	Bottom of cyclic portion of 3rd segment
PLBUP	Pointer for next addition to 3rd seg
PLBFC1	Top of cyclic portion of 3rd seg
PLBFC+1	Top of third segment

The following is not precisely correct, but conveys the general impression of the operation of the pointers and plot buffer. After the first segment has been filled with the frame, PLBFB, PLBFB1, PLBFPO and PLBFC are set to the next available location. PLBLS, PLBUP, PLBFC1 and PLBFC are set to the top of the buffer. PLBTOP is set to one fourth of the way between PLBFB and PLBFC, while PLBBOT is set to three fourths of the distance. Assume that just commands for plotting points are being added to the buffer segments. As points are added to the second segment, PLBFPO and PLBFC are incremented. Eventually these counters may reach PLBTOP. At this time, PLBTOP is incremented along with

PLBFC. As points are added to the third segment, PLBLS and PLBUP are decremented in a similar fashion. PLBBOT is also decremented when PLBLS pushes it down. If enough points are added to the second segemnt, PLBFC eventually reaches PLBBOT. At this point, the length of the second segment may not increase. If buffer cycling is enabled, PLBFPO is reset to PLBFB1, and the buffer segment space is reused, with the newest points replacing the oldest ones. If buffer segment recycling is not enabled, all additional points to be added to the second segement are ignored. The SVC 1 parameter list uses PLBFB and PLBFC for the second segment. Similar logic applies to the third segment.

If alternating labels are to be used, they would be added at the bottom of the second segment and the top of the third. PLBFB1 and PLBFC1 would be set so that the recycling points do not overlay the label commands. Once again, this discussion is not precisely correct, but should give the reader the general impression so that the code makes sense.

5.17. Refresh Graphics Hardware and Driver

The hardware operates as a 16-bit device. Each 16-bit word consists of 4 bits of command structure and 12 bits of additional data, often CRT screen addresses. It typically takes two words to specify a point, one of X and one for Y. In order to plot a vector, the point plotter must be set to the start of a vector with the intensity off and then the increments in X and Y must be sent to the vector generator. Interactions exist between the character generator and the point and vector generators. The time required for the device to be ready to accept the next word can vary between 1/2 microsecond (accept the next word of an X-Y pair) to milliseconds (plot a long vector at high intensity).

The driver for the CRT, PLTDVR, operates with standard SVC 1 calls, using the SELCH, with one exception. If the last word address of the buffer area to be output is even and less than 16 words from the start, then the area referenced is to be interpreted as a table of first-last address pairs. The driver runs through the table before considering the I/O finished and before sending a Q-block/interrupt to the calling task, or before returning to the calling task, if the call was I/O and wait. This feature was added to the driver so that a plot buffer can be constructed with different segments (O and X components) and some or all of the segments plotted without shuffling the plot buffer. This feature of the driver can also be used to blink particular features of the plot. The driver also contains the basics needed to read the light pen, but GRAPHER does not presently use the light pen. The light pen address is stored in the SVC 1 parameter block at SVC1.RAD + 2 when the light pen is active.

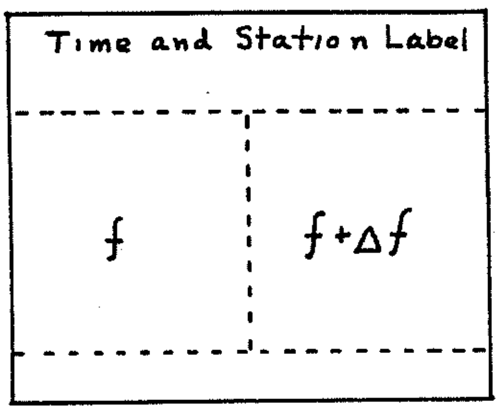
SVC1.RAD is zero upon exit unless the light pen was active. SVC1.RAD upon entry contains the number of times the buffers are to be plotted (once for ≤1).

The driver must decide if the starting address and ending address refer to one buffer or to a table of buffer segments. If the parameter block addresses are more than 2Ø apart, one buffer is assumed. If the end address is odd, one buffer is assumed. Otherwise, these words point to a table of pointers to buffer segments SAD1, EAD1, SAD2, EAD2, SAD3, EAD3, etc.

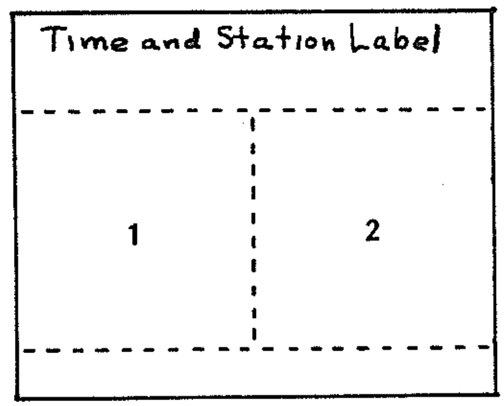
The SVC 1 parameter block must have a standard MT2 function code of ØØ1x xxxx, write data transfer. Any other function code will be rejected with a 'CØ' device independant status. GRAPHER uses a function code of hex '24'. Several other errors may cause status errors:

STATUS	CAUSE(S)
'CØ'	Function code not ØØ1x xxxx Starting address odd. Starting address less than ending address.
'9Ø'	CRT not available for some reason.
'84'	System abort.
'82'	Timeout.

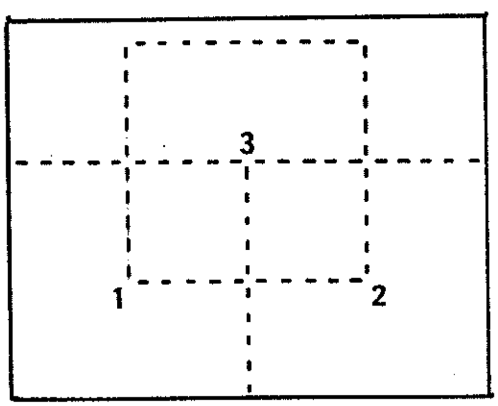
5.18. Figures



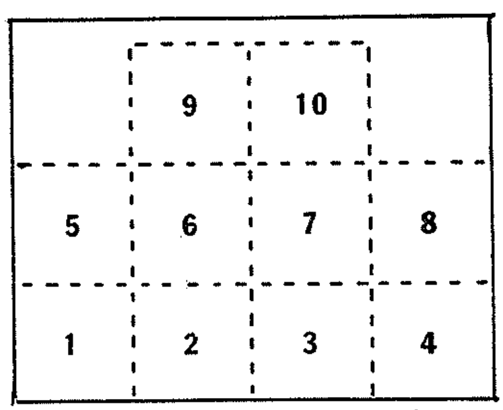
One Frequency



Two Frequencies



Three Frequencies



Four to Ten Frequencies

Kinesonde Scan & Kinesonde View Displays
Figure 5.1.

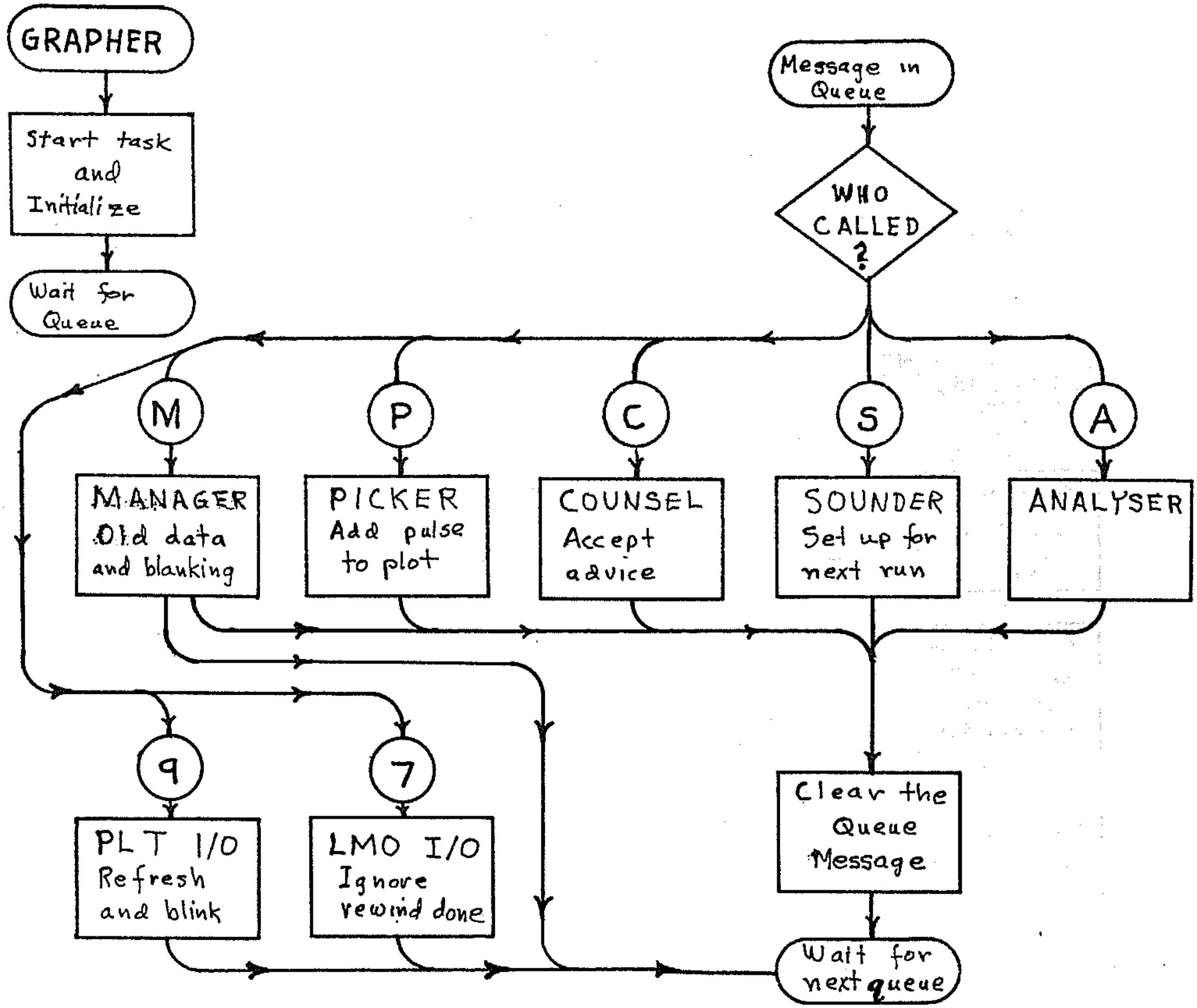


Figure 5.2. GRAPHER & Queue Processing

5.18. Figures

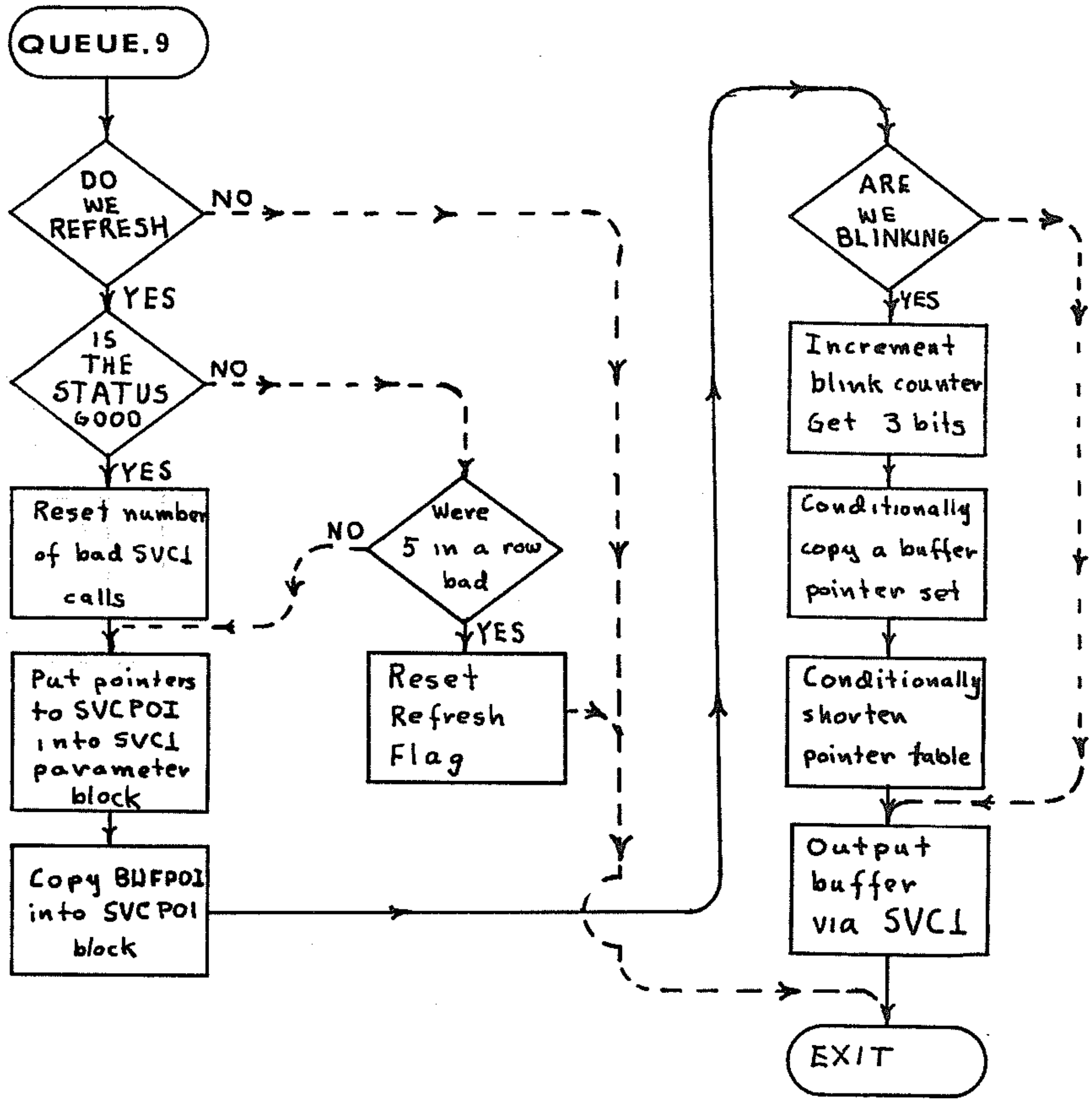


Figure 5.3. QUEUE.9

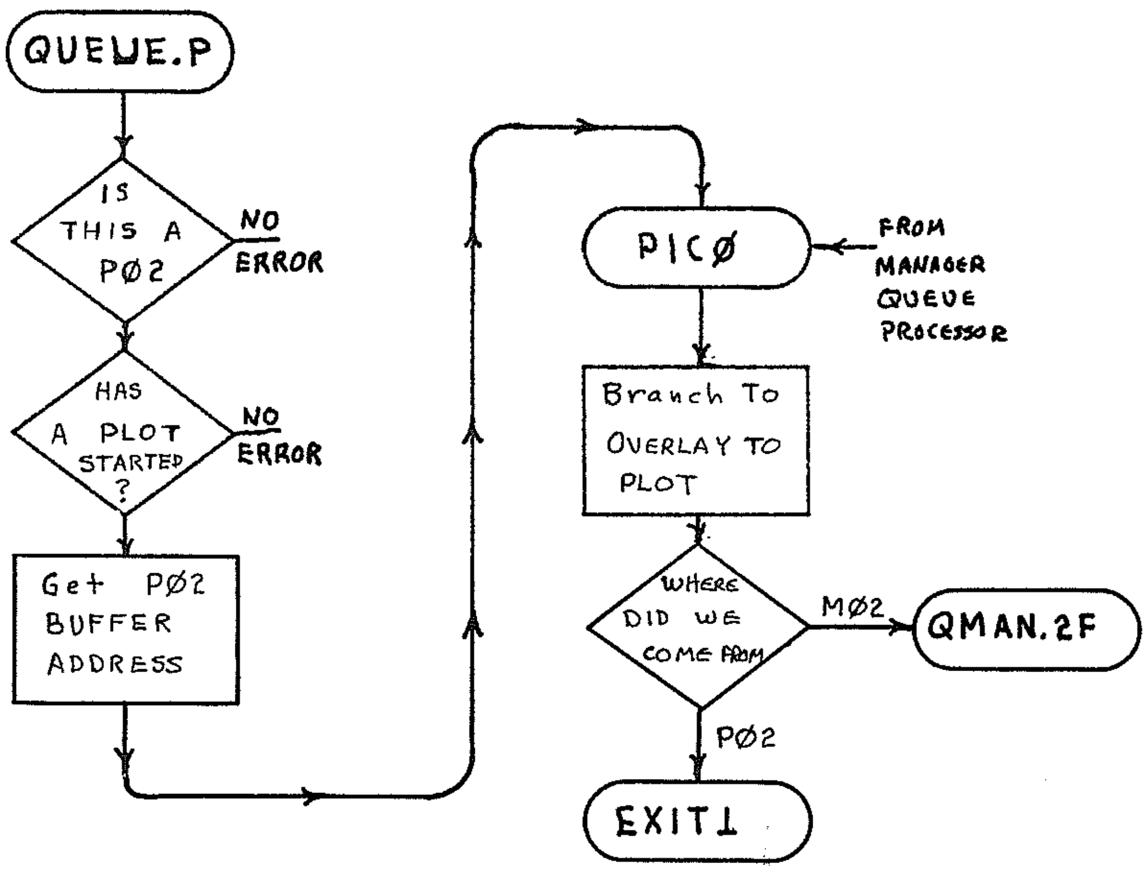


Figure 5.4. QUEUE.P

5.18. Figures

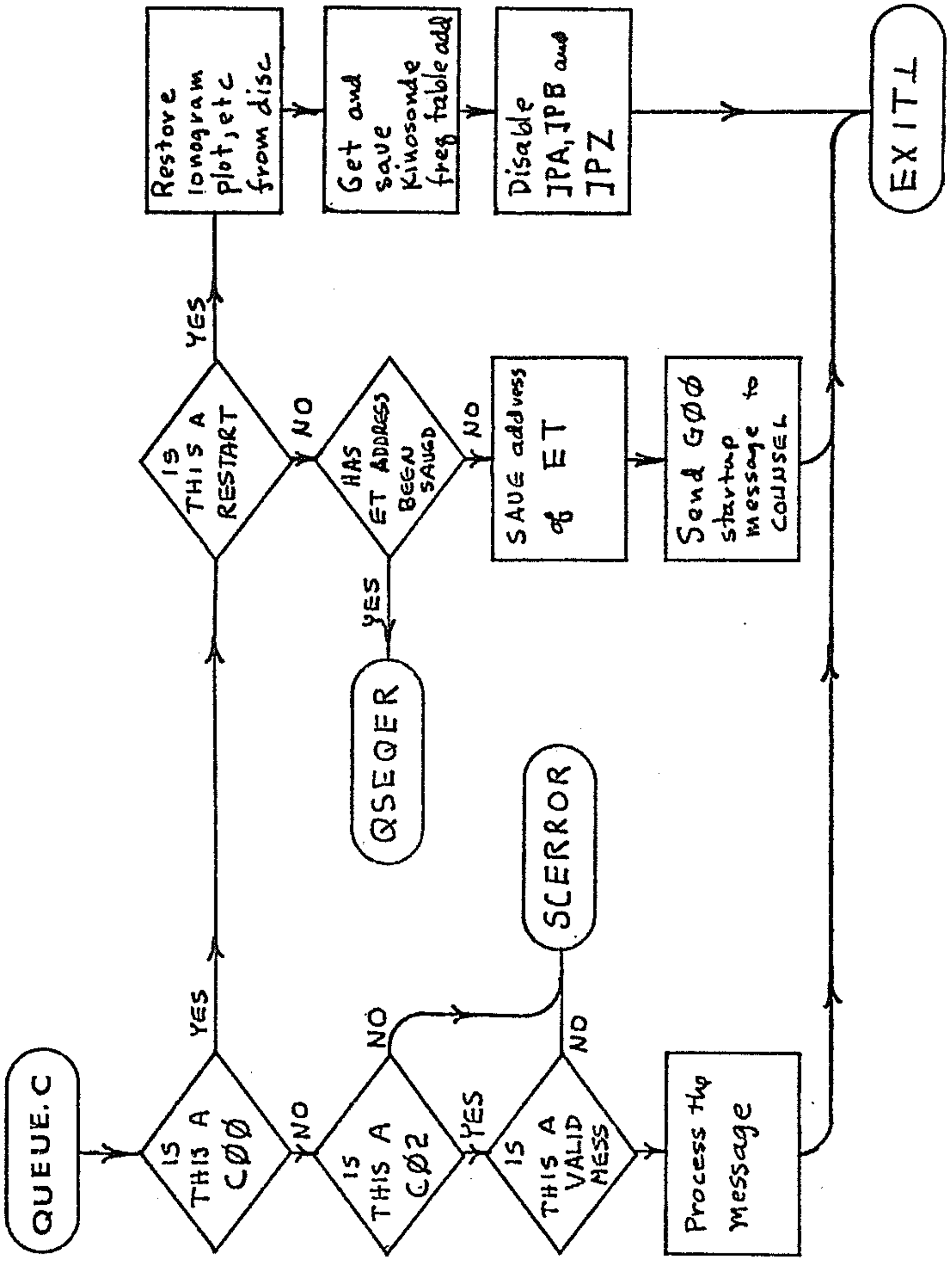


Figure 5.5. QUEUE.C

6. COUNSEL

James R. Winkelman

6.1. Introduction

Chapter 6 describes COUNSEL revision 2.Ø1, which logs in dated Ø7/28/83. For further details about changes made to COUNSEL 2.ØØ, refer to Appendix A, dated 1983 November.

COUNSEL serves five main purposes: (1) during start-up it loads and initiates execution of all other tasks; (2) it accepts all commands from the console device and sends them to the appropriate tasks; (3) it provides debugging commands that enable the user to examine and to alter memory; (4) it logs user task messages onto four devices, including the console; (5) it will ultimately monitor various hardware and software parameters.

COUNSEL is started with a file assigned to logical unit 1. This file is normally CON:. A command file from disc or TTY: may be assigned. This file is used for commands to configure the sounder. Various CSS files can be used to start the sounder and each file can have a configuration command file for a different study. This command file must end with the command CEND. If the RUN command appears in a command file, it must be followed immediately by CEND.

Control of the sounder software returns to the console (CON:) after the command file execution. The main function of COUNSEL is to interact with the sounder operator. This interaction is normally routing operator commands to the appropriate task, and logging commands and task messages.

The console is treated by the operating system as a half duplex device. Input and output cannot be overlapped. COUNSEL attempts to cancel a console read in progress, as needed to log a message, in a manner least disruptive to the operator.

COUNSEL executes a block of code, now empty, every ten seconds. It is planned to include tests on temperatures, voltages, and air-flows. The appropriate action for measurements out of range has not been formulated. These actions depend upon the presence of an operator, or upon the possibility of restart after some failures.

6.2. Start up

The partitions are set and the tasks are loaded before starting COUNSEL. Logical unit 1 should be assigned to the console or a command file.

COUNSEL starts the rest of the tasks in the order SOUNDER, GRAPHER, PICKER, MANAGER, and ANALYSER. A Q-block is sent to each task after starting and each task must send a 'STARTED' queue to COUNSEL before the next task is started.

The queue sent by COUNSEL is:

Ø	'ØØ'	
1	'C'	
2	status	unused
4	A(ET)	Environment Table address from SOUNDER

The 'STARTED' queue returned by the task is:

Ø	'ØØ'	
1	'*'	* is task initial S, G, P, M, or A
2	status	logging message pointer
4	A(SCAN)	task Scan Table location

The SCAN tables for all tasks except COUNSEL are referenced by ET.SCAN. The task must save the location of the ET table when the task is started (Sect. 7.3). COUNSEL waits to start each task until the STARTED queue has been received from the previous task. COUNSEL sends a 'SYS' command to SOUNDER when all tasks have been started.

Tasks may send logging messages to several devices by sending a LOG queue to COUNSEL. The message pointer replaces the status word. There is no hand-shaking for these queues, but COUNSEL is highest priority and should have no trouble keeping ahead of the tasks. Only one queue with a zero queue type should be sent during start-up. A task may use a zero queue type later.

Logging messages are sent to four output units, some of which can be NULL. The device assignments are made in the start-up CSS file.

Unit 2: A contiguous LOG file, four messages per sector. Sector zero is an index, messages are appended, and the file is circular. The contents of this file can be examined with CALL task LOG (Section 1Ø.2.21).

Unit 3: Any sequential file, e.g. LPR: or TTY:.
Unit 4: Any sequential file, 64 bytes per record.
Unit 5: CON: or an interactive command device. It is assumed that unit 5 is the same as unit 1, if unit 1 is interactive. When a message is to be written to unit 5, a HALTIO is given to unit 1, if unit 1 is waiting for an operator and not the OK response to a MEMORY command.

These four units are written asynchronously. Each logical unit is written as fast as possible, as long as there are messages to be written. The teletype can be several messages behind the line printer or console. The message buffer in memory has room for sixteen messages. The messages are written to unit 2 in sectors of four messages. There is no test for overlap when messages are entered into the buffer faster than they can be saved. A very slow device, which gets a full sixteen messages behind, will loose a block of sixteen messages.

The 64 byte logging message contains the date, time, identifier (such as P5) from the first word of the message queue, and a message of up to 43 bytes. The queue type byte may be either a true or ASCII number. Three message types are allowed. Byte 1 of the array pointed to, determines the format.

Byte 1<7 SVC 1 parameter block. Buffer start and end addresses are extracted from the parameter block.
Byte 1=7 SVC 2,7 LOG parameter block. Both direct and indirect text is permitted.
Byte 1>7 Direct text. Byte Ø contains the length and byte 1 is the first byte of the message.

6.3. Scan Table

Unless an Idle-mode CALL task (Section 1Ø.1) is running, all console commands are read by COUNSEL, which then searches the scan tables of the tasks in the order they were started. The scan tables are in the format required by the SVC 2,17 "Mnemonic Table Scan" described in the Programmer's Reference Manual OS/16 MT2.

6.3. Scan Table

The task whose scan table contains a match is sent a queue:

Ø	'ØØ'	
1	'C'	
2	status	-'1ØØØ'
4	COMMAND	
6	LINE	

Status is returned as zero if the command was valid and >Ø if in error. The value of the error is typed on the console.

COMMAND is the number of the item in the scan table; Ø is the first, 1 the second, 2 the third, etc.

LINE is the address of the next byte in the input buffer, immediately after the command word which was recognized by the scan. The queued task is responsible for any further scans or data conversions from the line (see SVC 2,17 and SVC 2,15 in the reference given above).

ET.SCAN contains the address of SCAN.TAB in COUNSEL. Table 6.1 describes the contents of the scan table that starts at SCAN.TAB.

SCAN.TAB	COUNSEL Scan Table
+2	SOUNDER Scan Table
+4	GRAPHER Scan Table
+6	PICKER Scan Table
+8	MANAGER Scan Table
+1Ø	ANALYSER Scan Table

Table 6.1. COUNSEL SCAN TABLE

During start up, only tasks appearing earlier in this table than the started task are known.

Table 6.1 defines both starting order and scan priority. If two tasks use the same name, the task which appears earlier in the above table is called.

6.4. Debug Mode

The debug mode allows the user to examine and alter memory. COUNSEL verifies addresses and prints the old and proposed new (if any) contents in hexadecimal and both the positive and negative decimal values.

In addition to the commands, COUNSEL defines values for four letters, W, X, Y, and Z, for the user's convenience; these are described with the commands and the format conventions below.

PAUSE, CEND, CANHF, and COMMAND are not debug commands but are listed here because they are commands to COUNSEL.

PAUSE — Pause for 30 seconds.

CEND — End of special COUNSEL command file. Open CON: as the input device and continue.

CANHF — Cancel all tasks. A C04 is sent to each of the other tasks, the logging buffer is flushed, all output is completed, and then COUNSEL cancels itself.

COMMAND fn — Open file fn as a command file. File fn normally defines a set of parameters, type formats, and the TSG and FEP programs. The file ends with CEND or another COMMAND. Files are not nested. The new one replaces the old. The file, fn, should include a file name extension to distinguish it from the name of an interactive device.

The debug commands described below also appear in Chapter 8. Any one of these commands invokes the debug mode. Underscored letters are mandatory.

Numeric values in fields required by commands may be either hexadecimal or decimal as shown under "Field format" below.

BIAS field — Define the bias, or base value, for displacements. The specified bias becomes the current value of W until changed by another BIAS command. If there are additional fields, they are ignored.

MEMORY field1[,field2]

Display the contents of the memory address in field1, and if field2 appears, replace the current contents with the value of field2 by following COUNSEL's question mark with OK or YES.

After an OK or YES, the results are shown as if MEMORY X had been typed. A NO or anything else simply types a COUNSEL prompt.

If more than two fields appear, COUNSEL uses the last two, which enables the user to make corrections before typing the carriage return that ends the command.

As soon as COUNSEL has displayed the values, X = field1, Y = field2 if present and X otherwise, and Z = the contents of address X. These may be used in subsequent fields as shown under Field format, below.

LINE field — Display 12 half-words, or 24 bytes, beginning with the address in the field.

After the command is executed, Y = X + 16, and Z = X + 24. These values are convenient steps for LINE Y or LINE Z.

Field format — The components of the fields required by the commands obey the following conventions:

NNNN	Hexadecimal number ending with the first non-hex character.
MMMM.	Decimal number which must end with a decimal point.
+	Add the next value to the field thus far.
-	Subtract the next value from the field thus far.
blank	Add or subtract according to the last sign read. A field starts as though the user had typed +Ø, so that a blank in the absence of a sign means +.
W	The value of BIAS.
X	The value of the last address used.

Y — The value of the previous data field, or X if there was no data field, except after LINE when Y = X + 16.

Z — The contents of the memory location addressed by the last address field, except after LINE when Z = X + 24.

Examples — The following includes comments. D means decimal and H means hex. COUNSEL uses > as its prompt character and the colon to identify its responses.

```
*COUNSEL>BI EØØØ
*COUNSEL: EØØØ:
Command acknowledged by display of value.
*COUNSEL>ME W 5A
Blank means + by default.
*COUNSEL: EØ5A:  2EØ   736 -648ØØ
          addr    H      D    -D
*COUNSEL>ME X / 1234
*COUNSEL: EØ5A:  2EØ   736 -648ØØ / 1234  466Ø -6Ø876 ?
          addr    H      D    -D  data H     D    -D
Replacement ignored:  neither OK nor YES follow.
*COUNSEL>ME 5 +2,Z
*COUNSEL:    7:    Ø     Ø -ØØØØØ /  2EØ   736 -648ØØ ?
Replacement specifically prevented.
*COUNSEL>NO
*COUNSEL>ME W5+2,Z
Scan of command cannot interpret W5.
*COUNSEL: SCAN ERROR
*COUNSEL>ME W 5 2,Z
*COUNSEL: EØØ7: EØØØ 57344 -Ø8192 /    Ø     Ø -ØØØØØ ?
*COUNSEL>NO
*COUNSEL>ME W 52,Z
*COUNSEL: EØ52:    Ø     Ø -ØØØØØ / EØØØ 57344 -Ø8192 ?
Execute replacement.
*COUNSEL>YES
*COUNSEL: EØ52: EØØØ 57344 -Ø8192
*COUNSEL>MEM Y,12345.-345.
Decimal replacement value.
*COUNSEL: EØ52: EØØØ 57344 -Ø8192 / 2EEØ 12ØØØ -53536 ?
*COUNSEL>Y
*COUNSEL: EØ52: 2EEØ 12ØØØ -53536
*COUNSEL>ME X/Z
*COUNSEL: EØ52: 2EEØ 12ØØØ -53536 / EØØØ 57344 -Ø8192 ?
*COUNSEL>OK
*COUNSEL: EØ52: EØØØ 57344 -Ø8192
```

6.5. Program Organization

COUNSEL starts at CON58100 through CON60200. This space is within the message buffers, but start-up is finished before any overlap can occur. The message pointer block is read, and the address of the next message block extracted. A fake ET address is used to start COUNSEL's message log-in. COUNSEL continues with the task wake-up code in CON11500 to CON14200. This code is replaced by the user input command buffer after all tasks are running.

CON14800 through CON22600 contain the main scheduling code. QUEUE is called for every interrupt to COUNSEL and determines the correct action to be taken. The choices are (1) time-out, (2) task-queue, (3) console, or (4) non-console device.

If the queue type is time-out, the status of unit 5 is checked. If unit 5 is busy, nothing needs to be done. If unit 5 is idle, the status of the command queue is checked. If the status is negative, nothing is to be done except to count seconds until time to check the environment for dangerous conditions. If the status is zero, a task has successfully completed a command, and a command read is done. If the status is positive, the value is reported in an ERR message to the operator.

If the queue is from a task, the queue block is passed to a message process which displays the message. If the queue is a non-console device, the message output device handler is called to continue outputting the messages.

If the queue is from the console, a word called PROMPT is tested. PROMPT is zero if an operator command is expected; SCANNER is called. PROMPT is positive for output to the console; the console is now free to accept a console read of an operator command. PROMPT is set to -2 for a memory change request while it is typing the old and new contents of memory. PROMPT is then set to -1 to wait for an OK or YES from the operator.

The code from CON21000 to CON21700 is reserved for testing for major problems with the sounder. CON23200 to CON24600 processes COMMAND file requests. Errors automatically revert to the console. Commands are not echoed unless the command device is interactive. CON24800 to CON25740 cancels all tasks, writes three blank messages (to flush the message buffer) and saves the current message sector address for later use.

SCANNER, from CON25900 to CON29200, scans the operator commands. When the first word of the command is found in the Scan Table of some task, that task is notified with a C02 queue, and the command is logged. If the command is not found, a SCAN ERROR message is typed. COUNSEL recognizes its own commands, but cannot accept C02 queues from another task.

The memory examine and modify commands are executed in lines CON30200 to CON44000. There is common code to convert the addresses, and some simple code to convert and print as requested.

Message processing starts at CON45400. The message buffer is cleared and a date and time inserted. The message bytes are located according to the format, and moved into the buffer by code from CON49000 to CON50200. The buffer lengths chosen, 64, 256 (4 times 64), and 1024 (16 times 64) were chosen so that tests for new block and wrap-around can be done with masking instructions. Code from CON50400 to CON51700 is concerned with start-up messages and operator commands requiring task queue calls.

Code from CON51900 to CON56800 starts output devices for new messages, does not start output when the device is already busy, and continues output when a device finishes the current line. The code from CON55000 to CON55900 checks for interactions between the console, unit 1, and unit 5, and halts unit 1 while outputting to unit 5.

7. TABLES AND Q-BLOCKS

David C. Walden

This chapter provides detailed descriptions of the global tables and the Q-blocks used by Product Two.

In the tabular descriptions of the SCT and of the other tables in the following sections, the displacement (DISPLACE) is the symbolic word location used in the HF Radar software to address each entry in the table. The decimal or hexadecimal offset under heading OFF is the numerical value assigned to the symbolic displacement by the assembler. The decimal value under BYTE is the number of bytes in a given field. In Section 7.3, the displacements are in alphabetic order instead of their order in the Interdata computer. The descriptions include references to task descriptions that discuss given entries.

Most preset values in the SCT are set by task MAKEM (Section 1Ø.2.1), other preset values come from code assembly and computation during SOUNDER start-up. Assembly code parameters used to preset table entries, table lengths, etc., can be integers, text, or expressions involving other pre-defined values. Groups of pre-defined values are made available at assembly time by invoking the COPY directive to the Assembler from the assembly source code. The argument of the COPY directive must be the label of a parameter group in file SYSC:PCB.CAL. (See the CAL User's Manual.) The HF Radar software uses this facility to define the structures and lengths of the tables described in this section.

All commands referred to below appear in alphabetic order in Chapter 8.

7.1. Sounding Configuration Table (SCT)

The Sounding Configuration Table (SCT) contains data relating to the sounding as a whole, as opposed to a particular pulse or set of pulses. It may be thought of as being divided into three logical parts. The first part contains information such as the time, sounding number, etc., that are kept and updated relatively often by the computer. Except for the sounding mode, none are mode dependent. These start with SCT.LEN and end with SCT.ETS. A second block of constants runs from SCT.GLAT through SCT.XTRA. These include such things as the site information, receiver channel offsets, task revisions and schedule extension. These are all items that are relatively fixed and independent of the sounding mode. The third part is all sounding mode dependent, starting at SCT.PD and ending at SCT.FTBL. The description below is in order of location, followed by an alphabetized list with offsets.

DISPLACE	BYTE	OFF	DESCRIPTION
SCT.LEN	2	Ø	Length of SCT in bytes, preset, presently hex 2ØØ. The sign bit is set to distinguish it from a pulse set record, making it hex 82ØØ.
SCT.MODE	1	2	Sounding mode: Ø I-mode, 1 K-mode, 2 V-mode, 3 B-mode, 4 P-mode, 5 G-mode, 6 Z-mode, 7 W-mode. Set by the MGET id or RUN id commands.
SCT.SPH	1	3	Soundings per hour. Set by the SPH command.
SCT.SN	2	4	Sound n times, then stop. Set by the SN command.
SCT.SNO	2	6	Sounding number started at 1 by system restart, or by SN.
SCT.STS	4	8	Sounding start time in seconds from 1/1/77.
SCT.STM	2	C	Sounding start time, milliseconds part, currently always Ø.
SCT.ETS	4	E	Sounding end time in seconds from 1/1/77.
SCT.GLAT	4	12	Geographic latitude, N is plus, S is negative. The value is a 32-bit, 2's complement number with the binary point between the two 16-bit 1/2-words.
SCT.GLON	4	16	Geographic longitude, E is plus, W is negative. The value consists of the 16 most significant bits being a 2's complement integer, with the sixteen least significant bits being the fractional part as in SCT.GLAT.
SCT.SITE	4	1A	Site code, four ASCII characters.
SCT.LTIM	2	1E	Local time meridian in tenths of hours, positive East (Boulder is -7Ø)
SCT.MLAT	4	2Ø	Geomagnetic latitude, scaling same as SCT.GLAT.
SCT.MLON	4	24	Geomagnetic longitude, scaling same as SCT.GLON, but always positive.
SCT.MDIP	4	28	Magnetic dip, with the scaling being the same as SCT.GLAT. Most of the northern hemisphere is plus, with Boulder being 67.8 N.
SCT.GYRO	4	2C	Gyrofrequency in MHz at 2ØØ kilometers. The binary point is betwen the two sixteen bit words. Boulder is 1.43 MHz.
SCT.TOTF	4	3Ø	Total magnetic field in Øersteds, again sixteen bits fractional part. Boulder is Ø.56

DISPLACE	BYTE	OFF	DESCRIPTION
			Øersteds.
SCT.USER	28	34	User defined values, entered in hex by the LOCAL task (Section 1Ø.2.2).
SCT.SFRQ	4	5Ø	Minimum frequency in Hertz/1ØØ. It was planned that this would be the actual minimum frequency, but currently it is the same as MD.SFRQ.
SCT.EFRQ	4	54	Maximum sounding frequency in Hertz/1ØØ.
SCT.1XØ	2	58	Receiver channel one X DC offset, from the FEP. This value is obtained experimentally from the FEP at the start of each sounding.
SCT.1YØ	2	5A	Receiver channel one Y DC offset.
SCT.2XØ	2	5C	Receiver channel two X DC offset.
SCT.2YØ	2	5E	Receiver channel two Y DC offset.
SCT.REVC	2	6Ø	COUNSEL task revision, set by COUNSEL.
SCT.REVS	2	62	SOUNDER task revision, set by SOUNDER.
SCT.REVP	2	64	PICKER task revision, set by PICKER.
SCT.REVM	2	66	MANAGER task revision, set by MANAGER.
SCT.REVG	2	68	GRAPHER task revision, set by GRAPHER.
SCT.REVA	2	6A	ANALYSER, Scheduler task revision.
SCT.MREV	2	6C	Sounder tables mode revision.
SCT.FGA	1	6E	Frequency generation algorithm index.
SCT.DGA	1	6F	DIO generation algorithm index.
SCT.SKDX	3	7Ø	Schedule extension or Øxx, set by the SRUN or SAB commands. If the most significant byte is a hex 'ØØ', then no schedule was running at the time of the sounding.
SCT.XTRA	15	73	Room to grow.
SCT.PD	8Ø	82	Pulse palette section. There are 8 pulse descriptions, each of which is 1Ø bytes in length. MD.NPD is the number of descriptions (default 8), and MD.PDSZ is the 1Ø bytes per PD segemnt. See section 7.5.
SCT.PSEQ	32	D2	Pulse set sequence table, set by the PSEQ command on a mode basis. The table is organized as 32 one-byte entries. The first entries, working from the start towards the middle of the table, contain pointers into the SCT.PD table. The SCT.PD pointer is in the low order 5 bits (mask 1F) of the byte; the high order 3 bits may contain flags. The last entries, working from the end towards the middle of the table, relate to repeated pulse sequences. A pointer of zero points to the first PD in the table, one to the second

DISPLACE	BYTE	OFF	DESCRIPTION
			PD, etc. The sequence repeats are also one less than the number of actual repeats. The pointers may be flagged with control information that is ORed with the actual pointer. Three flags exist and may all be ORed together with the pointer portion of one byte. A flag of hex 2Ø signals that there are no more pointers beyond this one. A flag of 8Ø signals the first of a repeat sequence and indicates an associated repeat count. A 4Ø signals the last of a repeat sequence and the end of the use of the current repeat count. A short sequence would be generated by a command of PSEQ 4(1), which would put one byte, EØ, at the start of the PSEQ table and a Ø3 at the end. The normal PSEQ 1,2,3,4 would put the four bytes Ø, 1, 2 and 23 into the start of the PSEQ table. A more complicated command, PSEQ 4,3(6,2),1,5(3),6,4 yields a PSEQ table of 3, 85, 41, Ø, C2, 5, 23, ..., 4, 2.
MD.MREV	2	F2	Mode revision, which must match SCT.MREV when mode image is read from disc. If MD.MREV = SCT.MREV, MD.MREV is set to the ASCII mode identifier, or name.
MD.DESC	8	F4	Sounding description is eight ASCII characters and is set by the DESC command on a mode basis.
MD.CALL	8	FC	Call task name is eight characters or blanks. Set by the CALL command on a mode basis.
MD.PIK	8	1Ø4	Picker task name. Set by the PIK command on a mode basis.
MD.TMDE	1	1ØC	Sounding mode, copied into SCT.MODE.
MD.CTRL	1	1ØD	Sounding control byte. Bits (Ø = most significant bit) Ø-1 Sounding class (set by MAKEM, implies MAP) ØØ = normal Sweep-class (S-class) Ø1 = pseudo-random sweep (not implemented) 1Ø = normal Table-class (T-class) 11 = extended T-class (listen only, not implemented) 2 W-class bit Ø = not W-class 1 = W-class sounding (set by MAKEM) 3-4 Open

DISPLACE	BYTE	OFF	DESCRIPTION
			5-6 Ramp description (S-class or T-class) ØØ = B-type log ramp (RAMP n,m,s,B) 1Ø = Z-type log ramp (RAMP n,m,s,Z) Ø1 = B-type linear ramp (RAMP n,m,Fs,B) 11 = Z-type linear ramp (RAMP n,m,Fs,Z) 7 Sweep step (S-class only) Ø = log (frequency index) (STEP n) 1 = linear (frequency in Hz/1ØØ) (STEP Fn)
MD.PPS	1	1ØE	Pulses per pulse set, normally 4. Derived from the PSEQ command. MD.PPS is actually the number of starts issued by SOUNDER to the TSG for each pulse set of a sounding. Complementary coding may yield a situation where PPS is 8, OPOB is 42, and HPE is 17.
MD.NTFR	1	1ØF	Number of active frequencies in the frequency table. Set by combinations of Fn F commands on a mode basis.
MD.OPOB	2	11Ø	Offset to range coincident data in buffer with the buffer address in the PØ2 Q-block. (See Sections 3.3 and 3.7.) Typically, the value is a decimal 42, or hex 2A. MD.OPOB is set from MD.PPS (implied by the PSEQ command), but for complementary coding PICKER sets MD.OPOB to its MAX.OPOB (Section 3.1).
MD.SFRQ	4	112	Starting frequency for this mode and sounding in Hertz/1ØØ. Set by the SFRQ command.
MD.EFRQ	4	116	Ending frequency for this mode and sounding in Hertz/1ØØ. Set by the command EFRQ.
MD.SFIX	2	11A	Frequency index corresponding to MD.EFRQ, the ending frequency.
MD.EFIX	2	11C	Frequency index corresponding to MD.SFRQ, the starting frequency.
MD.STEP	2	11E	Size of base frequency step. Set by the STEP command. It is either frequency index steps or Hertz/1ØØ steps as specified by bit 7 in MD.CTRL. It is a positive 16-bit integer. A frequency step must be less than 3.275 MHz. Frequency index steps are 1-3275.
MD.SUBN	2	12Ø	Number of steps in the ramp, set by the RAMP command.
MD.SUBS	2	122	Size of the ramp frequency step, set by the RAMP command. It is like MD.STEP.
MD.NFIL	2	124	Number of TSG NOP fills betwen pulse sets. These are in the units of the basic pulse rate, usually 1Ø, 2Ø, or 1ØØ ms. Set by the

DISPLACE	BYTE	OFF	DESCRIPTION
			command NFIL.
MD.NRAM	2	126	Number of ramps on the base frequency. Set by the RAMP command.
MD.NSET	2	128	Number of passes through the base frequency selections. Set by the NSET command.
MD.DELF	2	12A	Delta frequency in Hertz/1ØØ, set by the command DELF. It is a positive 16-bit integer and must be less than 3.275 MHz.
MD.PCT	1	12C	Number of bytes in the Pulse Configuration Table (PCT). Typically, the value is a decimal 1Ø or hex 'A'.
MD.HPE	1	12D	Number of 16-bit halfwords per echo, usually 17. Set implicitly by the PSEQ command, but for complementary coding set explicitly by the HPE command.
MD.EPF	1	12E	Maximum number of echoes per frequency, usually between 8 and 52. Set by the EPF command.
MD.M	1	12F	Selection for coincidence processing by range. Only those returns with M echoes out of N possible will be kept. Set by the KEEP command.
MD.N	1	13Ø	Number of possible echoes for coincidence processing. It is MD.PPS*2, usually 8. It is derived from the PSEQ command.
MD.FEPT	1	131	FEP thresholding level, set by the FEP command.
MD.FEPX	3	132	FEPMEM extension, the source of the program actually running in the FEP. The value is three ASCII characters set by the command BOOT FEP[,xxx]. The default extension is "TWO".
MD.TSGX	3	135	TSGMEM extension, the source of the program actually running in the TSG. These three ASCII characters are set by the BOOT TSG[,xxx] command. The default extension is "TWO".
MD.DIOX	3	138	DIO file extension, three ASCII characters set by the BOOT DIO[,xxx] command. The default extension is "TWO".
MD.NPD	1	13B	Number of pulse palette (PD) segments in SCT.PD, currently 8. MD.NPD is set from NPD in the SCT STRUC in PCB.CAL. NPD can be set from 1-32, depending on memory available. NPD also controls the size of the DIO.PTP pulse palette extensions (Section 7.9). See also DIOGEN, Section 1Ø.2.4.

DISPLACE	BYTE	OFF	DESCRIPTION
MD.PDSZ	1	13C	Number of bytes in a PD segment, currently decimal 1Ø.
MD.RXAN	1	13D	Receiver antenna field number. Set by the RXANT command. This is the sector number of SYS1:LOCAL.TWO that contains the receiver antenna geometry for this sounding mode.
MD.ECHO	1	13E	Output data merit ratio set by task PICKER.
MD.AGC	1	13F	AGC control byte, set by the AGC command. The byte is in the range of Ø to 3. Ø Both channels under manual gain control. 1 Channel 1 auto, channel 2 manual. 2 Channel 2 auto, channel 1 manual. 3 Both receivers under automatic gain control.
MD.PKLO	4	14Ø	Lower receiver dynamic range limit for AGC in millivolts. Set by the PKLO command by receiver channel. 1Ø volts is full scale. There are two bytes per channel, in channel 1, 2 order.
MD.PKHI	4	144	Upper receiver dynamic range limit for AGC in millivolts. Set by the PKHI command by receiver channel. There are two bytes per channel, in channel 1, 2 order.
MD.ADLY	2	148	Zero range correction for the antennas, receivers in microseconds (analog delay). Set by MAKEM.
MD.ELST	2	14A	Lost echo count.
MD.PLST	2	14C	Lost pulse count.
MD.STAT	2	14E	General HF radar status. The hex mask for each bit is given. Bit Ø 8ØØØ (most significant bit): Ø = high-power transmitter high voltage off. 1 = high-power transmitter high voltage on. Bit 1 4ØØØ Dummy load status. Ø = Dummy load on. 1 = Dummy load off. Bit 2 2ØØØ Synthesizer status. Ø = Local. 1 = Remote. Bit 3 1ØØØ TSG errors. Ø = No TSG errors. 1 = Loading errors, TSG timeout, or TSG self-test failure.

7.1. Sounding Configuration Table (SCT)

DISPLACE	BYTE	OFF	DESCRIPTION
			Bit 4 800 FEP errors. 0 = No FEP errors. 1 = FEP timeout (if FEP driver used). Bit 5 400 EKO errors. 0 = No EKO errors. 1 = EKO timeout. Bit 6 200 DIO errors. 0 = No DIO errors. 1 = DIO interface state error or verify error. Bit 7 100 TSG clock errors. 0 = No TSG clock errors. 1 = TSG clock read error. Bit 8 80 HKADC errors. 0 = No HKADC errors. 1 = HKADC timeout. Bits 9 - 15: to be determined.
MD.TXLP	1	150	Low power transmitter attenuation, in dB. Set by the LPAT command.
MD.TXHP	1	151	High power transmitter attenuation, in dB. Set by the HPAT command.
MD.BDW	1	152	Bandwidth commands, filters flag, set by the BDW command. The bandwidths are in the 4 MSB, 2 bits per channel, in channel 1 (hex mask C0), 2 (hex mask 30) order. 0 = 1 kHz, 1 = 3 kHz, 2 = 10 kHz, and 3 = 30 kHz. The 4 MSB are followed by one bit (mask 8) that is set if the wide band filters are selected. The 3 LSB are unused.
MD.NHSK	1	153	Number of HKADC multiplexer channels to read per sounding pulse.
MD.CSTP	8	154	Calibration attenuation step frequency indexes (4). Set by the CALA command.
MD.CALA	1	15C	Starting calibration attenuation, set by the CALA command.
MD.CDN	1	15D	Sounding countdown in seconds, set by the command CDN.
MD.MAP	1	15E	Protected frequencies map identifier, one ASCII character, usually "I" or "K". Set by MAKEM.
MD.SPIX	1	15F	Sounding initialization routine index or zero. (Not fully implemented.).
MD.PPU	2	160	Pulses per AGC update, set by the AGC command. There is one byte per channel, in channel 1, 2 order. (Not fully implemented.)

DISPLACE	BYTE	OFF	DESCRIPTION
MD.BAG	4	162	AGC base settings in dB, set by the AGC command. The order is RF1, IF1, RF2, IF2.
MD.FEPP	12	166	FEP parameters (locations 2 through 4). Set by the FEP command.
MD.EKOF	2	172	Echoes found by PICKER.
MD.EKOS	2	174	Echoes saved by PICKER.
MD.REPS	4	176	FEP averaging repeat counts from TSG zero page.
MD.TXAN	2	17A	TX antenna switching frequency (S-class), or TX antenna (T- or W-class). (Not fully implemented.)
MD.XTRA	8	17C	Room to grow in mode tables.
SCT.FTBL	124	184	Frequency table for T- and W-class modes. See section 7.4. There are ten entries in SCT.FTBL, each of which has twelve bytes, plus four bytes for the table itself. Entries are changed by the F n command, the RNGE n command, and by the RFAT and IFAT commands.

The following table lists the above variables in alphabetic order. All of the MD. variables come before the SCT. variables. Each entry consists of a name, the length in bytes, and the off-set in the SCT expressed as a hex number of bytes.

NAME	BYTE	OFFSET	NAME	BYTE	OFFSET
MD.ADLY	2	148	MD.STEP	2	11E
MD.AGC	1	13F	MD.SUBN	2	12Ø
MD.BAG	4	162	MD.SUBS	2	122
MD.BDW	1	152	MD.TMDE	1	1ØC
MD.CALA	1	15C	MD.TSGX	3	135
MD.CALL	8	FC	MD.TXAN	2	17A
MD.CDN	1	15D	MD.TXHP	1	151
MD.CSTP	8	154	MD.TXLP	1	15Ø
MD.CTRL	1	1ØD	MD.XTRA	8	17C
MD.DELF	2	12A	SCT.DGA	1	6F
MD.DESC	8	F4	SCT.EFRQ	4	54
MD.DIOX	3	138	SCT.ETS	4	E
MD.ECHO	1	13E	SCT.FTBL	124	184
MD.EFIX	2	11C	SCT.FGA	1	6E
MD.EFRQ	4	116	SCT.GLAT	4	12
MD.EKOF	2	172	SCT.GLON	4	16
MD.EKOS	2	174	SCT.GYRO	4	2C
MD.ELST	2	14A	SCT.LEN	2	Ø
MD.EPF	1	12E	SCT.LTIM	2	1E

7.1. Sounding Configuration Table (SCT)

NAME	BYTE	OFFSET	NAME	BYTE	OFFSET
MD.FEPP	12	166	SCT.MDIP	4	28
MD.FEPT	1	131	SCT.MLAT	4	2Ø
MD.FEPX	3	132	SCT.MLON	4	24
MD.HPE	1	12D	SCT.MODE	1	2
MD.M	1	12F	SCT.MREV	2	6C
MD.MAP	1	15D	SCT.PD	8Ø	82
MD.MREV	2	F2	SCT.PSEQ	32	D2
MD.N	1	13Ø	SCT.REVA	2	6A
MD.NFIL	2	124	SCT.REVC	2	6Ø
MD.NHSK	1	153	SCT.REVG	2	68
MD.NPD	1	13B	SCT.REVM	2	66
MD.NRAM	2	126	SCT.REVP	2	64
MD.NSET	2	128	SCT.REVS	2	62
MD.NTFR	1	1ØF	SCT.SFRQ	4	5Ø
MD.OPOB	2	11Ø	SCT.SITE	4	1A
MD.PCT	1	12C	SCT.SKDX	3	7Ø
MD.PDSZ	1	13C	SCT.SN	2	4
MD.PIK	8	1Ø4	SCT.SNO	2	6
MD.PKHI	4	144	SCT.SPH	1	3
MD.PKLO	4	14Ø	SCT.STM	2	C
MD.PLST	2	14C	SCT.STS	4	8
MD.PPS	1	1ØE	SCT.TOTF	4	3Ø
MD.PPU	2	16Ø	SCT.USER	28	34
MD.REPS	4	176	SCT.XTRA	15	73
MD.RXAN	1	13D	SCT.1XØ	2	58
MD.SFIX	2	11A	SCT.1YØ	2	5A
MD.SFRQ	4	112	SCT.2XØ	2	5C
MD.SPIX	1	15F	SCT.2YØ	2	5E
MD.STAT	2	14E			

7.2. Pulse Configuration Table (PCT)

There is a Pulse Configuration Table associated with each normal data producing pulse transmitted. The typical PØ2 block has four PCTs after the length word, one for each pulse transmitted in the pulse set. PICKER discards PCTs if the word count of the associated EKO buffer is less than 14. This is the case for complementary coded soundings and for soundings that employ TSG NOPs embedded within the pulse sets.

DISPLACE-MENT	BYTES	BITS	BIT COUNT	DESCRIPTION
PCT.TIK	2	Ø	1	1 = first pulse of pulse set, otherwise Ø. This bit, normally only used internally by SOUNDER and PICKER, will not be set in data recorded for complementary coded soundings, or for soundings that employ a TSG NOP for the first pulse of the pulse set.
		1-15	15	Ten millisecond interval count, cycles every five minutes.
PCT.FRQ1	2	Ø	1	1 = frequency protected, otherwise Ø.
		1-3	3	High order receiver frequency.
		4-15	12	Frequency index as in MD.SFIX.
PCT.FRQ2	2	Ø-15	16	Low order receiver frequency. This plus the 3 high order bits from FRQ1 gives Hertz/1ØØ.
PCT.RFIF	2	Ø-3	4	RF attenuation command, channel 1.
		4-7	4	RF attenuation command, channel 2.
		8-11	4	IF attenuation command, channel 1.
		12-15	4	IF attenuation command, channel 2. To obtain the attenuation from any of the above four commands, invert the command by subtracting it from 15 decimal or taking the XOR with F hex, and multiply by 4. These attenuations lie in the range Ø through 6Ø dB in 4-dB steps.
PCT.CALA	1	Ø-3	4	Open.
		4-7	4	Calibration attenuation command. To obtain the attenuation, invert the command by subtracting it from 15 decimal or taking the XOR with F hex, and multiply by 8. The attenuation ranges from Ø through 12Ø dB in 8-dB steps and may be set by CALA.

7.2. Pulse Configuration Table (PCT)

DISPLACE-MENT	BYTES	BITS	BIT COUNT	DESCRIPTION
PCT.PDIX	1	Ø-3	4	Open.
		4-7	4	Palette index, indexes into SCT.PD and DIO.PTP (Section 7.9). This index, stripped of any flags, ranges from Ø to MD.NPD-1 and comes from the pointer portion of SCT.PSEQ.

7.3. Environment Table (ET)

The six routines with addresses in ET are in SOUNDER (Section 2.6.3) but are available to other tasks including user-written ANALYSER or CALL tasks. CALL tasks were formerly TEST tasks in Product 1.

DISPLACEMENT	BYTES	DESCRIPTION
ET.ABUF	2	Address of shared EKO buffers area. Between soundings, the EKO buffers area is available to other tasks, including user-written versions of ANALYSER. The length of the shared area is in ET.LBUF, below.
ET.ADIO	2	Address of DIO.PTP - 2. DIO.PTP contains the pulse palette extensions (Section 7.9).
ET.AFRTB	2	A(FTOBCD), the address of routine FTOBCD to convert hex frequency into BCD frequency.
ET.AFRTN	2	A(FRTON), the address of routine FRTON to convert frequency into frequency index.
ET.AHKD	2	Address of HKADC/DIO table - 2.
ET.ALOGF	2	A(LOG), the address of routine LOG to compute the common or natural log.
ET.ANTFR	2	A(NTOFR), the address of routine NTOFR to convert frequency index into frequency.
ET.APAMP	2	A(PHAMP), the address of routine PHAMP to compute phase and amplitude.
ET.ASCT	2	Address of SCT.
ET.AXORO	2	Address of SOUNDER subroutine to discriminate between the Ordinary and eXtraordinary returns.
ET.CDN	2	Countdown value, ≥Ø or -1, if not counting down.
ET.HKADC	128	Table of 64 HKADC readings.

DISPLACEMENT	BYTES	DESCRIPTION
ET.LBUF	2	Length of shared EKO buffers area with address in ET.ABUF, above.
ET.SCAN	2	Address of SCAN list for sending commands under program control. See Section 6.3.
ET.SEC	4	Time-of-day in seconds past 1/1/77.
ET.TNXT	2	Time to next sounding (seconds).
ET.VOLN	4	Volume name for GRAPHER restart.

7.4. Frequency Table Description (SCT.FTBL)

This section describes sub-table SCT.FTBL of the SCT, the T- and W-class frequency table. SCT.FTBL entries are modified by commands F n, RNGE n, IFAT n, and RFAT n for T- and W-class sounding modes. Only command RNGE n is accepted for Sweep-class modes, and is used to set up the RANGE command for PICKER. There are currently 1Ø entries in SCT.FTBL numbered Ø through 9. The SCT.FTBL entry number is always the first argument of the commands mentioned above.

DISPLACEMENT	BYTES	DESCRIPTION
+Ø	2	Number of entries in table (1Ø).
+2	12	T- or W-class frequency Ø entry.
+14	12	1st entry.
+26	12	2nd.
+38	12	3rd.
+5Ø	12	4th.
+62	12	5th.
+74	12	6th.
+86	12	7th.
+98	12	8th.
+11Ø	12	9th.
+112	2	End-of-table flag (8ØØØ).

Each entry of 12 bytes contains the following:

DISPLACE	BYTE	OFF	DESCRIPTION
FR.FREQ	4	Ø	Frequency set by command F, or -1,Ø if not specified, or reset. The frequency is in Hertz/1ØØ.
FR.RMIN	2	4	Minimum range set by command RNGE. Default is Ø km (Ø microseconds).
FR.RMAX	2	6	Maximum range set by command RNGE. Default is

7.4. Frequency Table Description (SCT.FTBL)

DISPLACE	BYTE	OFF	DESCRIPTION
			5ØØ km (3333 microseconds) for T-class modes, and 1ØØØ km (6666 microseconds) for W-class modes.
FR.ATTN	4	8	RF/IF attenuations in dB (2 channels). The order is RF1, IF1, RF2 and IF2.

Ranges are set and displayed in kilometers, but are stored in microseconds.

7.5. The Pulse Palette (SCT.PD)

This section describes sub-table SCT.PD of the SCT, the pulse palette. There are presently eight Pulse Descriptions (PD) in the pulse palette. Each of these is one PD of ten bytes. The PD consists of attributes of a pulse that do not change from pulse set to pulse set during a sounding. Normally, four of these PD are selected in any order by the PSEQ command to define the pulse set for the sounding.

DISPLACE	BYTE	OFF	DESCRIPTION
PD.TDLY	2	Ø	TSG delay in microseconds.
PD.TSGC	1	2	TSG command byte, typically 86 hex.
PD.FEPC	1	3	FEP command byte, usually 52 hex for 1Ø microsecond sampling, occasionally 12 for 2Ø microsecond sampling.
PD.FEPA	1	4	FEP starting address, typically 13 hex.
PD.DFAC	1	5	Delta F factor, -128 to +128 in 2's complement form.
PD.TXPH	1	6	TX, TX phase control, not fully implemented.
PD.SPIX	1	7	Special routine index or Ø, not implemented.
PD.1RIN	1	8	Channel 1 receiver input, in DIO command form, one less than antenna number, Ø-3.
PD.2RIN	1	9	Channel 2 receiver input, Ø-3.

7.6. Local Site Parameters (LOCAL)

The HF Radar parameters that are site dependent are specified by the user from the console to CALL task LOCAL (Section 1Ø.2.2), which saves them on disc file SYS1:LOCAL.TWO. This file is contiguous and contains 15 256-byte records, 11 of which are currently used. Record Ø of the LOCAL file contains the local site parameters that are copied to the SCT (Section 7.1) at SCT.GLAT through SCT.USER at SOUNDER start up. The structure of this section of the SCT is identical to that of the LOC STRUC, below. CALL task LOCAL should be run stand-alone (Section 1Ø.2) before sounding at a new field site in order to establish the local site parameters.

Records 1-1Ø of the LOCAL disc file contain descriptions of up to 1Ø receiving antenna arrays, one array description per 256-byte record. A single array can contain up to 12 individual antennas, each of which is defined in order according to the RXA STRUC (below) from RXA.CHAN through RXA.GAIN. RXA.NANT and RXA.ADLY appear only once at the start of each 256-byte record, or array description. SOUNDER command RXANT is used to associate an array description with a given sounding mode. The default RXANT is 1.

The entire LOCAL file is copied to tape by MANAGER as a result of the TAPE command. The local site parameters are in the first 256 bytes of the LOCAL tape record (flagged with FØØØ hex); the value in MD.RXAN, from the RXANT command, is used to find the 256-byte array description block following the local site values.

The layout of LOCAL record Ø follows. See SCT.GLAT through SCT.USER (Section 7.1) for data formats. Also see CALL task LOCAL, Section 1Ø.2.2. Both the LOC and RXA STRUCs (below) are in PCB.CAL and are invoked with the COPY LOCAL directive to the assembler. The byte offsets (OFF) from the beginning of a record are displayed below in decimal.

DISPLACE	BYTE	OFF	DESCRIPTION
LOC.LEN	2	Ø	Size of LOCAL file or tape record in bytes (low order 12 bits) with flag (high order 4 bits), currently hex FFØØ.
LOC.GLAT	4	2	Geographic latitude.
LOC.GLON	4	6	Geographic longitude.
LOC.SITE	4	1Ø	Local site code.
LOC.LTIM	2	14	Local time meridian (hours/1Ø).
LOC.MLAT	4	16	Geomagnetic latitude.

7.6. Local Site Parameters (LOCAL)

DISPLACE	BYTE	OFF	DESCRIPTION
LOC.MLON	4	20	Geomagnetic longitude.
LOC.MDIP	4	24	Magnetic dip.
LOC.GYRO	4	28	Gyrofrequency (MHz at 200 km).
LOC.TOTF	4	32	Total field in Øersteds.
LOC.USER	28	36	User-defined (14 16-bit hex values).

Each receiver antenna array record (1-10) of the LOCAL file is structured as follows.

DISPLACE	BYTE	OFF	DESCRIPTION
RXA.NANT	2	0	Number of antennas in array (1-12).
RXA.ADLY	2	2	Analog delay of array in microseconds.
RXA.CHAN	1	4	Receiver channel (1 or 2) to which antenna 1 is connected.
RXA.PORT	1	5	Receiver input port (1-4) used by antenna 1.
RXA.X	4	6	X coordinate of antenna 1 in signed meters.
RXA.Y	4	10	Y coordinate.
RXA.Z	4	14	Z coordinate.
RXA.O	4	18	Antenna 1 orientation in degrees.
RXA.GAIN	2	22	Antenna 1 gain in dB.
	20	24	RXA.CHAN-RXA.GAIN for antenna 2.
	20	44	Antenna 3 description.
	20	64	Antenna 4.
	20	84	Antenna 5.
	20	104	Antenna 6.
	20	124	Antenna 7.
	20	144	Antenna 8.
	20	164	Antenna 9.
	20	184	Antenna 10.
	20	204	Antenna 11.
	20	224	Antenna 12.

For a given array record, there will be RXA.NANT antenna descriptions (RXA.CHAN-RXA.GAIN). The X, Y, and Z coordinates are 32-bit, 2's complement values with the binary point between the two 16-bit 1/2-words. The antenna orientation, like SCT.MLON, is a 32-bit value that is always positive. RXA.O is specified to CALL task LOCAL in degrees, minutes, and seconds, but is stored as a 32-bit integer with fraction, as are the coordinates.

7.7. SOUNDER and CALL Task Parameters (SNDR)

To make them universally available at assembly time, commonly used parameters are included in file SYSC:PCB.CAL at block label **SNDR. All of SOUNDER's modules and a number of the CALL tasks are assembled with the COPY SNDR directive in their code. SNDR may be modified, but changes must be made circumspectly.

In SNDR, labels are equated to numerical values with the EQU directive. They are listed below and grouped as they are in SNDR.

SOUNDER pipeline register definitions: (See Section 2.9.1.1.)

NAME	VALUE	DESCRIPTION
GO	4	(R4) Sounding pipeline condition.
NPIPE	5	(R5) Number of pulses in pipeline.
PULSE	6	(R6) TSG starts from sounding start.
S.	7	(R7) Address of the SCT.
E.	8	(R8) Address of the ET.
D.	9	(R9) Address of the DIO table.
P.TSG	1Ø	(R1Ø) TSG start description.
P.FEP	11	(R11) Descritpion of pulse at FEP stage.
P.EKO	12	(R12) Description of pulse at EKO stage.

Device and DIO addresses (all values are hexadecimal):

NAME	VALUE	DESCRIPTION
TSGPIA	24	TSG PIA (data) DIO address.
TSGEIA	25	TSG EIA (flags) DIO address.
TSGCLK	26	TSG clock DIO address.
HKSEL	33	HKADC channel-select DIO address.
HKREAD	34	HKADC data DIO address.
FEP1	36	FEP data-write DIO address.
FEP2	37	FEP control-byte-write DIO address.
ULM	37	DIO interface device address.
FEP	77	FEP/EKO interface device address.

Miscellaneous definitions:

NAME	VALUE	DESCRIPTION
LOCAL	1Ø(hex)	DIO local control bit mask.
EKOSIZE	1Ø4Ø	Maximum bytes in EKO transfer from FEP.
NEKOBFS	6	Number of PCT/EKO buffers.

7.7. SOUNDER and CALL Task Parameters (SNDR)

NAME	VALUE	DESCRIPTION
Q	4	Q-other-task SVC call number.
LOIX	274	Lowest synthesizer frequency index.
HIIX	3561	Highest synthesizer frequency index.
LOFR	1ØØ	Lowest synthesizer frequency in kHz.
HIFR	3ØØØØ	Highest synthesizer frequency in kHz.
LODF	Ø	Minumum delta F in kHz.
HIDF	3275	Maximum delta F in kHz.
NPROT	5Ø	Max number of protected frequencies bands.
PROTSZ	NPROT * 8 + 4	Size of protected frequencies map.
SØ2SZ	1Ø	Bytes in each SØ2 Q-block.

7.8. Q-blocks

This section contains descriptions of the Q-blocks used for inter-task communications in Product One. In general the first four bytes of a Q-block are in the standard format described in Section 1.6; exceptions are noted below. Some Q-blocks contain only these first 4 bytes, while others contain from 1 to 3 additional information fields of 2 bytes each. There is no standard format for the additional information fields.

Individual Q-blocks are described in Sections 7.8.1 through 7.8.6, grouped by the task that originates or sends them. Section 7.8.7 summarizes the Q-blocks by recipient.

7.8.1. COUNSEL Q-blocks

CØØ
- CØØ + Ø: Ø.
- CØØ + 1: "C".
- CØØ + 2: status.
- CØØ + 4: address of Environment Table (ET) from SOUNDER.

Destination: the CØØ Q-block is first sent to SOUNDER, then in turn to GRAPHER, PICKER, MANAGER, and ANALYSER.

Purpose: this is the first Q-block each task receives after being started by COUNSEL. It is the signal to start task execution and is also used to promulgate global information. The status is ignored.

Recipient action: SOUNDER puts the address of the ET in CØØ + 4. The other tasks, including COUNSEL, subsequently pick up A(ET), and thus the addresses of the global tables and of the reentrant subroutines (Sect. 2.6.3) are known.

C02 C02 + 0: 2.
C02 + 1: "C".
C02 + 2: status.
C02 + 4: task command index, number of item in task's SCAN (0 first, 1 second, etc.). See Sect. 6.3.
C02 + 6: command line address after scan.

Destination: the C02 Q-block is sent to the task whose command mnemonic scan table contains a match of the first word of the command typed at the console, unless the command is directed to COUNSEL.

Purpose: the C02 Q-block notifies a task that a console command has been directed to it. The command index indicates which command has been given; C02 + 6 contains the byte address, in COUNSEL's console input buffer, of the command argument list, if any, or of the terminating carriage return.

Recipient action: the queued task attempts to act on the advice given, and, if successful, clears the C02 status to 0. If the advice was not appropriately given, the status is set to a positive value. So long as the status is left in its initial state, -1000 (hex), COUNSEL will not initiate another command read from the console.

C04 C04 + 0: 4.
C04 + 1: "C".
C04 + 2: status.

Destination: COUNSEL sends the C04 Q-block to all other tasks when command CANHF is given.

Purpose: the C04 Q-block signals that an orderly shutdown of the HF Radar software has been requested.

Recipient action: each task gets its own house in order and then terminates itself. The status-clearing convention is not observed because COUNSEL also terminates itself.

7.8.2. SOUNDER Q-blocks

S00 S00 + 0: 0.
S00 + 1: "S".
S00 + 2: status.
S00 + 4: address of SOUNDER's command mnemonic scan table.

Destination: COUNSEL.

Purpose: S00 is sent to inform COUNSEL that SOUNDER has successfully started and to give COUNSEL the address of SOUNDER's scan table. (See Sect. 6.3 and ET.SCAN, Sect. 7.3.)

Recipient action: COUNSEL copies S00 + 4 to its SCAN table, then starts GRAPHER. The status is ignored.

S02 S02 + 0: 2.
S02 + 1: "S".
S02 + 2: status.
S02 + 4: EKO buffer address.
S02 + 6: address of associated PCT (Sect. 7.2).
S02 + 8: Frequency table entry address (Sect. 7.4), or 0.

Destination: PICKER (or CALL task, Section 7.8.7).

Purpose: S02 is sent to inform PICKER of the PCT/EKO buffer addresses associated with the latest data from the FEP. If the sounding is T- or W-class, S02 + 8 points PICKER to the range limits.

Recipient action: PICKER copies the PCT to its memory, then scans the EKO buffer for range coincident echoes. When through with the EKO buffer, PICKER clears the S02 status to 0. (See Sect. 10.3 for CALL task action.)

S04 S04 + 0: 4.
S04 + 1: "S".
S04 + 2: status.
S04 + 4: sounding mode (SCT.MODE*2).
0 = RUN I[x] (I-mode sounding).
2 = RUN K[x] (K-mode sounding).
4 = RUN V[x] (Poor man's VUF).
6 = RUN B[x] (B-mode sounding).
8 = RUN P[x] (P-mode sounding).
10 = RUN G[x] (G-mode sounding).
12 = RUN Z[x] (Z-mode sounding).

14 = RUN W[x] (W-mode sounding).

Destination: PICKER (or CALL task, Section 7.8.7).

Purpose: SØ4 is sent at the beginning of the countdown to signal the start of a sounding.

Recipient action: PICKER initializes for the sounding and passes the SØ4 Q-block to MANAGER. MANAGER in turn passes it to GRAPHER and to the ANALYSER task, if any (Sect. 4.11). When all pre-sounding action is complete, the SØ4 status is cleared to Ø. (See Sect. 1Ø.3 for CALL task action.)

SØ6 SØ6 + Ø: 6.
SØ6 + 1: "S".
SØ6 + 2: status.

Destination: PICKER (or CALL task, Sect. 7.8.7).

Purpose: SØ6 signals the end of a sounding.

Recipient action: PICKER packs its final output buffer, if any, then passes the SØ6 Q-block to MANAGER. MANAGER in turn passes it to GRAPHER and to the ANALYSER task, if it is running. When all post-sounding action is complete, the SØ6 status is cleared to Ø. (See Sect. 1Ø.3 for CALL task action.)

SØ8 SØ8 + Ø: 8.
SØ8 + 1: "S".
SØ8 + 2: status.
SØ8 + 4: T- or W-class frequency number (Frequency table entry number, Sect. 7.4); Ø for S-class.

Destination: GRAPHER.

Purpose: SØ8 informs GRAPHER of a change in a T- or W-class frequency or range limit.

Recipient action: GRAPHER displays the T- or W-class frequency markers on the previous ionogram plot. The status is ignored.

In addition to the Q-blocks described above, SOUNDER sends the following:

7.8.2. SOUNDER Q-blocks

CØØ to GRAPHER after terminating a CALL task or to PICKER after a task swap. (See ENDCALL and GTPICK, Sect. 2.8.4.)

MØ4 to GRAPHER to suspend display refresh before suspending GRAPHER to bring in a CALL task.

A(CALLST) to CALL tasks. (See CALLST under GTCALL, Section 2.8.4.)

7.8.3. PICKER Q-blocks

PØØ

PØØ + Ø: Ø.
PØØ + 1: "P".
PØØ + 2: status.
PØØ + 4: address of PICKER's command mnemonic scan table.

Destination: COUNSEL.

Purpose: See SØØ, Section 7.8.2.

Recipient action: COUNSEL copies PØØ + 4 to its SCAN table, then starts MANAGER.

PØ2

PØ2 + Ø: 2.
PØ2 + 1: "P".
PØ2 + 2: status.
PØ2 + 4: address of PICKER output buffer (sounding data record).

Destination: MANAGER.

Purpose: PØ2 gives MANAGER the address of the picked data for a pulse set and the associated PCTs.

Recipient action: MANAGER passes the PØ2 to GRAPHER and to ANALYSER, if it is running, then saves the PØ2 buffer on disc.

In addition to sending the PØØ and PØ2 Q-blocks, PICKER also passes on SØ4 and SØ6 to MANAGER. MANAGER, on receipt of the SØ4 Q-block, passes it to GRAPHER and ANALYSER (Sect. 4.11); does some initializaing based on commands DROP and SAVE; and writes the current SCT to disc. (See Q.S.4, Sect. 4.8.)

On receipt of the SØ6 Q-block from PICKER, MANAGER passes it to GRAPHER and ANALYSER, and if the operator has selected tape output (Sect. 4.7), copies the sounding data from disc to tape. (See Q.S.6, Sect. 4.9.)

7.8.4. MANAGER Q-blocks

M00 M00 + 0: 0.
M00 + 1: "M".
M00 + 2: status.
M00 + 4: address of MANAGER's command mnemonic scan table.

Destination: COUNSEL.

Purpose: See S00, Section 7.8.2.

Recipient action: COUNSEL copies M00 + 4 to its SCAN table, then starts ANALYSER.

M02 M02 + 0: 2.
M02 + 1: "M".
M02 + 2: status.
M02 + 4: address of SCT or data record, or 0.

Destination: GRAPHER (or user-written ANALYSER).

Purpose: M02 is sent in response to G02 and G04 requests (Sect. 7.8.5). A02 and A04 (Sect. 7.8.6) are the equivalent requests from a user-written ANALYSER task. When GRAPHER requests data from a previous sounding with the G02 Q-block, MANAGER responds with M02 + 4 = A(SCT), or M02 + 4 = 0 if the data are not available. GRAPHER requests data records with the G04 Q-block; MANAGER responds with M02 + 4 = A(data record), or M02 + 4 = 0 if the data EOF is sensed. Under certain conditions, a user-written analysis task can occupy PICKER's memory partition. If such is the case, PICKER can request data with P04 and P06, which are equivalent to requests A04 and A02, respectively. (See Section 4.11.)

Recipient action: Depending upon whether the M02 Q-block is a response to a request for a previous sounding (file, or data set, request) or to a data record request, GRAPHER either gets scaling and labeling information from the SCT or plots the data record on the system display.

The action of ANALYSER, a user-written task, is undetermined. A02 and A04 are not implemented in the current ANALYSER (Scheduler).

M04 M04 + 0: 4.
M04 + 1: "M".
M04 + 2: status.
M04 + 4: system display write flag.
0 = suspend refreshing system display.
1 = resume refreshing, if refreshing was selected before tape write began.

Destination: GRAPHER.

Purpose: the system display, or refresh CRT, and the magnetic tape share an I/O Selector Channel (SELCH), and a hardware problem has resulted in occasional anomalous or missing tape records. The hardware solution remains illusive, and the conflict is avoided by simply not running the magnetic tape and the system display concurrently. MANAGER tells GRAPHER when the tape write begins and ends.

Recipient action: GRAPHER refrains from refreshing the system display while MANAGER copies the sounding data from disc to tape. When the disc-to-tape operation is completed, GRAPHER resumes refreshing the CRT, unless it is inoperative or command [PE was given. GRAPHER does not clear the status of the M04 queue to suspend refreshing until the next refresh cycle is to begin.

In addition to sending the above Q-blocks, MANAGER also passes S04, S06, and P02 to GRAPHER. GRAPHER, on receipt of the S04 Q-block, initializes a display buffer based on the sounding mode in S04 + 4 (Sect. 7.8.2). The P02 queue causes GRAPHER to add the new data to its plot buffers; the S06 signals that the sounding has ended.

MANAGER also passes S04 and S06 to ANALYSER, if it exists. (See PASS.ON, Sect. 4.11.) The current ANALYSER/Scheduler adds 1000 (hex) to the S04 or S06 status field, and issues no scheduled commands during a sounding.

Finally, MANAGER sends a fake SABORT C02 queue to the Scheduler if command TAPE detects an error, if the end-of-reel marker is sensed during a tape write, or if the data from a single sounding overflows file IONOSOND.SAV.

7.8.5. GRAPHER Q-blocks

G00
G00 + 0: 0.
G00 + 1: "G".
G00 + 2: status.
G00 + 4: address of GRAPHER's command mnemonic scan table.

Destination: COUNSEL.

Purpose: See S00, Sect. 7.8.2.

Recipient action: COUNSEL copies G00 + 4 to its SCAN table, then starts PICKER.

G02
G02 + 0: 2.
G02 + 1: "G".
G02 + 2: status.
G02 + 4: address for sounding SCT.
G02 + 6: sounding number or 999 (end data request).

Destination: MANAGER.

Purpose: GRAPHER's G02 queue is a file request, or a request for the data set from a previous sounding (Sections 4.12, 5.6, and 5.13).

Recipient action: if the requested file is available, MANAGER reads the file's SCT from disc, copies the SCT to GRAPHER memory, and puts the SCT buffer address in M02 + 4. If the data do not exist or have been overwritten, M02 + 4 is cleared to 0. In either case the M02 Q-block is sent in response to the G02.

G04
G04 + 0: 4.
G04 + 1: "G".

Destination: MANAGER.

Purpose: the G04 queue is a request for data from the sounding (data) file selected by the previous G04.

Recipient action: MANAGER reads the next data record and sends the M02 response to GRAPHER. If EOF is sensed, M04 + 4 is cleared to 0; otherwise, the buffer address of the data record is put into M02 + 4. The status clearing convention is not used.

7.8.5. GRAPHER Q-blocks

MANAGER is written to handle file (AØ2 and PØ6) and data (AØ4 and PØ4) requests from user-written ANALYSER or PICKER tasks. GRAPHER and ANALYSER or PICKER requests are logically identical from MANAGER's standpoint and can be interspersed (Sect. 4.11). That is to say, either ANALYSER or PICKER requests may be interspersed with GRAPHER requests; however, mixing PICKER with ANALYSER requests may cause conflicts.

7.8.6. ANALYSER Q-blocks

AØØ AØØ + Ø: Ø.
AØØ + 1: "A".
AØØ + 2: status.
AØØ + 4: address of ANALYSER's command mnemonic scan table.

Destination: COUNSEL.

Purpose: See SØØ, Section 7.8.2.

Recipient action: COUNSEL copies AØØ + 4 to its SCAN table, re-orders the table in command priority, and puts the table address into ET.SCAN (Sect. 7.3). All tasks have been started by and have responded to COUNSEL; COUNSEL is ready to accept sounding advice from the console.

The AØØ Q-block is sent by the current ANALYSER/Scheduler and must also be sent by any user-written ANALYSER task. AØ2 and AØ4 requests can be sent and should be coded by the user to function like GØ2 and GØ4 (Sect. 7.8.5). AØ6 has tentatively been reserved for frequency selection feedback to SOUNDER. If AØ6 or other queues to SOUNDER are implemented, the user must also insert appropriate routines at A.ADV (Sect. 2.6.1).

When ANALYSER acts as the scheduler (see CALL task SCHED, Sect. 1Ø.2.5), it sends CØ2 Q-blocks as though it were COUNSEL (7.8.1).

7.8.7. Q-block summary by recipient

COUNSEL	SØØ	SOUNDER started
	PØØ	PICKER started
	MØØ	MANAGER started
	GØØ	GRAPHER started
	AØØ	ANALYSER started
SOUNDER	CØØ	start
	CØ2	Console advice
	CØ4	Terminate
	AØ6	Frequency selection (must be user-implemented)
	TFF	CALL-task-termination/console-release requests
PICKER	CØØ	Start (from COUNSEL)
	CØ2	Restart (from SOUNDER)
	CØ2	Console advice
	CØ4	Terminate
	MØ2	Response to data/file request (user implemented)
	SØ4	Sounding start
	SØ2	Sounding data
	SØ6	Sounding end
MANAGER	CØØ	Start
	CØ2	Console advice
	CØ4	Terminate
	PØ4	Data request
	PØ6	File request
	SØ2	Direct recording
	SØ4	Sounding start
	SØ6	Sounding end
	PØ2	Sounding data record
	GØ2	File request
	AØ2	File request
	GØ4	Data request
	AØ4	Data request
GRAPHER	CØØ	Start (from COUNSEL)
	CØØ	Restart (from SOUNDER)
	CØ2	Console advice
	CØ4	Terminate
	SØ4	Sounding start
	SØ6	Sounding end
	PØ2	Sounding data record
	SØ8	K-mode frequency, range limits
	MØ2	Response to file, data request
	MØ4	System display off, on

7.8.7. Q-block summary by recipient

```
ANALYSER  CØØ  Start
          CØ2  Console advice (Scheduler)
          CØ2  Console advice (user-written ANALYSER)
          CØ2  SABORT from MANAGER (Scheduler)
          CØ4  Terminate
          SØ4  Sounding start
          SØ6  Sounding end
          PØ2  Sounding data record
          MØ2  Response to file/data request (user-implemented)
```

All CALL tasks receive the address of table CALLST from SOUNDER as the start queue. (See Sect. 2.8.4.) In addition, the following CALL tasks receive queues from SOUNDER (Sect. 1Ø.3):

```
TENUS     (Copy PCT/EKO buffers to magtape)
          SØ4  Start
          SØ2  Data
          SØ6  End

MEANS     Compute, save means of P-mode data.
          SØ4  Start
          SØ2  Data
          SØ6  End

MENUS     Same as TENUS, but subtracts MEANS from data.
          SØ4  Start
          SØ2  Data
          SØ6  End
```

EXOUT, SPLOT and TPLOT, in order to gain access to data from a previous sounding from MANAGER, assume the role of GRAPHER, inasmuch as these tasks send a GØ2 queue to MANAGER to select a sounding data file and a GØ4 queue to get a data record. In turn, these tasks receive two queues from MANAGER. EXOUT is used as an example:

```
EXOUT     (Examine PICKER output data)
          MØ2  Response to GØ2
          MØ2  Response to GØ4
```

7.9. DIO Control Tables (DIO)

The configuration of the DIO system is specified from the console to CALL task DIOGEN (Section 10.2.4). DIOGEN writes the DIO configuration to a disc file, SYS1:DIO.xxx; the operator then invokes a DIO configuration for a given sounding mode with the command BOOT DIO[,xxx]. If xxx is not specified, the DIO control tables from SYS1:DIO.TWO are read into SOUNDER's DIO table.

Although DIOGEN writes sectors 0-8 to a DIO file, SOUNDER reads only sectors 6-8 into its DIO table. The first 6 sectors of a DIO file contain the DIO Configuration Table (DCT) and a raw pulse-to-pulse table (PTP) used internally by DIOGEN to set up the tables used by SOUNDER.

SOUNDER's DIO table contains several sections, or sub-tables. The first section, DIO.DED, is the table of dedicated DIO device addresses. Each dedicated DIO device is supported explicitly by routines in SOUNDER.

Following section DIO.DED are DIO.AMXS, DIO.FEPS, DIO.TXS, and DIO.XTRA, areas set aside for multiple DIO devices. Only DIO.AMXS is currently used. DIO.AMXS has room for the DIO addresses of 8 outboard antenna multiplexors, which are numbered according to their position in DIO.AMXS as specified to DIOGEN. (See command ANT, Section 2.7.6 and Chapter 8.) DIO.XTRA allows room in the DIO table for future growth.

Following next is the DIO Sounding Table, DIO.DST, containing the masks, control flags, and data used to control each DIO port during and after sounding. DIO.DST is sorted by DIOGEN in priority order for SOUNDER's DIOLOOP (Section 2.9.1.1).

The last section of SOUNDER's DIO table is DIO.PTP, containing the pulse-to-pulse data for all ports so defined by the user. Each DIO.PTP entry, or pulse palette extension, contains 10 bytes. The first 2 bytes hold a port's DIO address, and the remaining 8 bytes contain the pulse-to-pulse data for the port in pulse palette order. A port address of 0 flags the end of DIO.PTP. The sequence in which the data bytes are used during a pulse set is specified by command PSEQ (Section 2.7.4). The DIO.PTP section of SOUNDER's DIO table is written with each sounding data set by MANAGER, preceeded by a 1/2-word (2 bytes) with the number of bytes in the DIO.PTP record in the low order 12 bits and the flag, 9, in the high order 4 bits. The address of the DIO.PTP section is put into ET.ADIO for MANAGER.

7.9. DIO Control Tables (DIO)

The DIO table layout is summarized below.

DISPLACE	BYTE	OFF	DESCRIPTION
DIO.DED	32	Ø	Dedicated DIO device section.
DIO.AMXS	8	32	Outboard antenna mux section.
DIO.FEPS	8	4Ø	(Currently not used.)
DIO.TXS	16	48	(Currently not used.)
DIO.XTRA	64	64	(Reserved for future growth.)
DIO.DST	384	128	DIO Sounding Table section.
DIO.PTP	2ØØ	512	Pulse-to pulse section.

Dedicated DIO ports are always referenced by SOUNDER through DIO.DED. CALL tasks and other support programs that use the DIO bus do not consult this table for DIO addresses. When activating a dedicated port with DIOGEN, the user refers to the suffix, or the four characters following the period, of a DIO.DED entry. For example, the name of the channel 1 receiver input given to DIOGEN is "1RIN".

The current DIO.DED assignments are listed below. All DIO addresses are in hexadecimal; all DIO.DED entries are one byte.

DISPLACE	DIO	OFF	DESCRIPTION
DED.1RIN	1Ø	Ø	Receiver input, channel 1.
DED.2RIN	11	1	Receiver input, channel 2.
DED.1FLT	--	2	Octave filter, channel 1 (deactivated).
DED.2FLT	--	3	Octave filter, channel 2 (deactivated).
DED.1RFA	14	4	RF attenuator, channel 1.
DED.2RFA	15	5	RF attenuator, channel 2.
DED.1IFA	16	6	IF attenuator, channel 1.
DED.2IFA	17	7	IF attenuator, channel 2.
DED.1BDW	18	8	Bandwidth control, channel 1.
DED.2BDW	19	9	Bandwidth control, channel 2.
DED.SYNS	1E	1Ø	Synthesizer status.
DED.CALA	1A	11	Calibration attenuator.
DED.TXPB	--	12	Phase, TX B (deactivated).
DED.FRAB	1D	13	Synthesizer frequency, digits A, B.
DED.FRCD	1C	14	Synthesizer frequency, digits C, D.
DED.FREF	1B	15	Synthesizer frequency, digits E, F.
DED.TSGA	24	16	TSG command.
DED.TSGB	25	17	TSG flags.
DED.TSGC	26	18	TSG clock.
DED.TXCT	3Ø	19	Transmitter control.

DISPLACE	DIO	OFF	DESCRIPTION
DED.TXAT	31	20	Transmitter attenuator.
DED.TXSA	32	21	Transmitter status A.
DED.HKAD	33	22	HKADC address select.
DED.HKRD	34	23	HKADC read.
DED.TXSB	35	24	Transmitter status B.
DED.FEPA	36	25	FEP program load.
DED.FEPB	37	26	FEP control.
DED.XTRA	--	27	(5 bytes reserved for future growth.)

There is a DIO.DST entry for every DIO port, and each DIO.DST entry contains six bytes. The data written to a DIO port during sounding is put into DST.PDTA by a NXTFRQ (Section 2.9.1.2) call to subroutine SETDST (Section 2.6.2). The data byte put into DST.PDTA by NXTFRQ is computed, as in the case of the attenuators, or comes from the pulse palette (SCT.PD) or from a pulse palette extension in DIO.PTP.

The DIO.DST entry is shown below.

DISPLACE	BYTE	OFF	DESCRIPTION
DST.PORT	1	0	Port DIO address.
DST.DMSK	1	1	Data-direction mask.
DST.PMSK	1	2	Pulse-to-pulse mask.
DST.PDTA	1	3	Pulse-to-pulse data.
DST.IDTA	1	4	Post-sounding (idle) data.
DST.CTRL	1	5	Control flags.

The control flags in DST.CTRL are described below with their hexadecimal masks. A true (1) control bit means the following:

FLAG	MASK	MEANING
DIO.VRFY	1	Verify write with read-back.
DIO.IDLE	2	Write DST.IDTA after sounding.
DIO.RDIO	4	The DIO port device is remote.
DIO.UDIO	8	The port device is user-defined pulse-to-pulse.
DIO.AMUX	10	The port device is an outboard antenna mux.
DIO.AFEP	20	The port device is a FEP (not currently used).
DIO.ATX	40	The port device is a transmitter (not used).
DIO.DDIO	80	The port is dedicated.

8. COMMANDS (Console Advice)

Lorne D. Matheson
David C. Walden
James R. Winkelman

This chapter gives brief descriptions of all commands that may be entered from the console device to the HF Radar sounding tasks. Refer to the individual task chapters for details not presented here. (Commands accepted by the CALL tasks are described under Section 1Ø.2.) The commands are in alphabetic order. The marginal line after each command gives the initial of the task that acts on the command. The G for GRAPHER may be followed by Immediate if the command usually takes effect immediately rather than at the start of the next sounding or data replay from the disc. The S for SOUNDER is followed by IMA (Idle-Mode Advice) or SPC (Sounding Pipeline Control) denoting SOUNDER's state of execution, and is followed by MD if the command is sounding Mode Dependent; that is, if the command argument(s) are retained as part of the mode definition.

For all commands, optional arguments appear in square brackets.

SOUNDER commands ANT, DFAC, TSGC, and UDIO all contribute to defining the 8 pulse descriptions in the pulse palette. (See Section 2.5.) The pulse description index (pd) must be specified in the ANT argument list. All of the other palette commands, however, have an argument list of the form, n[,m,...]. The list specifies 1 to 8 values directed, in order, to pulse descriptions 1 through 8. The list may be terminated early, and if it is, the last value entered is taken to define the appropriate palette element for the remaining pulse descriptions not referred to by the shortened list. For example,

DFAC 1,2,3,4 is taken as DFAC 1,2,3,4,4,4,4,4.
TSGC A6 is taken as TSGC A6,A6,A6,A6,A6,A6,A6,A6.

Command PSEQ specifies the order in which pulse descriptions 1 through 8 are used to define the pulses in a pulse set; PTP and UDIO display what the above commands specify.

SOUNDER commands BDW, IFAT, RFAT, PKLO, and PKHI specify parameters used in the control of the two receiver channels, and have an argument list of the form, n[,m]. If n only is given, n applies to both receiver channels; if n,m is given, n applies to channel 1 and m applies to channel 2.

Mandatory command letters are underlined; the others can be typed, but are not required for command recognition.

AB S, IMA, SPC	Terminate the running of a sounding or cancel scheduled sounding. If the AB command is given during the running of a sounding, the sounding is terminated at the end of the current frequency's pulse set. If a sounding is aborted during the countdown (see RUN commands), the countdown continues to Ø but no pulses are transmitted; because SOUNDER sends the sounding-started Q-block (SØ4, Section 7.8.2) at the beginning of the countdown and the sounding-ended Q-block (SØ6) as a result of the AB, MANAGER saves a null sounding on the disc. If the AB command is given after the completion of a sounding, i.e., between soundings, the next scheduled sounding is canceled if an SPH schedule is active. At the end of a sounding, whether it runs to completion or is ABed, SOUNDER sets the receiver and transmitter to maximum attenuation and sets the control mode for receiver devices to local, if so configured by DIOGEN (Section 1Ø.2.4). The AB command cancels any NO T, NO F, NO E, or NO A commands (qq. v.) previously issued. That is, for the next sounding SOUNDER will assume that the TSG, FEP, EKO, and housekeeping ADC (HKADC) are cabled into the system and functioning normally, if so configured by DIOGEN (Section 1Ø.2.4). An AB during a tape replay run (see TPR) causes the magnetic tape to read forward to an EOF.
AB S, SPC	Given during a sounding, AB aborts both the sounding and the SPH schedule, if active.
AGC Ø S, IMA, MD	The RF and IF attenuators for both channels 1 and 2 are set to manual control, and can be set from the console with RFAT and IFAT.
AGC 1[,rf1,Pn] S, IMA, MD	The RF and IF attenuators for channel 2 are set to manual. Channel 1 uses AGC. The rf1,Pn is optional; rf1 is the base RF attenuation setting, Pn

(not fully implemented) is the number of pulses after which the AGC is updated.

AGC 2[,rf2,Pm]
S, IMA, MD

The RF and IF attenuators for channel 1 are set to manual. Channel 2 uses AGC. The rf2,Pm is optional.

AGC 3[,rf1,rf2,Pn,Pm]
S, IMA, MD

Both channels 1 and 2 are set for automatic gain control (AGC). The rf1,rf2,Pn,Pm are optional. If rf1 or rf2 is Ø through 2Ø dB, the base IF attenuation is set to Ø dB; if rf1 or rf2 is 24-6Ø, the base IF is 2Ø dB less than the RF. If rf1 or rf2 is greater than 6Ø dB, both the base RF and IF are set to 6Ø dB to allow testing at full attenuation.

ANT pd,a,b
S, IMA, MD

Specify the receiver antenna selection for palette description pd; a and b specify the inputs for channel 1 and channel 2.

The selection of antenna inputs depends on how the antennas are plugged into the inputs. If the N and E antennas are connected to inputs 1 and 2 of channel 1, and if the S and W antennas are connected to inputs 1 and 2 of channel 2, then the commands:

```
ANT 1,1,1
ANT 2,2,2
ANT 3,1,1
ANT 4,2,2
```

provide the default transmission sequence displayed in Figure 2.1 and in Table 2.1.

An expanded form of the ANT command can be used if outboard antenna multiplexors are configured into the DIO system. (See DIOGEN, Section 1Ø.2.4.) For example:

```
ANT pd,2:2-3,2:3-4
```

means that for palette description pd, SOUNDER is to select inputs 2 for both channels. However, input 2 for channel 1 is fed by antenna mux number 2, with input 3 of mux 2 selected; input 2 of channel 2 is fed with mux 3, with input 4 of mux 3 selected. If either mux 2 or 3 were not configured into the DIO system, the above command would result in ERR 18.

BDW
S, IMA, MD
Use current receiver bandwidths, and select the narrow band low power RF system filters.

BDW W
S, IMA, MD
Use current receiver bandwidths, and select the wide band low power RF filters, which allow for 3Ø microsecond transmitter pulses.

BDW n[,m]
S, IMA, MD
Specify receiver bandwidths; m specifies the second channel if different from the first channel. Select the narrow band low power RF filters.

BDW n[,m],W
S, IMA, MD
Specify receiver bandwidths and select the wide band low power RF filters.

BFILE
M, Tape.
Backspace one tape file.

BIAS field
C, Debug.
Set hexadecimal bias value for debugging.

BOOT DIO[,xxx]
S, IMA, MD
Load the DIO control tables from the specified file SYS1:DIO.xxx. If xxx is not given, TWO is assumed for the extension. (See DIOGEN, Section 1Ø.2.4.)

BOOT FEP[,xxx]
S, IMA, MD
Initialize and load the FEP. The FEP is reset, via the DIO, sent its program, which must reside on disc as file SYS1:FEPMEM.xxx where xxx, the file name extension, is any three alphanumeric characters. If xxx if not given, TWO is assumed for the extension.

BOOT TSG[,xxx]
S, IMA, MD
Load the TSG zero page and user pages 2 and 3 from file SYS1:TSGMEM.xxx, where xxx, the file name extension, is any three alphanumeric characters. If xxx is not given, TWO is assumed for the file extension.

CALA n[,list]
S, IMA, MD
Set the calibration attenuation in dB. The calibration attenuator is set in 8-dB steps from Ø through 12Ø dB. If an out-of-step setting is specified, the next lower step is selected. For

example, CALA 21 is equivalent to CALA 16. The optional list contains 1 to 4 frequencies at which to step the calibration attenuation -8 dB, for installations with large cable losses between the radar and the receiving antenna field.

CALL task S, IMA, MD	Specify a data recording or analysis task for run time. (See Section 1Ø.3.)
CANHF C	Cancel all tasks.
CDN n S, IMA, MD	Specify the countdown value in seconds. The countdown can be set to 1 through 19 seconds.
CEND C	End a special COUNSEL command file. Open CON: as the input device and continue.
CLOSE M, Tape.	Close tape, write double EOF, and position properly for a later TAPE command.
COMMAND fn C	Open file fn as a command file that normally contains parameters, type formats, and the TSG and FEP programs. The file must end with a CEND or another COM. Files are not nested, the new file replaces the old. The file, fn, should include a file name extension, to distinguish it from an interactive device.
CONT S, IMA	Continue with the aborted soundings per hour (SPH) schedule, with the sounding mode that was current when SPH was specified.
COPY [n] M, Tape.	Copy the previous n soundings to tape. If n is absent, COPY writes all soundings since MANAGER was started or since the last sounding which was copied to tape. Soundings which have been overwritten are not copied. Parameter n must be less than 124. The tape is left closed.
DELF f S, IMA, MD	Set the delta frequency in kHz. DELF is initialized to 8 kHz (Ø for W-class) and must be in the range Ø through 3275 kHz. This is the deltaF in Table 2.1 used by command DFAC. Command DELF is rejected with ERR 14, if given for a W-class sounding mode. (See W-mode under MAKEM, Section 1Ø.2.1.)

DESC desc S, IMA, MD	Follow command by up to 8 ASCII characters describing the sounding mode. Fewer than 8 characters are right filled with blanks by SOUNDER.
DFAC n[,m,...] S, IMA, MD	Set the delta F factor(s) (-128 through +127). The product DFAC*DELF is added to each base or ramp frequency to determine the sounding frequency for a given pulse.
DIO Da,Dd[,Dm] S, IMA	Set DIO address Da to data byte Dd [optionally masked with Dm]. All values are expressed in hexadecimal.
DLOF S, IMA	Select the transmitter antenna load in place of the dummy load. The filament and high voltage states are unaffected.
DLON S, IMA	Select the transmitter dummy load (DL). DLON preserves the states of the filaments and the high voltage.
DROP [m] M, Tape.	Drop m soundings (including the current one) by writing m soundings over the current one on disc. MANAGER enters the SAVE mode when the countdown ends. If m = Ø or is missing, it is set to 1. If it equals or exceeds 9999, MANAGER remains in the DROP mode without counting down. The latest sounding is available for replotting.
ECHO n P	There is no action taken if n is zero or blank (default). If n is not zero, the message: mmmm ECHOS OUT OF nnnn is printed after each sounding. If more than 65536 echoes have been processed, the numbers will be modulo 65536. Value mmmm is the count of data pairs written to the PØ2 buffers, nnnn also counts the echoes which failed the KEEP criterion. Note that use of the FEP PICKER or EQR has already imposed a KEEP 4 on the data.
EFRQ f S, IMA, MD	Set the S-class ending frequency in kHz. The ending frequency can be set to any value up to 3Ø MHz provided that EFRQ > SFRQ.
NOTE:	All S-class starting and ending frequencies are checked against the protected frequencies map (Sect. 1Ø.2.3). If a specified starting frequency is protected, the next higher unprotected frequen-

cy is used; if a specified ending frequency is protected, the next unprotected frequency that is STEP steps lower is used.

In addition to manipulating the sounding frequency directly, SOUNDER may calculate the next frequency for Sweep-class modes and Table-class ramps by doing arithmetic on a frequency index. The frequency index ranges from Ø through 36ØØ representing frequencies from 62.5 kHz through 32 MHz; the index is linear in the log of the frequency. The frequency range represented is 9 octaves with 4ØØ selectable frequencies per

FREQUENCY INDEX	FREQUENCY, kHz
Ø	62.5
4ØØ	125.Ø
8ØØ	25Ø.Ø
12ØØ	5ØØ.Ø
16ØØ	1ØØØ.Ø
2ØØØ	2ØØØ.Ø
24ØØ	4ØØØ.Ø
28ØØ	8ØØØ.Ø
32ØØ	16ØØØ.Ø
36ØØ	32ØØØ.Ø

Table 8.1. FREQUENCY INDEX vs. FREQUENCY

octave, as shown in Table 8.1. Subroutine NTOFR is called to convert frequency index to frequency; FRTON converts frequency to the nearest corresponding frequency index. (See Section 2.6.3.) Because of hardware constraints, SOUNDER limits the actual frequency range to 1ØØ kHz through 3Ø.Ø MHz, corresponding to indices 274 through 3561. See SNDR, Section 7.6.

The general formula relating frequency f and index I is f = Ø.Ø625*2**(I/4ØØ) MHz, where * means multiplication and ** exponentiation.

EPF n
S, IMA, MD

Specify the maximum number of range-coincident echoes per frequency for PICKER. EPF is normally 8, but can be up to 16 for the standard PICKER and 52 for the PICK4PPS, PICK5PPS, OR PICK6PPS configurations of PICKER. (See Section 3.1.) The

EPF n

EPF specified includes the mean, max (noise) record, if specified by MEAN, as well as the calibration pulse record.

F
S, IMA

Display all active T- or W-class frequencies and their associated range limits.

F nF
S, IMA, MD

Display the nFth T- or W-class frequency, if active, and its range limits, where nF lies between Ø and 9 inclusively. The following options affect the previously selected nFth frequency:

F nF+[m] Raise the nFth frequency one or m steps of Ø.1 kHz.

F nF-[m] Lower the nFth frequency one or m steps of Ø.1 kHz.

F nFX Delete the nFth frequency. The ranges (FR.MIN and FR.RMAX, Section 7.4) are reset to the default values of Ø-5ØØ km for T-class modes and Ø-1ØØØ km for W-class modes. The minimum range of Ø km keeps the calibration data by default for a newly selected frequency for entry nF. The attenuation settings in FR.ATTN are reset to the values in MD.BAG (Section 7.1).

F nF,<freq>
S, IMA, MD

Define the nFth T- or W-class frequency; nF is between Ø and 9 inclusively. The frequency, <freq>, is expressed in kHz. If the frequency selected is protected, the next higher unprotected frequency is used. The attenuation settings in FR.ATTN are reset to the values in MD.BAG.

FEP n
S, IMA, MD

Specify the FEP data threshold. The argument for the FEP command is an integer that corresponds to a thresholding factor.

FEP thresholding values:

INTEGER	FACTOR
1	1.5
2	2.Ø
3	2.5
4	3.Ø
5	3.5
6	4.Ø
7	4.5
8	5.Ø

FEP Fa,Fc,Fk
S, IMA, MD
Pass a count, Fc, and a constant, Fk, to FEP location Fa. The count and constant must be less than 1Ø24, and the location, Fa, must be 2, 3, or 4. FEP PARTIAL processing saves Fc data pairs, starting at FEP input memory location Fk, and gets this data from Fa = 3. FEP PASS uses location Fa = 4 the same way. Fa = 2 is currently unspecified. The FEP command forces Fk to be an even number.

FFILE
M, Tape.
Move the tape forward, one file.

FORMAT [n]
M, Tape.
Specify tape format:
n zero or missing: expanded format, if available.

n not zero: compact format with EOFs every n soundings.

GAB
G, Immediate.
Abort a GPION or GPSKY request and terminate disc activity.

GCAL
G, Immediate.
Plot the calibration pulses with the other data.

GHMAX hhh
G
Set the upper edge of the ionogram display to hhh kilometers virtual height. The specified value is truncated to 5Ø-km increments. GHMAX must be at least 1ØØ km greater than GHMIN.

GHMIN hhh
G
Set the lower edge of the ionogram display to hhh kilometers virtual height. The specified value is truncated to 5Ø-km increments. GHMIN must be at

least 1ØØ km below GHMAX.

GMOV G, Immediate.	Allow movement/segment alternation while plotting multiple segment buffers.
GNOCA G, Immediate.	Suppress the plotting of the calibration pulses, which is the default condition.
GNOMV G, Immediate.	Plot all segments in multiple buffers without alternation.
GNORM nn G	Set spectral analysis normalization.
GNPT[-]nnn G	The maximum number of points displayed is nnn. The minimum is set to 1ØØ points. The optional minus sign will suppress new points replacing the old when the buffer is full. (The FFT overlay in GRAPHER is not completely implemented in P2.Ø.)
GONO G,Immediate.	Plot ordinary ray without the extraordinary ray.
GONX G, Immediate.	Plot extraordinary ray without the ordinary ray.
GOVER n G	Force GRAPHER overlay n for graphics processing and display, where n has the following values: 1: ionogram display. 2: K-mode display. 3: skymap display. 4: partial reflection display. 5: selected height vs. time. 6: spectral analysis display. 7: signal, background display. 8: height vs. latitude display. 9: NS or EW deviation vs. time. Ø: restore automatic overlay selection.
GPION nnn G	Get previous sounding data set, nnn, from the disc. Plot number Ø refers to the most recent data set put on the disc, 1 refers to the penultimate set put on the disc, etc.

GPSKY nnn
G
Get the previous data set as in GPION and display it in SKYMAP mode.

GSCALE x1,x2,y1,y2,c,a
G
Set nondefault scaling parameters for several overlays. The meaning of the parameters is overlay specific. The abscissa scaling is set by x1,x2, the ordinate by y1,y2. Optional conversion from linear components to O,X is controlled by c. If c is zero, optional conversion takes place. The increment between linear components in bytes is set by a. It is normally a multiple of four. The command may be terminated early after the 2nd, 4th or 5th parameter.

GSCASK f1,f2,h1,h2,s
G
Set an acceptance window and scaling for SKYMAP type overlays. This command only influences the display, not the data recorded on the disc or tape. The window is from f1 to f2 kHz and from h1 to h2 km virtual height. The parameter, s, in km sets the scaling of the display. The command may be terminated early, after the second or fourth parameter.

GTHIN n
G
Plot only every nth pulse set. Only implemented for some overlays.

HPAT n
S, IMA, MD
Specify n dB attenuation for the high power transmitter. The default attenuation is 9 dB. The range is 3-31 dB in 1-dB steps. The default value is chosen to optimize high-power transmitter operation at maximum power. The attenuation may be increased from this value to decrease power.

HPE n
S, IMA, MD
Specify the number of half-words per echo for the picked data. HPE is typically 17 for the default 4 pulses per pulse set, allowing for a range value and 8 X-Y pairs. MD.HPE, the value in the SCT set by this command, is also set implicitly by command PSEQ. For complementary coded soundings, PSEQ should be followed by HPE. (See PSEQ, Section 2.7.4, for details.)

HVOF
S, IMA
Turn off the high power transmitter high voltage. If the system is left idle for any length of time, it is advisable to turn off the high voltage.

	HVOF may be issued at any time when not sounding, and does not affect the transmitter filaments or the state of the dummy load (DL).
HVON $\overline{S}$, IMA	Turn on the high power transmitter high voltage. The high power high voltage is controlled independently of the filaments, but the filaments must be on and warmed up before the HV is turned on. If TXON has not been issued or the TX HV HOLD is true, indicating incomplete filament warm-up, HVON produces an error message, ERR 8. HVON preserves the state of the dummy load (DL).
IFAT n[,m] $\overline{S}$, IMA	Specify Sweep-class receiver IF attenuation(s); use m for channel 2 if different from channel 1. This command only influences receiver channels under manual gain control. To set the base RF attenuations, use command AGC.
IFAT nF,n[,m] $\overline{S}$, IMA	Specify T- or W-class receiver IF attenuation(s) for frequency nF. This only applies to manual gain control channels.
IF n[,m] $\overline{S}$, SPC	Set Sweep-class receiver IF attenuation(s) during a sounding. The command is ignored if the receiver channel(s) are under AGC.
IF nF,n[,m] $\overline{S}$, SPC	Set T- or W-class receiver IF attenuation(s) for frequency nF during a sounding under manual gain control.
KEEP m $\overline{S}$, IMA, MD	Specify the M-out-of-N range coincidence criterion for PICKER. The number of possible coincidences (N) is set to (pulses per set)*2, normally 8. M can be set by the KEEP command to any integer in the range 1 through N. KEEP is ignored by PARTIAL processing (Section 3.11). For FEP EQR processing, only KEEP 4 and KEEP 8 have an effect; for FEP PICKER processing, only KEEP 4, 6, or 8 are used. (See Sections 3.12 and 3.13.)
LINE field $\overline{C}$, Debug.	Display the next 24 bytes (12 half-words) after the field address in hexadecimal. After execution, LINE defines Y = X + 16 and Z = X + 24 (in bytes) rather than the usual definitions described in Section 6.2. X, however, retains its

definition as the given field value.

LPAT n
S, IMA, MD

Specify n dB attenuation for the low power transmitter. The transmitter attenuator is set in 1-dB steps from 3 through 31 dB. This attenuation is initialized to 6 dB, which optimizes low power transmitter operation at maximum power.

MDEL <id>
S, IMA

Delete sounding mode <id> from the sounding mode directory.

MDIR
S, IMA

List the sounding mode directory.

MEAN [n]
P

n zero: MEAN and MAX are not included in the PØ2 block.

n not zero: The MEAN and MAX for each channel are saved. This "noise" record is given a range of -1.

MEMORY f1[,f2]
C, Debug.

Display contents of memory address f1. If field f2 is present, replace the contents of address f1 with the value f2. This replacement requires a YES or OK. The possible values for these fields are described in Section 6.2.

MGET [<id>]
S, IMA

Get and list sounding mode <id>. MGET (cr) lists the current mode.

MSAV <id>
S, IMA

Save the current sounding configuration as sounding mode <id>.

NFIL n
S, IMA, MD

Specify n TSG NOP intervals (fill pulses) between pulse sets. The pace interval of the last TSG command byte in the pulse set is used for the NOP fills.

NO A/E/F/T
S, IMA

Choose one or all of the choices: A for no ADC, E for no EKO device, F nor no FEP, or T for no TSG. A typical multiple argument command might look like NO A F with one space or separator between arguments.

NO A means flag the housekeeping ADC as not in the system. SOUNDER will not wait for the HKADC end-

NO A/E/F/T

of-conversion signal if NO A is used, so some time is saved in the SPC loop. NO A remains in effect for a single sounding, until SN is satisfied, or until AB is typed, i.e., until the IMA loop is re-entered. (See GETADV, Section 2.6.1.)

NO E means flag the EKO device as not in the system. NO E should be used in conjunction with NO T, and is canceled like NO A, above.

NO F does nothing since the FEP is not under explicit SOUNDER control in P2.Ø (See PIPLN, Section 1Ø.2.15.)

NO T means flag the TSG as not in the system. For debugging, SOUNDER can be run without the TSG by using the OS delay instead of waiting for the TSG DONE interrupt. NO T is canceled like NO A, above.

NOTE: If SOUNDER detects an error condition indicating that the TSG, EKO, or HKADC is not functioning normally, the error is flagged in MD.STAT (Section 7.1), and an error message is logged. (See ERROR, Section 2.6.2.) The SOUNDER task flags the device in question as not in the system, just as if one of the above commands were issued, and continues to run. The operator can now AB the sounding and intervene, for example, by BOOTing the TSG or FEP or by turning on the power for the receiver devices. The AB cancels the not-in-system commands (as described above) and permits the other tasks to terminate gracefully, e.g., MANAGER will write the double EOF to magnetic tape, if required. The operator can continue without having to reload the HF Radar software from disc.

NOQUEUE [n]
M

n zero: send PØ2 queue blocks to GRAPHER only.

n one: send PØ2 queues to GRAPHER and ANALYSER.

n not zero or one: save time by telling no one.

NSET n
S, IMA, MD

Specify n passes through the base frequency selection (Section 2.5). Normally, n is 1 for S-class and more than 1ØØ for T- or W-class soundings.

NTAPE
M, Tape.

When NTAPE is given before a sounding, it prevents that sounding from being written to tape. NTAPE might be used before a short sounding to check conditions, when the data are not wanted. The file is saved on disk, though, and will be written by a COPY, unless DROPped.

[PA
G

Display test pattern points only.

PAUSE
C, Debug.

Pause 30 seconds during debugging.

[PB
G

Display test pattern vectors only.

[PC
G

Generate ionogram grid and labels from the SCT stored in GRAPHER; primarily used for graphics diagnostics and system tests.

[PE
G, Immediate.

End display refreshing.

PIK task
S, IMA, MD

Specify the PICKER task for the currently selected sounding mode. The task name specified remains associated with the sounding mode. PICKER task configurations named PICKER, PICK4PPS, PICK5PPS, and PICK6PPS are described in Section 3.1.

PKHI n[,m]
S, IMA, MD

Specify the maximum FEP data peak in mV. PKHI is initialized to 9000 mV for S-class and 3000 for T- and W-class modes. If the maximum FEP data peak seen by SOUNDER for a pulse set is greater than this value, more attenuation is put into the receivers, if under AGC. Note that the two receiver channels are attenuated independently.

PKLO n[,m]
S, IMA, MD

Specify the minimum FEP data peak in mV. PKLO and PKHI (above) are used to set the limits for the automatic gain control (AGC) routine for the receivers. PKLO is initialized to 3000 mV for S-class and 1000 for T- and W-class modes. If the maximum FEP data peak for a pulse set is less than this value, less attenuation is put into the receivers.

PLOTAMP n,hhh

PLOTAMP n,hhh
G

Sets AMPTEST to Ø to plot amplitudes. The pulse, channel number to be plotted in the range of 1 to 12 is n. The first height above hhh kilometers is the point plotted. This command only applies to specific overlays.

PLOTPH n,hhh
G

Sets AMPTEST to 1 to plot phases. The pulse, channel number to be plotted in the range of 1 to 12 is n. The first height above hhh kilometers is the point plotted. This command only applies to specific overlays.

[PS
G, Immediate.

Restart display refreshing.

PSEQ n [,n] [,m(n,n)] [,m(n)]
S, IMA, MD

Specify pulse set sequence as a series of pulse palette indexes. Each index points to a particular pulse description in the pulse palette and are in the range of 1 through 8. One or more indexes may be enclosed within parentheses and preceded by a repeat count, m. The parenthetical series is repeated m times. Command PSEQ sets the SCT.PSEQ table (Section 7.1) and by inferrence MD.OPOB, MD.PPS, MD.N, and MD.HPE. For complementary coded soundings and soundings that have TSG NOPs embedded in the pulse sets, PICKER will set MD.OPOB to its MAX.OPOB, and command HPE should be given after PSEQ. (See PSEQ, Section 2.7.4, for further details.)

The default sequence for I-, K-, B-, V-, and Z-mode is 1,2,3,4.

PTP
S, IMA, MD

List pulse-to-pulse specifications, or palette descriptions, and the pulse set sequence set by PSEQ, above. Palette extensions are listed with command UDIO.

[PZ
G

Display test pattern points and vectors.

RAMP
S, IMA, MD

Cancel the existing ramp specifications. A null ramp (1,1,Ø,B) results.

RAMP nR,nS,i,B S, IMA, MD	Specify nR B-mode ramps with nS frequency index steps of i.
RAMP nR,nS,F<freq>,B S, IMA, MD	Specify nR B-mode ramps with nS steps of <freq> kHz. The frequency, <freq>, must be preceded by "F", e.g., F15.3 is a ramp step of 15.3 kHz.
RAMP nR,nS,i,Z S, IMA, MD	Specify nR Z-mode ramps with nS frequency index steps of i. The Z-mode does nR times nS pulse sets in a zig-zag pattern in which no frequencies are repeated. (See MAKEM, Section 1Ø.2.1.)
RAMP nR,nS,F<freq>,Z S, IMA, MD	Specify nR Z-mode ramps with nS steps of <freq> kHz. The frequency, <freq> must be preceded by "F", e.g., F12 is a ramp step of 12 kHz. All forms of the RAMP command are rejected with ERR 14, if given for a W-class sounding.
RANGE [n] P	Control Sweep-class range testing in PICKER. If n is zero or missing, no range testing is done. Otherwise, the range limits established for frequency n, by the RNGE command, are used by PICKER to exclude echoes outside the assigned range. See Section 3.1Ø.
RFAT n[,m] S, IMA	Specify Sweep-class receiver RF attenuation(s); use m for channel 2 if different from channel 1. The command is ignored unless the channel(s) are under manual gain control. To set the base RF attenuations, use command AGC.
RFAT nF,n[,m] S, IMA	Specify receiver RF attenuation(s) for T- or W-class frequency nF. The command is ignored unless the channel(s) are under manual gain control.
RF n[,m] S, SPC	Change the Sweep-class receiver attenuation(s) during a sounding. The command is ignored unless the channel(s) are under manual gain conrol.
RF nF,n[,m] S, SPC	Change the receiver attenuation(s) for T- or W-class frequency nF during a sounding. The command is ignored if the channel(s) are under AGC.

RNGE nF,rmin,rmax

RNGE nF,rmin,rmax $\overline{S}$, IMA, MD	Define the minimum and maximum ranges, in kilometers, for frequency nF of a T- or W-class sounding mode. (See command F nF.) The default range values are Ø to 5ØØ km for T-class modes and Ø to 1ØØØ for W-class. RNGE can also be used to set up the RANGE command to PICKER for S-class modes.
RUN $\overline{S}$, IMA	Immediately start the countdown followed by the production of a sounding of the current mode. The countdown may be set by CDN to 1 through 19 seconds.
RUN mm $\overline{S}$, IMA	Schedule a sounding of the current mode to run at mm minutes past the current hour displayed on the TSG clock. The n-second countdown will begin n seconds before mm minutes past the current hour.
RUN mm:ss $\overline{S}$, IMA	Schedule a sounding of the current mode to run at mm minutes and ss seconds past the current hour. The n-second countdown, set by CDN, will begin n seconds before the specified time.

If a time-specific RUN command is given, it must be given at least n seconds before the specified time to allow for the countdown, otherwise ERR 3 results. Seconds must be in the range Ø through 59; minutes (mm) may range from Ø through 546. Thus, a sounding can be scheduled to run up to 9.1 hours after the start of the current hour (9.1 hours is 3276Ø seconds; 32767 is the largest positive integer than can be expressed in a 16-bit half-word).

For example, if the TSG clock reads 1Ø:15:37, the command RUN 12Ø will result in scheduling a sounding to run at 12:ØØ:ØØ, or 12Ø minutes past the start of the current hour 1Ø; RUN 16:7 will run a sounding at 1Ø:16:Ø7; If CDN = 1Ø seconds, RUN 15:45 would be rejected as an illegal start time, with ERR 3.

RUN id[time] S, IMA	Run the sounding specified by sounding mode id at the time given. The new current sounding mode is id. The separator between id and time must be a blank.
RUN task S, IMA	Run CALL (support, diagnostic, or analysis) task in the GRAPHER partition. (See Section 1Ø.2.) Note that RUN task takes effect immediately, whereas CALL task takes effect at run-time.
RXANT n S, IMA, MD	Specify the receiving antenna field to be used for this sounding mode. n is in the range 1 through 1Ø. (See LOCAL, Sections 7.6 and 1Ø.2.2.)
SABORT A	Abort the schedule initiated with a previous SRUN command. If a sounding is in progress, it will continue unless it is terminated with the AB command. If the Scheduler (ANALYSER) is not active, the SABORT command is ignored. MANAGER also sends the SABORT command to the Scheduler if an error is detected by command TAPE, if the end-of-reel is sensed during a tape write, or if a single sounding overflows file IONOSOND.SAV.
SAVE [m] M	Save the next m soundings on disc by writing them in order after the current sounding. The data are packed into the LU4 buffer, normally IONOSOND.SAV in compact format (Section 4.12). MANAGER starts in this mode and remains in it until this command sets a count or a DROP is executed. MANAGER enters the DROP mode when the SAVE countdown ends. If m = Ø or is missing, it is defined as 1. If it equals or exceeds 9999, MANAGER remains in the SAVE mode without counting down.
SFRQ f S, IMA, MD	Set the starting frequency for a Sweep-class sounding mode in kHz. For example, SFRQ 25ØØ sets the starting frequency to 2.5 MHz. See the NOTE under EFRQ about protected frequencies and indexing.
SLIST xxx A	List schedule xxx to the console. The schedule identifier, xxx, is that specified by command SSAVE of CALL task SCHED (Sect. 1Ø.2.5). Command SLIST is rejected if a sounding is in progress, or if the Scheduler (ANALYSER) is active. See com-

mand SRUN xxx,n (below) for error messages.

SN n
S, IMA

Tell SOUNDER to run n soundings in an SPH schedule (below) and then wait for more instructions. Command SN sets SCT.SN (Section 7.1) to n. After n soundings have been run, SCT.SN becomes -1, the default condition under which an SPH schedule runs until ABorted. Satisfying SN has the same effect of canceling SOUNDER's debugging commands as does the command AB. (See command NO A, above.)

SORT [n]
M, Tape.

If n = Ø or is missing, no sort to tape (default). If n > Ø, zero LU2 IONOSOND, wait for sounding data to be written on LU2 in ascending frequency order, and copy the non-zero sectors of LU2 onto tape.

SORT can be modified by an assembly parameter to speed the disk writing process. When writing large blocks of data, such as the seven-sector, 5Ø echo blocks of partial reflection data, the blocks are written directly to an intermediate file, and packed later. This mode is much faster on an 8/16, where the controller can write large blocks.

SPH n
S, IMA

Tell SOUNDER to run n soundings per hour. If a SPH schedule is interrupted with AB, it may be restarted on schedule with CONT.

SRUN xxx
A

Execute commands from schedule xxx. The first time-specific RUN command will get its argument from the first string-substitution table entry. (See CALL task SCHED, command *, Sect. 1Ø.2.5). See the next command for error messages from the Scheduler.

SRUN xxx,n
A

Execute schedule xxx, but use the nth string-substitution table entry for the first RUN *.

The SRUN command is rejected if a sounding is in progress, or if the Scheduler (ANALYSER) is already active.

A schedule just executed, or terminated with SABORT (above), may be repeated by typing SRUN or SRUN xxx,n.

Errors resulting from SLIST and SRUN are listed below.

ERR 1 Schedule xxx does not exist.
ERR 2 The Scheduler is active.
ERR 3 Time-table entry is invalid (SRUN only).
ERR 4 Sounding in progress.

STEP i S, IMA, MD	Step Sweep-class base frequencies by i increments of frequency index.
STEP F<freq> S, IMA, MD	Step Sweep-class frequency by <freq> kHz. The frequency, <freq>, must be preceded by "F", e.g., F123.4 means step by 123.4 kHz.
SWAP OS mod.	Write the current user memory onto file SWAPFILE.ROL. if a swap has occurred previously, the user memory is read from the other half of SWAPFILE.ROL; if not, user memory remains unchanged. Swap time-out is set to Ø for no time out. See Section 11.11. This command is given to the MT2 OS only.
SYS S, IMA	Display the HF Radar system status and error counters. This display is also called at startup and by command TICK, below.
TAPE [n] M, Tape.	If n is zero or missing, open tape. If n = 1, open tape, skip to double EOF, and backspace over the second EOF. The tape is positioned after a single EOF or BOT. See also CLOSE, FORMAT and SORT.
TICK S, IMA	Time the four TSG pace intervals by running at each of them for one second. The HARDware tick displayed is in milliseconds as measured by TICK; the HARD tick is rounded up to the next multiple of 1Ø ms for the SOFTware tick. (Time is reckoned in SOUNDER in 1Ø ms intervals.)
TNEW M	Rewind LU4 and restart at sector 1. This restarts the LU4 file after a COPY for subsequent writing of soundings on disc in the DROP or SAVE modes. See Section 4.7 for details.

TPR
S, IMA

Run dummy sounding from previously recorded EKO data. The data are read from a magnetic tape normally written by CALL task TENUS (Section 1Ø.3.1). The SCT and the SCT.FTBL frequency table are established from tape, and after the count-down, SOUNDER reads the PCT/EKO buffer records and sends SØ2 Q-blocks to PICKER as it would during normal sounding. The sounding mode is set from SCT.MODE (Section 7.1) as read from tape. When the tape EOF is sensed, SOUNDER sends the SØ6 Q-block to PICKER, and recovers the SCT that was current when TPR was issued.

During a TPR, SOUNDER does nothing but read tape and send Q-blocks to PICKER. The other tasks function normally, although tape recording must not be specified for MANAGER (see the TAPE command). During TPR, SOUNDER rejects all advice except AB, which command sends the SØ6 Q-block to PICKER and causes the tape to skip to an EOF.

TSGC n[,m,...]
S, IMA, MD

Specify the pulse-to-pulse sequence of hexadecimal TSG program command bytes and, by inference, the associated TSG range-Ø correction, FEP program, etc. TSGC is normally initialized to 86 hex, specifying 1ØØ pulses per second and a repeat count of 1 for scenario number 6, but may be set to any legal TSG command byte. Odd numbered scenarios are rejected with ERR 4, unless specified for a W-class sounding mode. See the Technical Manual for the TSG, Table 5.2, page 5-41, for descriptions of current TSG programs. Also, see Chapter 12 of this document.

The standard TSGMEM files are TSGMEM.TWO and TSGMEM.TOO. These files are used with TSG PROMs FØ1 and B12 (or BWØ8), respectively. (See Appendix A, dated 1983 November, for further details.) Both files use scenario 2 to invoke EQR processing, 4 for SNDR, and 6 for FEP PICKER.

TXANT f
S, IMA, MD

Specify a frequency for a Sweep-class sounding at which the transmitting antenna changes from Ø to 1. (Not fully implemented.)

TXANT n
S, IMA, MD

Specify transmitting antenna Ø or 1 for a T- or W-class sounding. (Not fully implemented)

TXOF
S, IMA

Turn off the high power transmitter. The high power TX filaments and high voltage are shut down immediately, the dummy load is selected, and and the transmitter is fully attenuated. TXOF can be issued at any time.

TX
S, SPC

The command TX is used to shut down the transmitter during a sounding.

TXON
S, IMA

Turn on the high power transmitter. The high power TX ON sequence turns on the filaments and initiates a 9Ø-second filament warm-up period during which the transmitter high voltage cannot be turned on (see HVON). TXON selects maximum transmitter attenuation and preserves the state of the dummy load (DL).

NOTE: When the HF Radar is first started, maximum transmitter attenuation is put in and the transmitter dummy load is selected, but the current states of the filaments and the transmitter high voltage are preserved.

TXPH n[,m,...]
S, IMA, MD

Specify the pulse-to-pulse sequency of special (MPI) transmitter phasing. (Not fully implemented.)

UDIO d,n[,m,.]
S, IMA

Specify the palette extension, or the pulse-to-pulse sequence, for user defined DIO number d. If only UDIO d is given, the current palette extension for DIO d is displayed. (See DIOGEN, Section 1Ø.2.4.)

VUF n
S, IMA

Run a 3Ø-second single-frequency test sounding at the nth K-mode frequency (see F n) with NFIL NOP periods between individual pulse sets. No recording on disc or tape is made of the sounding. The system reverts to the command mode and sounding display at the end of the sounding.

Command VUF is not yet implemented in P2.Ø. (See V-mode under MAKEM, Section 1Ø.2.1.)

9. OPERATION

James R. Winkelman

This chapter begins with a tutorial for new users. The order of its steps allows for practice and introduces tape handling after everything else is going. Section 9.2 discusses emergencies for practiced as well as new users. Section 9.3 gives a short, practical operating sequence with reference to the tutorial steps for details.

9.1. How to Run a Sounding: Tutorial

The operating sequence step numbers have the prefix T for tutorial and to provide reference for Section 9.3.

T1 Power on — Verify that the power is on. Every installation will have its own power-up sequence.

Hints: plug the plugs into the wall; open the back of the computer and receiver racks and snap the switches on the plug assemblies; there may be one plug assembly on each side of the rack (four in all). The switch for the terminal is on the pedestal under the keyboard (use Braille or a flashlight). On the NSF system, the terminal is left on but plugged into one of the plug assemblies turned on earlier. Newer systems have a single switch on the front of each rack.

T2 Start Disc — 7/16: depress the left button (labeled START STOP) on the blue thing in the bottom of the computer rack.

8/16: Rock POWER ON switch (lower right) to ON and it will light. Rock the RUN-LOAD switch (lower center) to RUN.

Have a cup of coffee (40 - 1000 seconds). Eventually the ready light will go on: you must wait for it. Anything typed during disc not ready causes a system crash.

Comment on 8/16 Interdata discs:
This disc system has internal checks on disc speed and temperature. If the disc was cold when started, it can take a long time before the READY

light goes on. It takes even longer for the PROTect light to go out when you switch off the write protection. Once the system is stable, the protect switches can be simply switched on and off.

T3 Screen

Clear the screen on the 4Ø12 terminal. The screen comes on slowly to full brightness and should be cleared by depressing the RESET PAGE key on the upper left of the keyboard. This key can be pressed at any time without affecting the system. After pressing the key, wait for the flashing prompt before continuing typing.

T4 Switches

Check the clock on the TSG panel. Set the time, if necessary. Remember that the clock should not be left in the SET mode.

Check the switches on the frequency synthesizer. The LINE switch should be ON, and the CONTROL switch should be REMOTE.

You might determine that the disc enable switch, disc selection switch, and the boot load select switch are correct.

T5 Unprotect

To run a sounding, the bottom (fixed) disc must be unprotected. To run some of the CALL tasks (Section 1Ø.1), the top (cartridge) disc must also be unprotected.

You have two choices:
a) Unprotect both discs: 7/16 buttons out, 8/16 buttons bottom edge in. The lights behind the switches will go out on the 7/16 immediately, and on the 8/16 sooner or later.
b) Go to step T6 and then at some time before typing HFRADAR (T7), set the switches as in a) above and type:

```
MARK DSC1:,ON
MARK DSC2:,ON
```

Side issue.
The bottom disc is mounted permanently in the unit and is often referred to as "fixed". Our system refers to it as device DSC2:. When it is marked

T5 Unprotect

on, the name of the data volume recorded on DSC2: is displayed (SYS1 in this case). In referring to the device (discheck, mark on, etc.), use the device name DSC2:. In referring to the data or files, use the volume name SYS1.

The removable cartridge (upper disc) is device DSC1:. To run soundings the cartridge SYSC must be mounted; the gummed label on the cartridge is the same as its volume name.

The names and statuses of devices may be determined by typing:

DISPLAY DEVICE

The devices DSCn: will list as OFF or the volume name will appear. If the name is followed by PROT, the disc may not be written upon without marking it on (T5).

T6 INI

Press the INI button on the right of the INTERDATA console momentarily. The wait light will go out and 10 - 12 seconds later a message of the following form will appear on the screen of the 4012:

```
OS16 MT2 W3.024
DSC2:   SYS1
** START OS16WD08 SYSTEM (8/16)
```

The third line was printed by the start-up CSS: STARTOSW.D08, when system D08 was booted.

T7 Initialize

Type the following commands:

HFRADAR

The marking on of protected discs must precede HFRADAR.

Relax! The file HFRADAR will set partitions, load the tasks, assign files, and finally start COUNSEL. Eventually all tasks will be logged in and the FEP and TSG loaded. When this is all done, the prompt *COUNSEL> appears on the screen. It's time for you to do something.

T8 Sounder — Type any sounder commands desired, waiting for a prompt before typing the next (some are slow). Delay tape commands until T11 and T12 while practicing.

T9 RUN — Type DLOF and RUN to run a sounding. Additionally, TXON can be typed to start the filaments warming.

Personal bias (JRW): leave the transmitter off during these early practice sessions and use a KEEP 2 to get noise on the plot. It's not pretty, but very quiet for the neighbors.

T1Ø Tape use — To use tape:

Make certain there is a write ring in the groove in the back of the tape. Mount the tape following the directions in the lower left of the 8/16 drive or the upper right of the door on the 7/16.

Nothing has to be opened. The tape simply slides into slots and goes around pulleys. Wrap the end of the tape two or three times around the take-up reel, keep taut, and keep finger out of hole while pressing the LOAD button. Keep trying. The ON-OFF switch is at the lower left of a 7/16 unit; the POWER button corresponds on an 8/16.

The tape will move to the load point (a bright spot on tape). If the tape keeps moving for a long time, press REWIND.

Press ON LINE. Close the door. Relax.

To remove a tape, press ON LINE on a 7/16 to get it off line, or RESET on an 8/16. Press REWIND. After the tape is rewound, press REWIND again. Remove the tape. Remove the write ring if desired.

T11 Record on Tape — Recording on tape requires a mounted tape, on-line, and write enable. If the tape is positioned properly, type TAPE to make normal format recordings. If you wish to add data to the end of an existing tape and the tape is not at the correct position from previous recordings, type

TAPE 1.

T12 Terminate When all is done, type:

CANHF

This command is followed by a group of END OF TASK's.

If all the tasks did not appear in the list of 'END OF TASK', then the tasks which did not appear were not cancelled (they might have been paused). The CSS files CANHF and CANJRW might cancel them. If not, type DISPLAY MEMORY to list the surviving tasks.

For each of them type:

TASK taskname;CANCEL

T13 Protect When done using the system, type:

MARK DSC1:,ON,PROT
,OFF if removable disc to be changed
MARK DSC2:,ON,PROT

The discs are protected to make certain that no files were left open. Set switches to protect.

T14 Power off When turning off the system completely, release the START STOP switch on the 7/16 or move the RUN LOAD switch to LOAD and wait until the start switch goes out or the LOAD light comes on. This removes power from the disc gently and is important.

T15 Synthesizer Switch the LINE switch on the frequency synthesizer to STDBY, to use the standby power source. Clear the Tektronix screen before turning the terminal off.

9.2. Emergencies

What to do until the doctor comes. For both experienced and new users, some suggestions on what to do when nothing happens or something goes wrong.

Symptom: you type and nothing happens, not even the echoing of characters. Sometimes the display stops counting seconds. This is termed a *CRASH* and happens to everyone occasionally.

After a crash, press INI. If SYSC were unprotected before the crash, type DSC1 (which closes all open files on DSC1:). Do a V SYSC followed by a DSC2. A disc check cannot be run automatically from its own volume.

You have started the system and are getting many error messages. Nothing is running quite right. Someone probably shut it off wihtout closing down properly. Try DSC1 and DSC2 as in the paragraph above.

If none of the above work, get help! I don't know what to do either!

9.3. Practical Operating Sequence

The following operating sequence makes a better use of time than the tutorial sequence in Section 9.1. This practical sequence serves as a general checklist of necessary processes and necessary times to do optional processes. Reference to details in Section 9.1 is made by the T-step number.

1 Power	Turn on both switches in the digital cabinet and both switches in the analog cabinet (T1).
2 Disc	Turn on the disc (T2).
3 Screen	Clear the 4Ø12 screen (T3). Switch synthesizer LINE switch ON (T4).
4 Tape & TX	While the disc comes up to speed: mount a tape (T1Ø), turn on the transmitter, or get something to drink (local option).
5 Unprotect	After the disc READY light is on, unprotect the fixed disc (T5). The rightmost switch on the disc must be out and its light off.

6 DSC & File	Set DSC switch down and file switch to 1.
7 INI	Press the INI button on the Interdata console (T6). After the system has been loaded, a response appears on the 4012.
8 Initialize	Type (T7):

MARK DSC2:,ON
HFRADAR

9 *COUNSEL>	Relax until the *COUNSEL> prompt appears and then enter desired commands (T7, T8).
10 RUN	Type RUN after all advice has been given to the programs.
11 High Power	Type TXON to warm up the filaments of the high power transmitter. The HV HOLD light (third row, first light) will go on, and then will go out 90 seconds later. When the HV HOLD light goes out, type HVON to turn of the high voltage. Check that things look right (no smoke?), and type DLOF to switch the transmitter output from the dummy load to the antenna.

The following commands are required to close down the system (T12, T13).

89 CANHF	CANHF	
90 Protect	MARK DSC2:,ON,P MARK DSC1:,ON,P	OFF if disc to be changed
91 Disc off	Turn off the disc unit and switch the synthesizer to standby.	
92 Power off	Clear the Tektronix screen before turning the terminal off. When the disc ready light goes out, you may turn off power switches as required. Enjoy the silence.	

9.4. Assembly and Loading

In order to make a change in the HF Radar software, the files must be edited, assembled, established as a task, and finally loaded and started. This sections provides information on these procedures.

CSS files CSS files appear to the casual user to be the same as system commands. They may not be abbreviated and are actually lists of commands. They are widely used to save typing and to obviate the need to remember all of the details of running certain programs.

The CSS files mentioned in the first three sections include HFRADAR, CANHF, CANJRW, and DSC1.

The editor is called by typing EDIT. Get the file to be edited and SAVE the edited version. The names are associated with the listings. The assembly files are extension .CAL, which get assembled into .OBJ and established into .TSK, the actual tasks.

Current .CAL files are:

COUNSEL	GRAKI	SNDRADVC	SNDRINIT	SNDRO004	SNDRO102
PICKER	GRSKY	SNDREENT	SNDRMSUB	SNDRO005	SNDRO103
MANAGER	GRAMPH	SNDRNXTF	SNDRSTUP	SNDRO006	SNDRO104
ASKED	GAMPHT	SNDRPIPE	SNDRSYST	SNDRO007	SNDRO105
GRAFPRO	GRAFFT	SNDRTBLS	SNDRO000	SNDRO008	SNDRO300
GRAFRAM	GRASIN	SNDRBOOT	SNDRO001	SNDRO009	SNDRO301
GRAFBUF	GRHVL	SNDRDISP	SNDRO002	SNDRO100	SNDRO302
GRION	GRXYT	SNDRDSUB	SNDRO003	SNDRO101	

Assembly Option CROSS for obtaining cross reference lists is always present. The assembler is invoked by CAL followed by the name, listing, and options. CAL16 is faster and has the same call. Examples:

```
CAL PICKER
```

PICKER.CAL is assembled into PICKER.OBJ with listing on PICKER.LST.

```
CAL MANAGER,,NLSTC
```

MANAGER.CAL is assembled into MANAGER.OBJ with listing on MANAGER.LST with option NLSTC.

CAL COUNSEL,MAG:

COUNSEL.CAL is assembled into COUNSEL.OBJ with listing on magnetic tape.

TET

There are three Task Establishment Tasks: TETGRAF, TETSNDR, and TET. They are invoked with a name, bias, and priority.

The priorities have been assigned as follows:

COUNSEL	43
SOUNDER	6Ø
PICKER	7Ø
MANAGER	8Ø
GRAPHER	9Ø
ANALYSER	11Ø
.BG	128

The biases are found from the CSS file that will load and run the tasks, in this case HFRADAR (see below). The tasks may load in any order, but they must find a partition with exactly the same starting address as their bias at TET time.

Examples:

```
TETGRAF GRAPHER,6CØØ,9Ø
```

combines the twelve GRAPHER OBJ files and creates GRAPHER.TSK, GRAPHER.LMO, and GRAPHER.MAP.

```
TETSNDR SOUNDER,93ØØ,6Ø
```

combines the 31 SOUNDER OBJ files and creates SOUNDER.TSK, SOUNDER.OVL, and SOUNDER.MAP.

```
TET COUNSEL,5F8Ø,43
```

COUNSEL.OBJ to COUNSEL.TSK and COUNSEL.MAP.

JTET

JTET.CSS runs the same as TET.CSS, except that if a file of the name.TET exists, the .TET file is used instead of the command file built by TET.CSS. For example, if a file COUNSEL.TET exists, JTET COUNSEL will build COUNSEL.TSK.

CALTET — CALTET.CSS combines the CAL16.CSS and JTET.CSS functions. CALTET COUNSEL compiles COUNSEL.CAL, then uses COUNSEL.TET to produce the files COUNSEL.OBJ, COUNSEL.LST, COUNSEL.TSK, and COUNSEL.MAP.

Load and run — HFRADAR.CSS controls loading. The six paritions are set. Each partition must start at a bias which was assigned during TET. The distance to the next partition must be long enough to hold the task to be loaded into the partition.

The load commands specify both the name assigned to the task and the name of the file to be loaded. The task names appear many times in the code and should not be changed.

Figure 9.1 is a listing of HFRADAR.CSS. Note the LOAD ANALYSER, ASKED.TSK.

Some of the tasks require assignments of logical units for both flexibility and to save space in the task.

GRAPHER needs a contiguous file to save the plot buffer of the latest ionogram. This file, GRAPHER.PLT must be at least 26 sectors long in the current version:

```
ALLOCATE GRAPHER.PLT,CO,26
```

MANAGER requires two or three files:

```
LU2, optionally IONOSOND.
LU4, normally IONOSOND.SAV
LU5, normally LOCAL.TWO
```

LU2 is used for sorting Sweep-class soundings. It is not used, otherwise. Its length is defined in Section 4.12 as LAST.

Sample lengths for IONOSOND.:

```
3000 kHz to 15000 kHz (930,STEP=2),(1860,STEP=1)
100 kHz to 30000 kHz (3292,STEP=2),(6584,STEP=1)
```

If the disc is running short of space, you must limit STEP, the width of the scan, or the number of echoes. The above estimates are halved for a maximum of 6 echoes per P02 block rather than 8.

```
TASK .BG
$COPY
*LOAD: 1/5F8Ø 2/6CØØ 3/93ØØ 4/CEØØ 5/D5FØ 6/DFAØ .SYS/F91Ø
$NOCOPY
SET PARTITION 1/DFAØ,.SYS/F91Ø
SET PARTITION 6/DFAØ,5/D5FØ,4/CEØØ,3/93ØØ,2/6CØØ,1/5F8Ø
LOAD COUNSEL
LOAD SOUNDER
TASK SOUNDER
$IFNN @1
ASSIGN 7,SYS1:SMODES.@1,SRW
$ELSE
ASSIGN 7,SYS1:SMODES.TWO,SRW
$ENDC
LOAD PICKER
LOAD MANAGER
TASK MANAGER
ASSIGN 4,SYS1:IONOSOND.SAV
ASSIGN 5,SYS1:LOCAL.TWO,SRO
LOAD GRAPHER
LOAD ANALYSER,ASKED
TASK COUNSEL
ASSIGN 1,CON:
*The above CON: may be replaced by a command file.
ASSIGN 2,SYS1:LOG.
ASSIGN 3,NULL:
ASSIGN 4,NULL:
ASSIGN 5,CON:
START
$EXIT
*  HFRADAR.CSS  Ø8/29/83
```

Figure 9.1. HFRADAR.CSS

LU4 is used to save old soundings. With compact tape, LU4 must be large enough to hold the entire sounding. LU2 is only required if the command SORT is given.

I recommend starting with a clean SYS1:, adding an OS16 system or two, a copy routine, a few CSS's, the necessary .TSK's, and then defining GRAPHER.PLT and IONOSOND. A FILE command will give both unused space and maximum contiguous space allowable for files. Allocate the largest IONOSOND.SAV allowable while leaving perhaps 1ØØ sectors unused.

9.5. Disc Compress, Copy, and List

This section collects a miscellany of useful, related operations.

DISCMP There are six disc compresses:

DISCMP1M.CSS	DSC1: to MAG:
DISCMPM1.CSS	MAG: to DSC1:
DISCMP2M.CSS	DSC2: to MAG:
DISCMPM2.CSS	MAG: to DSC2:
DISCMP12.CSS	DSC1: to DSC2:
DISCMP21.CSS	DSC2: to DSC1:

These CSS's copy a complete disc to or from another disc or magnetic tape. They are used to save a disc so that it can be recovered later. They are also used to copy a new set of files from another installation onto a disc. If a disc has insufficient contiguous sectors, the disc may be compressed to tape and back to disc.

The buffers used are 12k in size. This long buffer runs faster and more reliably than a shorter one.

I know of no way to recover from a tape error.

To tape:
Mount a tape at load point and run DISCMP1M or DISCMP2M. The disc being copied must be protected.

From tape:
No file name on disc can match a file name on tape. This example starts from an empty disc:

```
MARK DSC1:,OF
INIT DSC1:,SYSB
DSC1
DISCMPM1
```

This process takes about an hour.

The DSC1 checks for sectors that may be hard to read. It is more thorough than READCHECK in INIT.

LIS

This operation lists lines on the 4Ø12 (CON:) and has the following arguments:

LIS.CSS name,F,L

Lines F through L are displayed. The line number and date are printed at the top of each page.

If F is omitted, listing starts with line 1.

If L is omitted, the listing ends with line 6553Ø.

Any separator may be used between F and L.

Lines longer than 72 characters are continued on the next line preceded by ">-->".

TYP

This is the same as LIS for the teletype (TTY:).

CPY

This is a replacement program for OS COPY for many of the normal copy needs. It has the following arguments:

CPY name1,name2

A file called name2 is made that is the same as name1, that is, with the same record size, type (contiguous or indexed), etc. If both old and new files are on disc, the new file will be contiguous if the old file had record length 256.

COP

COP copies a file from one disc to the other:

COP name,from,to

The volume names are assumed to be of the form SYSx. Any separators may take the place of the commas.

COP EXEC.CAL,D,1

copies the file SYSD:EXEC.CAL to SYS1:EXEC.CAL.

9.6. Saving and Restoring the Disc

The following routines constitute an alternative to DISCMP, which is extremely sensitive to tape condition and is slow.

The n with the first three commands described below is 1 for DSC1:, the top disc (removable cartridge), or 2 for DSC2:, the bottom (fixed) disc.

SAVDISK n
GETDISK n

SAVDISK copies the disc to tape, one record for each of the 408 tracks. Additional identical records are written for any record with a parity error during writing.

GETDISK copies the tape to a disc, allowing for duplicate records.

The tape format for each file, one disc to a file, is: end-of-file (EOF), 40-byte label, and 408 or more 12290-byte data records. The tape should be positioned ahead of the EOF. For example, to GETDISK the second file on a tape, type:

```
FF MAG:
FF MAG:
BF MAG:
```

The first FF will move a few inches over the EOF at the start of the first file. The second FF will skip the file (about 3 minutes). The BF will only move the tape about an inch.

Before writing the tape, SAVDISK will ask for a label. Type in up to 40 characters. For example:

```
SYSD 06/24/79 JRWINKELMAN.
```

GETDISK will display the label. If the file is correct, type Y. If the file is not the one desired, type N. The program will pause and you may reposition the tape using FF and BF commands. Type CON to continue the program and it will display the new label.

SAVDISK and GETDISK have some advantages over Interdata's DISCOMP: they are faster, they allow multiple files on a tape, they identify the files, preserve the created and used dates, and the have good error recovery. They have the disadvantage of not compressing the disc.

COMDISK n — COMDISK does a partial compression of the disc without the use of tape. The number of sectors required for the data is determined. Any sector associated with an indexed file which is above the necessary boundary is copied to an unused sector below that boundary. Only contiguous files remain above the boundary. Do not cancel this program, or otherwise interfere with its operation, or files may be lost.

JC file name — Copies files to another location on disc. JC ends with an allocate-size error that resets the file manager to assign the next file as low on disc as possible.

This routine is normally used to copy contiguous files as low on the disc as they will fit, leaving the top of disc as empty (long contiguous space) as possible. It moves continguous files, not handled by COMDISK, under the supervision of an operator.

10. CALL AND UTILITY SOFTWARE

David C. Walden

10.1. Introduction [DCW]

An effort has been made to provide the HF Radar users with a set of utility, support, and diagnostic programs closely associated with the basic HF Radar software. These programs, or tasks, provide information about the health of both the hardware and the sounding software; set up tables of information to be used later by the sounding tasks (SOUNDER, PICKER, MANAGER, GRAPHER, AND ANALYSER); provide near-realtime, or offline review of the sounding data in tabular and graphic form; and tape record data directly from the FEP. To make these tasks easily available to the user, SOUNDER loads them into GRAPHER's memory partition, then after they have executed, reloads and restarts GRAPHER, leaving memory intact for normal sounding. In P1.0 these tasks were designated TEST tasks; in P2.0, their function has been expanded beyond testing, and the tasks are called CALL tasks. CALL tasks can be run in one of two ways: when the sounding tasks are idle, the SOUNDER command

RUN <name> (See Section 10.2.)

is given. The task name, <name>, must contain more than two characters to distinguish it from a sounding mode identifier. The above command is the exact equivalent of the P1.0 command, TEST <name>. Most of the Idle-mode CALL tasks can also run by themselves as stand-alone tasks. See Section 10.2 for details.

CALL tasks that occupy GRAPHER's partition during a sounding and that bypass the standard data flow through PICKER, MANAGER, GRAPHER, and ANALYSER (e.g. TENUS) are put on call to be loaded and run at sounding time with the SOUNDER command

CALL <name> (See Section 10.3.)

Thus, CALL task commands and functions are an extension of SOUNDER, and the time-consuming and irritating necessity of canceling the HF Radar software in order to test the system, etc., is obviated. Furthermore, testing can be done under conditions that most nearly resemble those of actual sounding, and alternative data recording modes are made available.

There are a number of SEL tasks that have not been incorporated into the CALL task system. These will be referred to as utility tasks to distinguish them from the tasks that are executable with the RUN or CALL commands. Instructions for running the utility tasks are given in Section 10.4.

All CALL task commands end with a carriage return, designated by "(cr)". It is usually omitted from the command descriptions, but appears whenever it is necessary to distinguish forms of a command and when it is itself a command. Tasks replacing GRAPHER during a sounding (CALL <name> tasks) generally do not accept command input, in order that they may be scheduled.

Mandatory command characters are underlined. Optional arguments appear in square brackets.

CALL and utility tasks display the prompt character ">" whenever they require command input from the console. If a task requests further input, the request itself is the prompt.

10.2. Idle-mode CALL Tasks [DCW]

Between soundings, and if GRAPHER has nothing to do, CALL tasks can be loaded into GRAPHER's partition and executed. Execution of these tasks is started with the RUN command, as noted above; all Idle-mode CALL tasks are terminated with the END command, at which time GRAPHER is reinstated and normal sounding can resume.

If any of the tasks under Section 10.2 need to be reassembled, they must also be reTETed. Tasks TPLOT, SPLOT, and DIO, in order to run stand-alone, are TETed with SOUNDER's reentrant subroutines, thus:

TETRUN <name>,<bias>,65,WRCON,SNDREENT

The bias is that of GRAPHER's memory partition and is currently 6C00; 65 is the CALL task priority. None of the other tasks require SNDREENT and are TETed thus:

TETRUN <name>,<bias>,65

Most of the Idle-mode CALL tasks can also run stand-alone. Those that cannot are noted in the following sections. To run a CALL task when no sounding or other tasks are active, type:

V SYSC
RUN <name>

The CALL tasks are stored on volume SYSC. The RUN command to the MT2 OS invokes disc file RUN.CSS, which sets memory partitions, loads task <name>, and starts it executing.

The prompt character for all Idle-mode CALL tasks is ">". Very few explicit error messages are given for erroneous command input; in general, a <bad command>, results in "CAN'T DO <bad command>".

1Ø.2.1. MAKEM 2-Ø1 [DCW]

(For details about changes made to MAKEM 2-ØØ, refer to Appendix A, dated 1983 November.)

Task MAKEM is used to construct the permanent sounding modes from which all user-defined modes ultimately are made. The permanent modes are identified by a single alphabetic character; user modes have identifiers of two characters, the first of which must be non-numeric. MAKEM can also be used to modify user-defined modes. (There are no SOUNDER commands to modify MD.TMDE, MD.MAP, MD.ADLY, etc. See Chapter 7 for details.) All sounding modes are stored on disc files SYS1:SMODES.xxx, which are 128 contiguous sectors long. These files are invoked with HFRADAR xxx, where xxx is the extension of the SMODES file on SYS1. If xxx is not specified, SYS1:SMODES.TWO is used. SYSC:SMODES.NEW is initialized at SEL to contain modes, I, K, B, Z, P, G, V, and W. These modes are described below. SYSC:SMODES.NEW can be copied to SYS1:SMODES.TWO to make it the system default. The first mode in the SMODES.NEW directory is I, and being first it is the default active mode when the HF Radar tasks are first loaded.

Individual installations may wish to use MAKEM to modify or delete any of these permanent modes, or to create new ones. A file SMODES.xxx has room for 1Ø permanent modes and 43 user modes. Each mode occupies 2 sectors; two sectors are reserved for the mode directory, which can be listed with SOUNDER's command, MDIR (Chapter 8). Two copies of each permanent mode are maintained. The first copy is created by MAKEM and is read, never written, by SOUNDER. The second copy is made from the first one at SOUNDER start up. Thereafter it is updated, or written back to SMODES.xxx as are user modes: the currently active mode is updated whenever a different mode is accessed by RUN or MGET, or when CANHF is typed to COUNSEL. (See Chapter 8.) (Note that if CANHF is typed to the MT2 OS, the currently active mode in SOUNDER is <u>not</u> updated.) Thus a permanent mode always appears in its original form when the HF Radar software is first loaded, but changes made during a sounding session are preserved for the duration of that session, that is, until CANHF is typed.

The sounding modes are stored on disc as binary images of SOUNDER's table, SCT (Section 7.1). MAKEM has its own image of the SCT, labeled XSCT, with many values preset to defaults. Both SOUNDER's SCT and MAKEM's XSCT must conform to the SCT STRUC in SYSC:PCB.CAL. If the SCT STRUC is modified, all of the sounding and most of the CALL tasks must be reassembled and reTETed, and table XSCT in MAKEM must be modified accordingly. In principle multiple sets of the sounding and CALL tasks can exist with different SCT layouts. If this is done, compatibility can be guaranteed by making SCT.MREV, assembled in SOUNDER, match MD.MREV, assembled in MAKEM. SOUNDER's RUN and MGET commands reject a sounding mode (ERR 13) unless SCT.MREV = MD.MREV.

MAKEM task commands are:

NEW xxx — Create a new file, SYS1:SMODES.xxx or delete all modes from the file, if it already exists. If NEW can allocate a new 128-sector contiguous file, or if it finds an existing SYS1:SMODES.xxx, it replies with "NEW FILE" or OLD FILE". If the operator responds with (cr) only, NEW proceeds to set up the new file or clear the old file. Any other character followed by (cr) aborts the NEW command.

GET xxx — Get SYS1:SMODES.xxx for sounding mode editing. MAKEM starts up with SYS1:SMODES.TWO, the system default, as the active editing file.

CLASS a — Specify the class for a new permanent sounding mode. The argument, a, must be S or T. MAKEM builds only Sweep-class or Table-class modes automatically. (See MD.CTRL, Section 7.1.) CLASS initializes MAKEM's SCT image, setting certain defaults accordingly to sounding class. The class dependent defaults are:

FIELD	S	T
MD.TMDE	0	1
MD.CTRL	0	X'80'
MD.NSET	1	512
MD.NFIL	0	1
MD.PKLO	3000	1000
MD.PKHI	9000	3000
MD.MAP	"I"	"K"
MD.TXAN	2000	1

In addition, the 32-bit frequencies in SCT.FTBL are set to -1,0 (inactive) for T-class modes. For S-class modes they are set to 0,0 (active, 0 kHz), to allow SOUNDER's RNGE command to be used to set up PICKER's RANGE command. (See Chapter 8.)

MGET <id> Get old mode <id>, where <id> is one or two characters. The SCT image is read from SYS1:SMODES.xxx for editing. The edited SCT is written back to disc by MSAV, below.

MSAV <id> Save the current SCT image on SYS1:SMODES.xxx as mode <id>. The mode name may be one or two characters; the first must not be numeric. MSAV displays the message, "NEW MODE" or "OLD MODE". The operator confirms or aborts the MSAV command as described in NEW, above. The SCT image in MAKEM memory resulted from CLASS or MGET, above. If <id> is a single alphabetic character, both the permanent and the session default copies are written. In either case, mode <id> is added to the mode directory, if the mode is new.

Because the current mode in MAKEM memory is unaffected by the GET command, the command sequence

```
GET SM1      (Get SYS1:SMODES.SM1)
MGET U1      (Get user-defined mode, U1)
GET SM2      (Get SYS1:SMODES.SM2)
MSAV U1      (Save mode U1)
```

copies mode U1 from SMODES.SM1 to SMODES.SM2, leaving the original copy undisturbed. The mode name, U1, may be the same or different in the two directories and may also be one or two characters. Thus a user-defined mode can be elevated to permanent status.

Each mode directory entry is 8 bytes in length and contains the following:

OFFSET	CONTENTS
+0	1 or 2 character mode name, or 1, if the entry is empty in the permanent segment. In the user segment of the directory, an empty entry is flagged with 0. The end of the SMODES directory is sensed if the mode name is -1.
+2	16-bit creation date. Bits 0-8 contain the Julian day number, bits 9-15 contain the year - 1900 (0=MSB).
+4	16-bit date that mode was last run (like creation date).
+6	15-bit mode run count, modulo 32768, cleared to 0 by MSAV.

MDEL <id> Delete mode <id> from directory. The SCT image itself is not overwritten until its directory entry is used for a new mode.

MDIR List the SMODES directory.

END Terminate the MAKEM task.

The following commands are used to edit either a new permanent mode, created by CLASS, or on old permanent mode, gotten from disc by MGET:

d T:<text> Copy ASCII <text> into SCT image, starting at hex displacement, d, where SCT.MD < d < SCT.END. (See Section 7.1 or an assembly listing of SNDRTBLS for the hex displacements associated with SCT.MD labels.)

d B:<list> Copy <list> of hex 8-bit bytes into SCT image, starting at displacement, d.

d H:<list> Copy <list> of hex 16-bit 1/2-words into SCT image, starting at displacement, d.

d D:<list> Copy <list> of decimal 16-bit 1/2-words into SCT image, starting at displacement, d.

The permanent sounding modes established at SEL in disc file SYSC:SMODES.NEW are set up as follows:

I-mode is essentially like the default I-mode of P1.0. It conducts a sweep from 1.5 through 15 MHz with a frequency index step of 4, using the standard PICKER task (Section 3.1). The delta-frequency is 8 kHz. There are 4 pulses per pulse set; the antenna selection and delta F sequences are standard. (See Table 2.1.) All devices on the DIO bus are controlled as they were in P1.0, except that the RF and IF attenuators for the two receiver channels are controlled independently. Mode I is the prototype Sweep-class sounding mode.

K-mode is also essentially like the P1.0 K-mode and is the prototype Table-class sounding mode. It is initialized with all of the frequencies (0-9) inactive. Otherwise, K-mode is initialized like I-mode, above.

V-mode is exactly like K-mode except that MD.TMDE (and SCT.MODE) is 2 so that MANAGER does not save any data on disc or tape, if selected. We thus have a poor man's view-mode until SOUNDER's command VUF is implemented. Notice that once a quiet frequency has been found with V-mode, it must be copied to the user's K-mode in order that the data can be recorded. Use a sequence like:

```
MGET V
F0 12345.6
RUN          (Determine quiet frequency, but don't record)
MGET KU
F0 12345.6
RUN          (Record data from quiet frequency)
```

B-mode was developed by J. W. Wright, of CIRES, for the prototype Dynasonde sounder. In the P2.0 software it can be thought of as I-mode but with a ramp of frequencies based on each I-mode frequency. Each frequency ramp is repeated a number of times for good time resolution over the frequency band of the ramp. B-mode is set up as is I-mode above, but with a frequency index step of 24 and with a 6-frequency ramp every 24 frequency index steps. The frequency index step within the ramp is 4; each ramp is repeated 6 times, as if the SOUNDER command, RAMP 6,6,4,B had been given. B-mode, like I-mode, is a Sweep-class sounding mode and uses the standard PICKER task.

Z-mode is an enhancement of B-mode developed by Wright and M. L. V. Pittaway, of Brunel University, and provides the time and frequency resolution benefits of B-mode, but in 1/6th the time and with 1/6th the use of the HF spectrum, assuming 6 ramps with 6 steps/ramp. For this reason, it is anticipated that Z-mode will largly replace B-mode. The Z-mode ramp is derived as follows. Assume a sequence of 36 frequencies, numbered 1 through 36. They can be arranged in the following order:

1	12	13	24	25	36
2	11	14	23	26	35
3	10	15	22	27	34
4	9	16	21	28	33
5	8	17	20	29	32
6	7	18	19	30	31

Z-mode sounds on these frequencies from left to right and from top to bottom, i.e., 1, 12, 13, 24, 25, 36, 2, 11, etc. The above example would be invoked by the SOUNDER command, RAMP 6,6,1,Z. THe first 6 is the number of rows, the second 6 is the number of columns, and the 1 is the Z-ramp frequency index step. The "Z" is required to specify the Z-ramp as opposed to the B-ramp, described above. If STEP 36 were given, the next ramp sequence would be 37, 48, 49, 60, 61, 72, 38, 47, etc. Z-mode is initialized as if RAMP 6,6,4,Z had been given, which yeilds the above sequence with the column, or ramp, step being 4 instead of 1. The ramp-to-ramp step (STEP) is initialized to 144 (6*6*4). For further details, see The Z-ramp Algorithm, Section 2.7.1. Z-mode is a Sweep-class sounding mode and uses the standard PICKER task.

P-mode is included in anticipation of an experiment planned by G. W. Adams, of Utah State University, using the five-by-five partial reflection antenna array at Boot Lake, CO. For this reason, P-mode has a pulse set of 5. P-mode is initialized to use PICK5PPS (Section 3.1) which supports PIK processing at 4 pulses per set and PARTIAL at up to 5 pulses per set and 52 echoes per frequency. P-mode is a Table-class sounding mode; F0 is initialized to 2.666 MHz, the frequency at which the partial reflection array is tuned.

G-mode, like P-mode, is a Table-class sounding mode, and like P-mode saves all of the data from the FEP. G-mode is initialized to use the PICK5PPS configuration of PICKER (Section 3.1). Using PARTIAL processing, no peak-finding is done by PICK5PPS; each X,Y pair is considered to be an echo. In this respect, P- and G-mode function much like K. Bibl's Digisonde. It is anticipated that

G-mode will be used for spread-F studies.

W-mode is a variation on partial reflection modes G-mode and P-mode that utilizes the HF Radar's capability to switch TSG scenarios on a pulse-to-pulse basis. W-mode collects data over a range of 450 km, if the FEP-receiver ADCs are sampling every 10 microseconds, or over a range of 900 km, if sampling every 20 microseconds. W-mode thus offers a nearly instataneous picture of the D-, E-, and F-regions in partial reflection form. W-mode is a W-class sounding mode with MD.TMDE (and SCT.MODE) set to 7. W-class is distinguished from T-class by bit 2 of MD.CTRL (Section 7.1), which is 0 for T-class and 1 for W-class. This bit is tested by the command routines for TSGC, RAMP, and DELF. Otherwise, T- and W-class modes are functionally the same.

W-mode utilizes all six available TSG scenarios (2-7) and PICK6PPS (Section 3.1) for 6-pulse/set sounding in such a way that the FEP WRITE ENABLE windows for each of the 6 pulses of a pulse set combine to form an overall window containing 300 range bins, or X,Y pairs. Moreover, this combined window can be adjusted over a range of about 700 km (10 microsecond sampling) or about 1400 km (20 microsecond sampling) using the FEP 3 command (Chapter 8).

There are two important caveats that must be observed if W-mode and its derivitives are to be used. First of all, remember that SOUNDER adds 1 to a specified even-numbered scenario if a protected frequency is encountered. (See PSCONT, Section 2.9.1.2.) All but W-class soundings are prevented from transmitting on protected frequencies because command TSGC will not accept odd-numbered scenarios, and because scenarios 3, 5, and 7 are normally listen-only scenarios. A W-class, like a T-class sounding mode, is prevented from selecting a protected base frequency. However, because W-mode uses scenarios 2, 3, 4, 5, 6, and 7, the operator must be prevented from inadvertantly sounding on a protected frequency as the result of a ramp or delta F computation. For this reason DELF is initialized to 0 for W-mode, RAMP is initialized null, and commands DELF and RAMP are inappropriate (ERR 14) for W-mode. The default W-mode TSGC sequence is 2, 3, 4, 5, 6, 7, but any sequence of scenarios numbered 2-7 may be specified by TSGC (Chapter 8). Each scenario in the default sequence is initialized with a repeat of 2 at the 20 ms pace. The duration of each default 6-pulse set is thus 240 ms. Because of the large volume of data collected and the full use of SOUNDER's 6 EKO buffers, if NFIL = 0 a minimum repeat of 2 is necessary if pulses and echoes are not to be lost. (See NXTPLSE and GETBFR, Section 2.9.1.1.) If NFIL is large enough, however, the 10 ms pace may be selected. In addition, MANAGER's SORT option (fast disc write for 8/16's, Section 4.7) may be invoked. Finally, the 20 ms pace is absolutely required as a minimum if 20 microsecond sampling is invoked. The FEP program assumes that 509 X,Y pairs of data are acquired during each cycle, a process requiring 10.18 ms at the 20 microsecond sampling rate.

The second caveat is that the PICKER task assumes that the data acquisition for each pulse of a pulse set begins at a constant range, which it indeed does for all but W-class soundings. PICKER therefore assigns a single range value for the pulse set data. In practice this is the range computed for the final pulse of a given set and is the range used by GRAPHER's display routines. For W-class soundings, however, the TSG delay, or the time between the TX KEY and the FEP WRITE ENABLE window, varies from pulse to pulse within a pulse set, so the range value assigned by PICKER for a given echo applies only to the data from the final pulse of a set. However, all of the information needed to compute the ranges off-line is contained in the PCTs and the SCT. The palette index for a given pulse is in the low order 4 bits of PCT.PDIX (Section 7.2) of the PCT for that pulse. PCT.PDIX is used to index into SCT.PD (Section 7.5) in order to get the values PD.TDLY, the TSG delay, and PD.FEPC, the FEP command vector, for a given pulse, n, of a pulse set. PD.FEPC is used to determine the sampling rate of the FEP ADCs: if PD.FEPC(PCT.PDIX(n)).AND.64 = 64, the samples were taken every 10 microseconds; if 0, 20 microsecond sampling was in effect. The range in microseconds of the first X,Y data pair for a given pulse, n, of a set is then:

```
 5*Fk+PD.TDLY(PCT.PDIX(n))-MD.ADLY    (10 microsecond sampling)
10*Fk+PD.TDLY(PCT.PDIX(n))-MD.ADLY    (20 microsecond sampling)
```

where Fk is specified by FEP 3,Fc,Fk (Chapter 8) and is the index into FEP input memory at which the FEP program begins its processing. Fk is the fourth 16-bit value in MD.FEPP. MD.ADLY is the analog delay of the receiver/antenna system. PD.TDLY(PCT.PDIX(n)) implies that PD.TDLY is indexed by the nth PCT.PDIX, where n is the pulse number of a set. Notice that the 16-bit value in PD.TDLY may be positive or negative and that PCT.PDIX may be 0-7.

The range thus computed for the first X,Y data pair actually applies to the third X,Y pair of a pulse set. The first is the noise record; the second is the calibration data record. The ranges of successive X,Y data pairs are found by adding multiples of 10 or 20 microseconds to the initial range calculation.

Two TSG zero pages have been composed at SEL for running W-mode at 6 pulses/set and 52 echoes/frequency. (EPF 52 is effectively 50, because the first X,Y pair, with a range of -1, is the noise record, and the second is from the calibration pulse.) TSGMEM.S10 is used for 10 microsecond sampling and TSGMEM.S20 for 20. Table 10.1 shows the TSG delays (microseconds) for the scenarios in TSGMEM.S10 as well as the ranges of the 50 echoes, assuming FEP 3,52,100 and an analog delay of 87 microseconds, that used at Boot Lake, CO.

SCENARIO	TSG DELAY	RANGE (km)
2	-10	60.45-133.95
3	490	135.45-208.95
4	990	210.45-283.95
5	1490	285.45-358.95
6	1990	360.45-433.95
7	2490	435.45-508.95

Table 10.1. TSGMEM.S10

Table 10.2, TSGMEM.S20, assumes FEP 3,52,50 and MD.ADLY of 87. Both TSGMEM.S10 and TSGMEM.S20 are to be used with FEPMEM.TWO and TSG PROM F01.

SCENARIO	TSG DELAY	RANGE (km)
2	-10	60.45-207.45
3	990	210.45-357.45
4	1990	360.45-507.45
5	2990	510.45-657.45
6	3990	660.45-807.45
7	4990	810.45-957.45

Table 10.2. TSGMEM.S20

Notice that only modes I, K, B, Z, and V invoke processing in either the FEP or the PICKER task that finds the peaks of echoes. Because they invoke PARTIAL processing, modes P, G, and W treat all X,Y pairs as if they were echoes.

For quick indentification of data sets, MD.TMDE is initialized at SEL to the standard values shown in Table 10.3.

MODE	MD.TMDE
I	Ø
K	1
V	2
B	3
P	4
G	5
Z	6
W	7

Table 1Ø.3. STANDARD MODE FLAGS.

At the beginning of each sounding, MD.TMDE is copied into SCT.MODE (Section 7.1). Also, (MD.TMDE)*2 is put into bytes 4 and 5 of SOUNDER's SØ4 queue block (Section 7.8.2). For a more complete identification of the sounding mode, MD.CTRL should be examined.

1Ø.2.2. LOCAL 2-ØØ [DCW]

Task LOCAL is used to establish values that are unique for individual HF Radar sites. These values include geographic and geomagnetic parameters and receiving antenna field geometry descriptions. In addition, 14 user-defined 16-bit values can be established. (Because the formats of these values cannot be predicted, the values are entered as 16-bit hexadecimal.)

The HF Radars are portable and are frequently moved, and local site parameters are of integral importance to the interpretation of upper atmospheric and ionospheric data. LOCAL allows the experimenter to easily change these parameters from the console, without having to modify and reassemble SOUNDER, as P1.Ø required.

The local site parameters are stored on disc file SYS1:LOCAL.TWO, which is 15 contiguous sectors long. Sector Ø contains the geographic and geomagnetic values, site code, etc., described in Section 7.6 as being from LOC.GLAT through LOC.USER. Sectors 1-1Ø contain descriptions of up to 1Ø receiving antenna fields, one description per sector. The antenna field description or sector number for a given sounding mode is set by SOUNDER command, RXANT (Chapter 8), and is kept in MD.RXAN (Section 7.1). Sectors 11-14 of LOCAL.TWO are currently unused. They were intended to hold descriptions of transmitting antennas, etc.

Currently the LOCAL site parameters become associated with the sounding data in two ways. First, when SOUNDER is first started, LOC.GLAT through LOC.USER from sector Ø of LOCAL.TWO are copied into SOUNDER's SCT at SCT.GLAT through SCT.USER. (Notice that these areas are identical in the SCT and LOCAL STRUCs in PCB.CAL. See Sections 7.1 and 7.6.) Because the SCT is the first record of every data set, SCT.GLAT-SCT.USER are written to disc and tape with each data set. Second, the entire contents of SYS1:LOCAL.TWO are written to tape each time the TAPE command is given to MANAGER. The LOCAL record on tape is flagged with X'FFØØ' in the first 1/2-word.

Given the clarity of hindsight, a better solution to the problem of associating the local site values with the sounding data might have been to have MANAGER copy to tape with each data set the record, or sector, of LOCAL.TWO specified by SOUNDER command RXANT and stored in MD.RXAN (Section 7.1). Because the pertinent data from sector Ø get to tape as part of the SCT, MANAGER would not need to copy LOCAL.TWO to tape when TAPE is given. (Unfortunately, the TXANT-MD.RXAN mechanism was not complete when

MANAGER 2.00 was written.)

CAUTION: Under the current procedure, the operator must remember to issue a CLOSE tape when a tape is fully written, then TAPE at the start of each new tape, in order to guarantee that the receiving antenna field geometry data is written to each tape.

The LOCAL task commands are:

SITE

The SITE command leads the operator by stepping through the local site parameters, displaying each current value, then waiting for a new value to be entered or for a (cr) confirmation. If (cr) only is typed, the old value is preserved; after a new value is typed, it is displayed back to the console, and the operator can either correct the new value, or confirm its correctness with (cr). The site parameters are automatically updated on disc when the SITE command runs to completion.

The site code is a 1-4 alphanumeric character description. For example, the site code for Boulder, Colorado is "BC".

The local time meridian is signed, and + = East, - = West. For example, the local time meridian for Boulder is entered as "-7.0".

The magnetic dip, and the geographic and geomagnetic latitudes and longitudes can be entered as either degrees and fractional degrees, (e.g., 67.4 N or 49 W), or degrees, minutes, and seconds (e.g., 67 49 32 S or 23 59 59 E). In the latter form, minutes and seconds are optional, and if not given, 0 is assumed. If the direction (N, S, E, or W) is not specified, the positive direction (N or E as appropriate) is assumed. The direction for geomagnetic longitude, if given, is ignored. This value is expressed in degrees, 0-359 59 59.

The gyro frequency is entered as MHz, e.g., 1.22, and the total field in Øersteds is entered in fractional form, e.g., 0.513.

10.2.2. LOCAL 2-00

The user-defined values are entered as 16-bit hexadecimal numbers, because their use, and hence their formats, cannot be predicted.

In order for the SITE values to be recorded with the sounding data, LOCAL should be run stand-alone before sounding commences.

RXANT n — Define receiving antenna array n (1-10). The argument, n, is exactly that of SOUNDER's RXANT command (Section 2.7.6)). The operator is led as in SITE, above; (cr) confirms a correct entry. The receiving antenna array description is automatically updated on disc upon completion of RXANT.

The first input is the number of antennas in the receiving array, the second is the analog delay of the recieving system and the array in microseconds. This value does not get copied to MD.ADLY at present. (See Section 7.1.)

Each receiving antenna is then defined individually, by the number of the receiver to which it is connected, the number of the receiver input port to which it is connected, and the geometry of the antenna. The X-, Y-, and the Z-coordinates are entered as signed meters, e.g., -123.4 or 234. The antenna orientation is expressed as an angle, 0-360, e.g., 270.332 or 90. Gain is expressed in dB.

END — Terminate the LOCAL task.

10.2.3. PROTF 2-00 [DCW]

PROTF constructs and displays the protected-frequencies maps for a sounding mode or for a class of sounding modes. Although the nominal range of the system is 0.10-30.0 MHz, the ultimate responsibility for frequency selection within this range must be taken at each site as local conditions dictate.

We have assumed that all sites will wish to protect the WWV channels at 2.5, 5, 10, 15, 20, and 25 MHz. In general, Table-class maps will be more restrictive than Sweep-class maps: frequencies may be protected from repetitious soundings, while not protected from a single sounding during a sweep. The map for B-mode should proabably be more restrictive than that for I- and Z-mode, but at present B-mode uses the I-mode map. (See MAKEM, Section 10.2.1.)

The PROTF task commands are:

GET a
: Select a map for editing and listing. The alphabetic character, a, is the map identifier. Two maps exist on SYS1 for standard operation. They are I, for S-class sounding modes, and K, for T- and W-class modes. Additional maps may be created with the SAVE command. To invoke a non-standard map b, a permanent mode must be made with task MAKEM (Section 10.2.1), and MD.MAP must be set to b.

  The edited map exists only in memory until explicitly written to disc (see SAVE).

PROTECT f1[-f2]
: Edit the map to protect frequency f1 or band f1-f2. The frequencies are expressed in decimal kHz. The band protected is the specified band expanded 50 kHz on each end to allow for hardware uncertainties.

UNPROTECT f1[-f2]
: Edit the map to release protection of frequency f1 or of band f1-f2. The specified band, expanded 50 kHz on each end, is unprotected. The command:

  UN 100-30000

unprotects the entire useable frequency band, except for internationally protected frequencies.

LIST — Display the frequency bands protected by the map being edited.

SAVE a — Save the map on the disc for use by the SOUNDER task. The SAVEd map may be edited further or a new map may be selected (see GET). SAVE displays the message "NEW FILE" or "OLD FILE", then waits for confirmation. A (cr) allows the file to be written, while any other character prevents it.

Note: If any file or disc errors are detected during the execution of the above PROTF commands, the MT2 system error codes are displayed (see the Programmer's Reference Guide, Chapter 8). PROTF requests another command and does not execute the command in which the error was detected. If it is necessary to get back to the MT2 command processor in order to mark a disc on, rename a file, etc., END the PROTF task and type PAUSE to COUNSEL. If corrective action cannot now be taken, type CANHF to COUNSEL or MT2 and start from square one. Any map referenced by a sounding mode must be successfully written to disc in order for SOUNDER to RUN the mode. If a map is not found on SYS1 by RUN or MGET, COUNSEL displays "ERR 23".

END — Terminate the PROTF task.

10.2.4. DIOGEN 2-00 [DCW]

The HF Radar DIO hardware offers great flexibility for the software control of standard devices (e.g., the attenuators, bandwidth selectors, and the receiver input multiplexors), as well as for the control of devices needed for special and yet to be designed experiments (e.g., outboard antenna multiplexors, T/R switches, etc.). In meeting deadlines for the P1.0 software deliverly, we were unable to fully utilize the capabilities of the DIO hardware. Software routines for controlling devices added to the DIO bus were written on an ad hoc basis, and the P1.0 software never exploited the versatility of the hardware. See Chapter 3, Volume 1 of the HF RADAR MANUAL for a description of the DIO hardware.

A major concern in rewriting the software has been to correct this deficiency. The problem, although tractable, is not straight forward, because of the great variety of devices serviced by the DIO bus. In the simplest cases, a single device is assigned a single DIO address, and is then controlled on a pulse-to-pulse basis, or perhaps set to a given state for the duration of a sounding. A more complex case, however, is presented by the synthesizer status DIO port (DIO 1E), which for some installations, is shared with the low power RF system filters control. Moreover, the 5 low power RF filters, if installed, are controlled via 3 DIO ports. The synthesizer status bits of DIO 1E cannot be set, so in a sense, are read-only; the filters bit however can be set and may or may not be read back to verify a setting. Another consideration is that DIO ports used for loading programs into the TSG and FEP microcomputers, or for controlling the transmitter filaments and HV, may be written to between soundings, but must not be disturbed while a sounding is in progress. Finally, different installations with different needs and interests should not be required to have identical hardware attached to the DIO bus in order to operate with the standard software, and the writing of special-purpose software routines to accomodate new DIO devices should be kept at a minimum. DIOGEN covers most contingencies, but does not directly support DIO devices that require frequency band dependent control.

CALL task DIOGEN is used to construct tables of information which define the DIO configuration to SOUNDER. A given DIO configuration may be shared by a number of sounding modes, or may be unique to a single mode of sounding. Thus, DIO configurations may vary from site to site and from experiment to experiment without requiring any modifications to the SOUNDER task. Each configuration is stored on disc as a file, SYS1:DIO.xxx; a DIO configuration is invoked with the SOUNDER command, BOOT DIO,xxx (Chapter 8).

Other CALL tasks (e.g., DIO, FEPEKO) that use or exercise the DIO hardware do not depend upon the DIOGEN tables in any way. Their DIO definitions are internal and/or from the SNDR block in PCB.CAL. (See Section 7.7.)

DIOGEN recognizes DIO ports as either dedicated to a particular function (such as controlling the IF attenuation on receiver channel 2) or not dedicated to any specific function (user-defined). The essential difference is that a dedicated port is supported by specific software routines in the SOUNDER task; the control of many dedicated ports is affected by commands to SOUNDER. The default list of dedicated DIO ports is assembled as a part of the DIOGEN task at DIO+DIO.DED and conforms to the DED STRUC in SYSC:PCB.CAL, invoked by the "COPY DIO." directive to the CAL assembler. Any additions to or deletions from the dedicated port list must be reflected in DIOGEN, in PCB.CAL, and in the SOUNDER task code. This list covers the original P1.0 DIO structure as well as all additions made before P2.0 and DIOGEN were written. The add-ons include the TX and TX phase control, installed in the MPI system, and the central remote DIO unit, installed in the SEL/NSF system. Although considered dedicated DIO ports, the add-ons are assembled in DIOGEN as deactivated, as are the octave filters (DIO 12 and 13) which have not as yet been installed. If a deactivated DIO port is referenced by a SOUNDER task command, ERR 18 is displayed by COUNSEL. DIOGEN allows for up to 8 outboard antenna multiplexors, referenced by number by the ANT command (Chapter 8). The DIO ports that control these devices are not considered dedicated.

The DIOGEN task commands are:

<u>GE</u>T xxx — Get the DIO control tables for editing from SYS1:DIO.xxx and display port DIO addresses, descriptions, and their priority, if any, in SOUNDER's pulse-to-pulse DIO loop. If no GET is typed before editing, the P1.0 configuration is made available by default. (See Figure 10.1.)

<u>DI</u>O [n] — Display all control parameters for DIO port n. Masks and data are displayed in hexadecimal and binary; Xs in a binary number mean that bit is not considered, or is a "don't care", because of a prior specification. (See CONNECT, below.) If n

DIO 1D: SYNTH AB	1	DIO ØA: FREE
DIO 1C: SYNTH CD	2	DIO ØB: FREE
DIO 1B: SYNTH EF	3	DIO ØC: FREE
DIO 10: RX INP 1	4	DIO ØD: FREE
DIO 11: RX INP 2	5	DIO ØE: FREE
DIO 14: RF ATN 1	6	DIO ØF: FREE
DIO 15: RF ATN 2	7	DIO 1F: FREE
DIO 16: IF ATN 1	8	DIO 2Ø: FREE
DIO 17: IF ATN 2	9	DIO 21: FREE
DIO 18: BDW 1	1Ø	DIO 22: FREE
DIO 19: BDW 2	11	DIO 23: FREE
DIO 1A: CAL ATN	12	DIO 28: FREE
DIO 1E: SYNTH ST		DIO 29: FREE
DIO 24: TSG CMND		DIO 2A: FREE
DIO 25: TSG FLGS		DIO 2B: FREE
DIO 26: TSG CLK		DIO 2C: FREE
DIO 3Ø: TX CTRL		DIO 2D: FREE
DIO 31: TX ATN		DIO 2E: FREE
DIO 32: TX ST A		DIO 2F: FREE
DIO 33: HKADC AD		DIO 38: FREE
DIO 34: HKADC RD		DIO 39: FREE
DIO 35: TX ST B		DIO 3A: FREE
DIO 36: FEP PROG		DIO 3B: FREE
DIO 37: FEP CTRL		DIO 3C: FREE
DIO Ø1: FREE		DIO 3D: FREE
DIO Ø2: FREE		DIO 3E: FREE
DIO Ø3: FREE		DIO 3F: FREE
DIO Ø4: FREE		DIO Ø5: TX PH B
DIO Ø6: FREE		DIO 12: FILTER 1
DIO Ø7: FREE		DIO 13: FILTER 2
DIO Ø8: FREE		DIO 27: CENT REM
DIO Ø9: FREE		DIO ØØ: NOT USED

Figure 1Ø.1. P1.Ø DIO CONFIGURATION.

is not specified, all ports are displayed in short form, as in GET, above. A DIO before any GET displays the P1.Ø configuration.

PRIORITY n Set the priority in SOUNDER's pulse-to-pulse DIO loop for DIO port n. If port n has previously been CONNECTed (below) as a pulse-to-pulse port (pulse-to-pulse mask > Ø) and is not read-only (data-direction mask > Ø), PRIORITY requests the new priority in SOUNDER's DIO loop. The priority can be set to 1 through 64. The highest priority is 1, the lowest is 64. SOUNDER writes to the DIO ports in ascending order of priority, for example, a port with priority 13 is set before one of priority 14. If two ports, say DIO 11 and DIO 46, have the same priority, DIO 11 is set first. Devices with long settling times or those whose settings may be prerequisite to the settings of other devices can be given highest priority. The DIO loop priority of a DIO port can also be set as part of the CONNECT procedure, described in detail below. If two or more ports are given the same priority, SOUNDER writes to them in DIO address order.

PTP — Display, in priority order, all DIO ports in the pulse palette extension to be set in SOUNDER's DIO loop. These ports are also displayed at the top of the DIO and GET command dislplays, above.

AMUX — Display all DIO ports configured as receiver antenna multiplexors with their mux numbers. The antenna mux numbers are used in the ANT command (Chapter 8) to specify that they are receiver channel inputs. The antenna mux display is in DIO loop priority order.

DISCONNECT n
Disconnect (in the software sense only) DIO port n. If the port was dedicated, it remains dedicated, but is considered deactivated. An undedicated port is simply made free for another user-defined function by DISCONNECT.

CONNECT n — Connect DIO port n. A DIO port must be free or deactivated in order to be CONNECTed. (See DISCONNECT, above.)

If the port, n, is dedicated and deactivated, CONNECT requests the new dedicated port key. The suffix, or the last four characters, of the DED STRUC entries are used as the dedicated port keys. (The DED STRUC is part of the DIO definitions invoked by COPY DIO., and is listed in Section 7.9.) If the dedicated port key specified is in use already, the port using it is displayed, and CONNECT is aborted. Otherwise, the CONNECT argument, n, becomes the DIO address associated with the specified key.

Whether port n is free or deactivated, CONNECT requests its new description. Up to 8 characters may be used to describe the port.

CONNECT next asks if the device to be connected to port n is remote. "Y" means yes; "N" or (cr) means no. (See task RDIOT, Section 10.2.13.)

CONNECT next asks if the device is a receiver antenna multiplexor. Again "Y" means yes; "N" or (cr) means no. If the response is yes, CONNECT requests the mux number (1-8). This number is used, by the ANT command (Chapter 8), to specify that an input to receiver channel 1 or 2 is a mux. If the mux number specified above is in use, a message is displayed, and the mux number is requested again. (See AMUX, above.)

CONNECT next requests the port Data-Direction mask (DD-mask). The response is Ø-FF (hex); (cr) means Ø. The DD-mask is strictly a software mechanism; bits cannot be selectively written to a DIO port. A 1-bit in any position in the DD-mask means that the device accepts data in the corresponding bit of a data byte; a Ø-bit in any position in the DD-mask means that the device cannot read that data bit. A DD-mask of Ø (zeroes in all bit positions) means that SOUNDER will not write to the device except from special, dedicated routines; the device is considered to be read-only by SOUNDER's pre-sounding, sounding, and post-sounding DIO setting routines. If the DD-mask is specified to be Ø, no more port specifications are requested.

Unless the DD-mask is Ø, CONNECT next requests the Pulse-To-Pulse mask (PTP-mask). The response is Ø-FF (hex); (cr) means Ø. The PTP-mask specified is logically ANDed with the DD-mask specified above. A 1-bit in any position of this result means that the device accepts data in that bit position on a pulse-to-pulse basis; a Ø means that it does not. A PTP-mask of Ø means that the port can have no priority in SOUNDER's pulse-to-pulse DIO loop. A PTP-mask of all ones (hex FF) means that all data bits are set pulse-to-pulse and that none of them are to affect the entire sounding.

Unless the PTP-mask is hex FF, CONNECT next requests the pre-Sounding setting (S-data) for the port device. The response is Ø-FF (hex); (cr) means Ø. The S-data byte specified is ANDed with the DD-mask and that result is ANDed with the inverted PTP-mask. S-data bits in this result are written to the device before a sounding begins, and if the PTP-mask is not Ø, the S-data byte is ORed with the pulse-to-pulse data. Thus, the S-data setting prevails for the entire sounding.

CONNECT next requests the post-sounding or Idle-state setting (I-data). The response is Ø-FF (hex); (cr) in this case means that there is no post-sounding setting. The I-data byte specified is ANDed with the DD-mask. These data are used to enable local receiver control, put in full transmitter attenuation, etc., between soundings. (Using the I-data option, the user can tune the receiver to his favorite radio station between soundings.)

CONNECT next asks if SOUNDER is to write-verify any pre-sounding, pulse-to-pulse, or post-sounding DIO device setting. "Y" means yes; "N" or (cr) means no. To verify a DIO write, SOUNDER reads the data back and compares the data, ANDed with the DD-mask, with what was written. If the verify fails, SOUNDER increments the DIO error counter. (See command SYS, Chapter 8.) No other action is taken.

Next, if the PTP-mask is not Ø, CONNECT asks for the port's priority in SOUNDER's pulse-to-pulse DIO loop. The response is 1-64 (decimal). The priority can be adjusted later with PRIORITY, above.

Finally, if the PTP-mask is not Ø, CONNECT asks if the user wishes to specify the Pulse-To-Pulse data (PTP-data). "Y" means yes; "N" or (cr) means no. If the response is no, the DIO port is fully defined. Otherwise, CONNECT prompts the user for each data byte with "Pn:", where n = 1-8, and inputs hex data bytes (Ø-FF) which become associated with the palette descriptions in SCT.PD. (See Section 7.1 and the PTP command display.) The data bytes are ANDed with the PTP-mask and are stored in the DIO.PTP section, or pulse palette extension, of the DIO control tables. These data are written to tape with each sounding data set (Sections 4.3, 4.9, and 7.9). The DIO.PTP data are flagged with a 9 in the high order 4 bits of the first 1/2-word. A (cr) in response to "Pn:" tells CONNECT to use the data byte used for Pn-1, or 0, if n = 1. The 8 bytes of PTP-data can also be temporarily set for a given DIO port with the UDIO command (Chapter 8). UDIO sets the PTP-data in SOUNDER's DIO.PTP table only, not on disc.

The PTP-data input for a dedicated port may be overridden if the data are calculated by SOUNDER in dedicated routines (e.g., as for the attenuators), or if they are set by a SOUNDER command (e.g., the antenna inputs are set by ANT).

When the port is fully defined, CONNECT displays the control parameters as does command DIO, above.

<key> n — This command is used to change the DIO address of a dedicated port to n. It should only be used if for some reason a dedicated DIO device is strapped to a non-standard address. The command, <key>, is the dedicated port key and is the suffix of a DED STRUC entry. (See CONNECT, above.) If the DIO port, n, is already in use, it is displayed, and the command is ignored. If it is not in use, the dedicated device to which <key> refers is given DIO address n, and its previous port is disconnected (made free or deactivated). Both the old and the new ports are displayed.

SAVE xxx — Save the DIO control tables just edited on SYS1:DIO.xxx. This disc file can be edited further and can be accessed by SOUNDER with the BOOT DIO,xxx command (Chapter 8). SAVE displays either "OLD FILE" or "NEW FILE" and awaits confirmation; (cr) allows the disc write, and any other character prevents it.

(SAVE writes 9 sectors to disc. Sectors 0-5 are used only by DIOGEN and contain the DCT and the DIO.PTP construction area. (See the DIO STRUC, Section 7.9.) Only Sectors 6-8 are used by SOUNDER.)

END — Terminate the DIOGEN task.

10.2.5. SCHED 2-01 [DCW]

(For details about changes made to SCHED 2-00, refer to Appendix A, dated 1983 November.)

Task SCHED constructs a schedule of sounding commands. SCHED enables the operator/experimenter to preselect a sequence of HF Radar sounding commands, to intermix sounding modes, to nest repeated sequences, and to specify optional sequences of commands in case command errors occur. Schedules of commands saved on disc by SCHED can subsequently be read and executed by an SEL-written ANALYSER task supplied with Product Two.

SEL's scheduling version of ANALYSER (referred to below as the Scheduler) assumes the role of COUNSEL in the HF Radar software to the extent of sending SOUNDER, PICKER, MANAGER, GRAPHER, and ANALYSER itself C02 advice Q-blocks. The information in bytes 4-7 of the Scheduler's C02 Q-block (Sect. 7.7.1) is obtained from the file of scheduled commands written to disc by SCHED.

SCHED builds the file as follows: when the operator/experimenter enters an HF Radar task command to be scheduled, SCHED checks its validity by scanning the tasks' command mnemonic tables. (Essentially the same scan is performed by COUNSEL when a command is typed at the console. ET.SCAN (Sect. 7.3) contains the address of the mnemonic tables address list.) If SCHED finds the command to be valid, it saves the information needed to construct a C02 Q-block: the index of the task command, the ASCII text of the command, and the index of the task to which the address of the C02 Q-block will be sent by the Scheduler. The command information is thus accumulated and is written to disc by SCHED command SSAVE (below).

SCHED 2.01 allows GRAPHER commands to be scheduled by keeping its own copy of GRAPHER's command list. This copy must, of course, be identical to that in the GRAPHER task. Some task commands may not be scheduled (e.g. TPR, SLIST, PTP, SYS, MGET) to avoid conflicts arising over console and magtape use. However, all commands essential to unattended, scheduled sounding can be used. Because SOUNDER 2.02 preserves all sounding mode dependent commands, SCHED 2.01 schedules need not be burdened with frequency selection commands, etc.

Because task commands are queued by the receiving tasks, whether typed at the console or sent under program control, any appropriate command can be given at the console even though the Scheduler is active. Thus a schedule can be adjusted, or even overridden, if desired, provided the rules concerning command usage, described in Chapter 8, are observed.

SCHED accepts the time-specific forms of the RUN command as described in Chapter 8. In addition, to provide more flexible timing, SCHED allows for the construction of a table of starting times. (See command *, below.) The string-substitution table entries are numbered from 1 to 47 and are referenced in numerical order each time an asterisk (*) is seen as an argument of the RUN command by the Scheduler. For example, if the first string-substitution table entry is "3:17" and the Scheduler encounters RUN *, "3:17" is substituted for the *, and the sounding starts 3 minutes and 17 seconds past the current hour. The second string-substitution table entry will be referenced by the next RUN *. The end of the table is established by SCHED to be the highest numbered entry, n, set by the *n command. When the Scheduler senses the end of the table, it begins again with entry 1. All entries 1 through n need not be set. If a command, RUN *, causes reference to an entry that has not been set, it is as if the command RUN(cr) had been given.

After the Scheduler has sent a C02 queue to SOUNDER initiating a sounding, that is, a RUN command, it issues no further commands until the sounding and all post-sounding activities have been completed.

Note: SCHED and ASKED, the Scheduler source, are assembled with the SKD STRUC from PCB.CAL, which parameterizes the schedule tables. SKDSR, the number of sectors in a schedule disc file, is currently 3. The storage segment sizes for the schedule tables are all derived from SKDSR, which can be increased to 4, or even 5, if required and if memory space allows. If SKDSR is changed, only SCHED and ASKED need be reassembled and reTETed.

Because SCHED must have access to the sounding tasks' command lists, it cannot run stand-alone.

The SCHED task commands are described below.

BUILD — Enter the schedule-building mode. After BUILD has been issued, SCHED accepts valid sounding task commands as well as any of the schedule control commands, which are described below. The user is advised to use the most compressed form possible of a given command to avoid overflowing SCHED's text storage area, which is 384 bytes long. (For example, KE8 is preferred over KEEP 8.)

SSAVE xxx — Write the schedule just built to disc file SYSC:SSCHED.xxx for use by the scheduling version of ANALYSER. SSAVE displays the message, "NEW FILE" or "OLD FILE". The operator confirms with (cr) only, or aborts the SSAVE command with any other character, followed by (cr). This file contains the information that the Scheduler needs for composing C02 Q-blocks, together with the text of the schedule, which can be displayed by issuing command SLIST (Chapter 8). (See also command SRUN xxx, Chapter 8.)

TSAVE name.xxx — Save the text of the schedule on disc file SYSC:name.xxx. The operator confirms or aborts the TSAVE write as described in SSAVE, above. This file can be modified with EDIT, the Interdata text editor, and can be the argument for a subsequent GET command (below).

END — Terminate the SCHED task.

After the BUILD mode has been entered, SCHED accepts the following schedule control commands:

REPEAT n — Repeat the sequence of task commands that follows the REP command n times, where n > 1. A REP loop is terminated by ENDR or ENDB (below). Loops may be nested 10 deep.

ENDR — Terminate the innermost REP loop. Commands REP and ENDR are roughly equivalent to the FORTRAN statements DO and CONTINUE.

IFERR — Begin the specification of a sequence of task commands to be executed in case the command preceding IFE resulted in an error. The IFE command is rejected if given within the body of an IFE sequence. An IFE sequence is terminated by ENDE (below) or, by implication, ENDR or ENDB (below).

ENDE — End an IFE sequence. The Scheduler skips over all commands between IFE and ENDE unless the command preceding IFE resulted in an error.

ENDB — End the BUILD mode. The schedule built can now be saved with SSAVE or TSAVE (above), or both, and a new BUILD can begin. From the Scheduler's viewpoint, single or nested REP loops are terminated by ENDB as is an IFERR sequence. ENDB is also an implied SABORT (Chapter 8), although the Scheduler will satisfy any and all REP loops terminated by ENDB before stopping.

n <text> — Specify a string to be substituted for an asterisk encountered in a task command. This command is an expansion of the SCHED 1.0 command, TIME. It is still mainly used to express a time-specific RUN, but may also be used in other ways (e.g., BOOT TSG,). The first argument, n, is the string substitution tables entry number (1-47). The <text> argument is any alphanumeric string appropriate to the task command in question. The * command can be issued at any time in the form given above. Out of the BUILD mode, command * with no arguments lists the string substitution table to the console.

LIST — List the partial or completed schedule to the console. The LIST command can be given at any time.

GET name.xxx

Get BUILD-mode commands from SYSC:name.xxx. The file can be the output from a previous TSAVE command (above), or from the system editor, EDIT. GET files may contain any BUILD-mode command except GET. Because a second BUILD command without an intervening ENDB is ignored, GET files to be concatenated can each begin with BUILD. An ENDB encountered in a GET file terminates the BUILD mode; if no ENDB is seen, SCHED remains in the BUILD mode and accepts further command input from the console. Thus, BUILD-mode commands from the console can be concatenated with those from disc files.

Examples of schedules are shown below.

For clarity, commands in the examples are not compressed, and indeed need not be for short schedules. Note that SCHED checks the validity of only the task command, not the command argument(s). A command accepted by SCHED could be rejected by the task to which it is ultimately sent if the command argument string is inappropriate. (See Chapter 8 for more complete descriptions of the task commands.)

Example 1, assuming that mode I is the current mode, initializes the high power transmitter, then runs an ionogram. Example 1 demonstrates the use of error option sequences.

COMMAND	DESCRIPTION
BUILD	Enter SCHED build mode.
STEP 1	Set frequency index increment to 1.
TXON	Begin transmitter filament warm-up.
RUN	Run ionogram to allow time for warm-up.
HVON	Turn on transmitter high voltage.
IFERR	Begin first error option sequence.
RUN	Allow more time for filament warm-up.
HVON	Try HVON again.
ENDE	End first error option sequence.
IFERR	Begin error sequence for second HVON.
SABORT	Abort schedule; second HVON failed.
ENDE	End second error sequence.
DLOF	Select antenna TX load.
STEP 4	Set frequency index step to 4.
RUN	Run final sounding, HV on.
ENDB	End build mode and schedule (implied SABORT).

The first ionogram should allow enough time for the 90-second transmitter filament warm-up between the TXON and the HVON commands. If not, the HVON command will produce an error; the first error option sequence runs another ionogram and tries again to turn on the high voltage. If the second HVON command fails, the schedule is aborted. If either HVON command is executed without error, the Scheduler skips the subsequent error option commands and runs a final I-mode sounding with the high voltage on and the antenna load selected for the transmitter.

Example 2 assumes that all initialization has been done for I- and K-mode sounding in order to demonstrate simply the use of REPEAT and ENDR. Example 2 produces an ionogram followed by five consecutive K-mode soundings; this sequence is repeated three times.

COMMAND	DESCRIPTION
BUILD	Enter build mode.
REPEAT 3	Begin outer loop.
RUN I	Run I-mode sounding.
REPEAT 5	Begin inner loop.
RUN K	Run K-mode sounding.
ENDR	End inner loop.
ENDR	End outer loop.
ENDB	End build mode and schedule (implied SABORT).

In the above case, one or both of the ENDRs may be omitted, since ENDB implies ENDR, if ENDR is missing. The following sequence is equivalent to that above; the commands are compressed.

```
BU
REP3
R I
REP5
R K
ENDB
```

Example 3 demonstrates the use of time-specific sounding starts. It is designed to run three "standard" I-mode soundings (mode I) on each quarter hour, to run a K-mode sounding (mode K) five minutes past each quarter hour, and to run three user designed I-mode soundings (mode IU) ten minutes past each quarter hour. This quarter-hourly sequence is repeated for eight hours. The following explanation assumes that command SRUN xxx,7 is given to the Scheduler (ANALYSER), so that the first time-specific I-mode sounding starts at 30 minutes past the hour. (See command SRUN, Chapter 8.) Note the exact form of the RUN * command; the space is required.

COMMAND	DESCRIPTION
BUILD	Enter build mode.
REPEAT 8	Begin hourly sequence loop.
REPEAT 4	Begin quarter-hourly sequence loop.
RUN I *	Run "standard" I-mode sounding at 30, 45, 60, 15, 30,... minutes past start of current hour.
REPEAT 2	Begin loop to run 2 more I-mode soundings.
RUN	Run next I-mode sounding as soon as possible.
ENDR	End "standard" I-mode sounding loop.
RUN K *	Run K-mode sounding at 35, 50, 5, 20, 35,.. minutes past start of current hour.
RUN IU *	Run IU-mode sounding at 40, 55, 10, 25, 40,.. minutes past start of current hour.
REPEAT 2	Begin loop to run 2 more IU-mode soundings.
RUN	Run next IU-mode sounding as soon as possible.
ENDR	End IU-mode sounding loop.
ENDR	End quarter-hourly sequence loop.
ENDR	End hourly sequence loop.
*1 60	*1 through *12 set up the string substitution table for the time-specific starts.
*2 5	
*3 10	
*4 15	
*5 20	
*6 25	
*7 30	
*8 35	
*9 40	
*10 45	
*11 50	
*12 55	
ENDB	End build mode and schedule (implied SABORT).

1Ø.2.5. SCHED 2-Ø1

Simple schedules that do not mix sounding modes are easily set up and do not require the use of the Scheduler. For example, to produce an ionogram on each quarter hour for an 8-hour period, type the following commands at the console to COUNSEL:

COMMAND	DESCRIPTION
SPH 4	Run 4 soundings per hour.
SN 32	Stop after 32 soundings or 8 hours.
RUN I 6Ø	Start at the beginning of the next hour.

A third mechanism exists for building non-repetitive schedules. See COUNSEL's command, COMMAND (Chapter 8).

10.2.6. EXOUT 2-01 [DCW]

(For details about changes made to EXOUT 2-00, refer to Appendix A, dated 1983 November.)

EXOUT examines HF Radar sounding output data stored on disc. These data are the direct output of the FEP, if PARTIAL (partial reflection) processing was used. Otherwise, they are output from the PICKER task. In either case, they are written to the disc by MANAGER.

The data from a given pulse set are displayed on the console as described in the following paragraphs.

The four PCTs for the pulse set are listed in order in their compacted, hexadecimal format as described in Section 7.2.

If PICKER, given its range gating and KEEP criteria, has found time-coincident echoes in the EKO data from the FEP for the pulse set, EXOUT displays the average range in microseconds of the echoes, together with the corresponding virtual height in kilometers. The echoes are numbered from 0. (See TPLOT command, DEFINE, Section 10.2.8.) The calibration pulse "echo" will always be numbered 0, unless the noise record has been selected. (See command, MEAN, Section 3.6.) If selected, the noise record, flagged with a range of -1 (65535), will be numbered 0; the calibration pulse "echo" will be echo 1. The noise record does not contain quadrature components, so the phase and amplitude computations for this record should be ignored.

Next, for modes I, K, B, and Z (and their variants), EXOUT displays the quadrature components of the peaks of the time-coincident echoes, as derived by PICKER, together with true amplitude and phase (degrees), as computed by subroutine PHAMP in the SOUNDER task. For modes P, G, and W (and their variants), the display is the same; every X,Y pair is considered to be an echo.

These data are listed in 8 columns that alternate channels 1 and 2 for the typical four pulses of the pulse set. (See Figure 10.2.) The default order for the peak-finding modes is shown below. (See the default sequence in Table 2.1.) The columns are numbered from left to right. F is the base frequency of the pulse set; delta F is set by command DELF and its default value is 8 kHz.

1Ø.2.6. EXOUT 2-Ø1

Although EXOUT can handle data sets with up to 6 pulses/set, only the data from the first 4 pulses of a set are displayed because of the line length of the standard TEK 4Ø12 console screen. For debugging purposes, however, a pulse set defined by PSEQ 1,2,3,4,5,6 can be temporarily redefined, e.g., as PSEQ 5,6,1,2,3,4, in order to examine data from the last two pulses of a 6-pulse set.

```
                    FREQUENCY (1ST): 2660.0 kHz

                     PCTs:
                     81AE 0875 67E8 FFCC 0A00
                     01B8 0875 67E8 FFCC 0A01
                     01C2 0875 67E8 FFCC 0A02
                     01CC 0875 67E8 FFCC 0A03

RANGE: 65535 us, 9830.25 km (ECHO: 0)

    X:      4        4        6        5        4        2        3        2
    Y:    733      749      479      808      647      655      398      774
 AMPL:    733      749      479      808      647      655      398      774
PHASE:     89       89       89       89       89       89       89       89

RANGE: 683 us, 102.45 km (ECHO: 1)

    X: -  661  -  528  -  131  -  505  -  597  -  413  -  118  -  542
    Y:    145     443  -  414  -  540     101     449  -  339  -  464
 AMPL:    676     689     434     739     605     610     359     714
PHASE:    167     139     252     226     170     132     250     220

RANGE: 753 us, 112.95 km (ECHO: 2)

    X:    346     291      82     332     422     272     101     327
    Y: -   98  -  268     297     134  -  113  -  264     318     118
 AMPL:    359     395     308     358     436     378     333     347
PHASE:    344     317      74      22     344     315      72      19

RANGE: 1383 us, 207.45 km (ECHO: 3)

    X:     68      40      29       4      61      40      25  -    4
    Y: -    3  -   34      14      55  -    5  -   31      14      51
 AMPL:     68      52      32      55      61      50      28      51
PHASE:    357     319      25      85     355     321      29      94
```

Figure 1Ø.2. EXOUT DISPLAY.

COL	PULSE	CHAN	FREQUENCY	ANTENNA
1	1	1	F	N
2	1	2	F	S
3	2	1	F + delta F	E
4	2	2	F + delta F	W
5	3	1	F + delta F	N
6	3	2	F + delta F	S
7	4	1	F	E
8	4	2	F	W

Because EXOUT gets data from MANAGER, it cannot run stand alone.

The EXOUT task commands are described below.

DISC n — Get data set for previous sounding n. The data set number, n, is the same as for the GRAPHER command GPION in Chapter 8. 0 means the data set just recorded, 1 means the data set before that, etc.

RANGE [n] — Specify the number of a pulse of a W-class sounding pulse set for which the ranges are to be computed for the EXOUT display. (See W-mode under MAKEM, Section 10.2.1.) The pulse number, n, must be in the range 1-MD.PPS, and a W-class data set must be selected before the RANGE command is given. The ranges for the pulse n data are computed as described under W-mode, and the display of the nth PCT is flagged with "**". RANGE(cr) selects the range calculations as provided by PICKER. This is the default condition when EXOUT 2-01 first starts up.

LIST — Display the first/next pulse set data or echo records of attached data set n.

LIST f — Display the echo records for frequency f (kHz) or the next higher frequency, if frequency f was not used in the sounding. The data set is scanned forward to find the specified data.

LIST ALL — Display all echo records from the current position in the data set to the end.

(cr) If a data set has been selected with the DISC command above, a carriage return is equivalent to the LIST command; otherwise, it is ignored.

END Terminate the EXOUT task.

10.2.7. PLOTEKO 2-01 [DCW]

(For details about changes made to PLOTEKO 2-00, refer to Appendix A, dated 1983 November.)

PLOTEKO plots and lists EKO data as output directly from the FEP. After a sounding, the last four SOUNDER EKO buffers are available for plotting and listing, although the first data group, or echo (usually the calibration pulse), will be corrupted because PICKER (SNDR) uses the top of SOUNDER's EKO buffers for scratch storage. Tapes written by CALL task TENUS (Section 10.3.1) may also be used as input to PLOTEKO. If the TAPE command is given to PLOTEKO (see below), the four EKO buffers will always represent one pulse set.

Currently, PLOTEKO only plots data from the P1.0 original (SNDR) FEP program. (FEP SNDR proccessing does not include the mean-max, or noise record.) This data format is selected by scenario 4, if TSGMEM.TWO and FEPMEM.TWO are used together. PLOTEKO will however list data from any FEP program.

Because PLOTEKO uses SOUNDER's EKO buffers, it cannot run stand-alone.

PLOTEKO accepts the following commands:

PLOT n — Plot EKO buffer n, n = 1 through 4. Receiver channel 1 amplitude is plotted above the horizontal center line, channel 2 below. (See Figure 10.3.) Range (virtual height or time) increases from left to right. Range 0 is the start of the FEP WRite ENaBLe window (see Figure 10.8), the time at which the FEP, under command of the TSG, begins listening for returns. Full scale range (at extreme right) represents 5.12 ms after the beginning of the FEP WRite ENaBle. Thus, about 768 km of range are represented on the X-axis. The computed amplitude (see AMP, BARXY) is plotted vertically and is scaled against the largest amplitude found in all groups of both channels. Data groups are delineated in range by tick marks extending up or down from the X-axis for receiver channels 1 or 2, respectively, and are numbered left to right from 0. Data groups (the "spikes" plotted) may or may not be true echoes; PICKER's job is to separate the echoes from the noise. PLOTEKO plots all data that passes the FEP thresholding criteria.

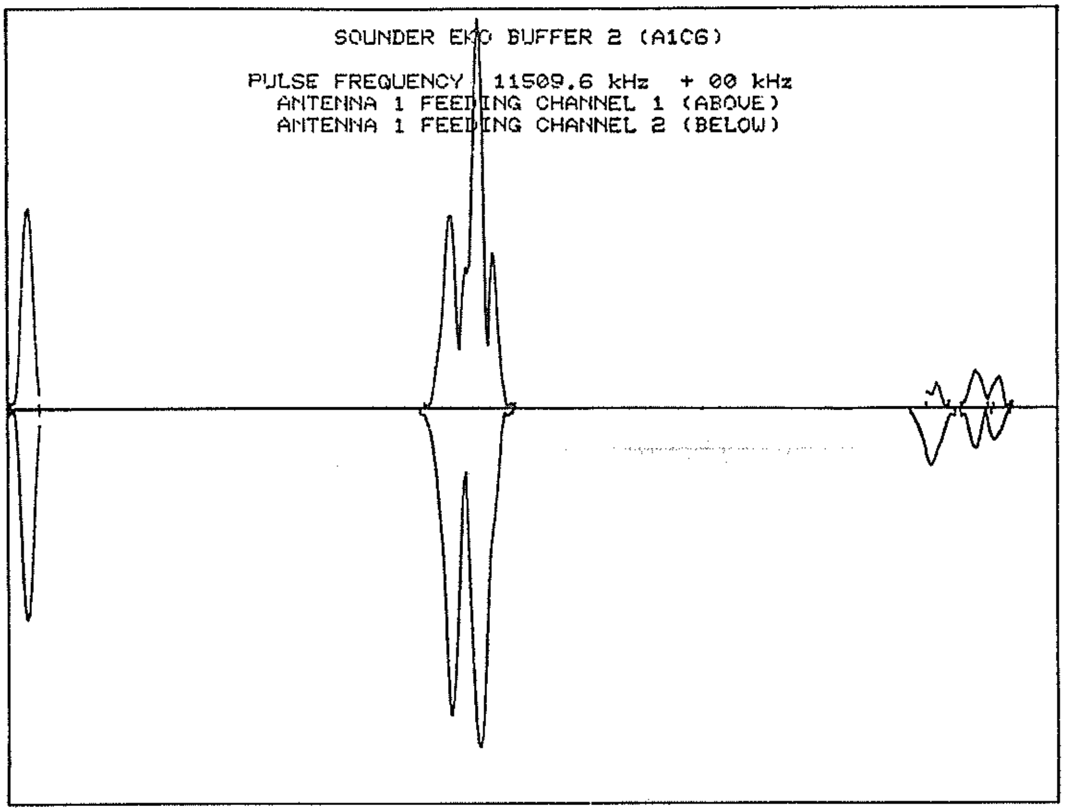


Figure 10.3. PLOTEKO PLOT DISPLAY.

DETAIL n Plot individual groups 0 for both receiver channels of EKO buffer n (1-4) if within 40 FEP range units of each other. (See Figure 10.4.) One FEP range unit equals 5 microseconds, or about 750 meters. The channel 1 group is plotted to the left of the vertical center line, the channel 2 group to the right. Straight lines connect the amplitudes of the individual X,Y pairs taken at FEP ADC sampling intervals (usually of 10, but may be 20 microseconds), and the phi symbol is used to plot phase. Phase ranges between -pi and +pi radians (as labeled for each channel) with a center line at 0 phase for each channel. Range increases from bottom to top. Group 0, the calibration pulse, always occurs at range Ø, and is plotted first by DETAIL. If the starting ranges for the groups plotted side-by-side are not identical, the plots will be offset so that range is mirrored left to right.

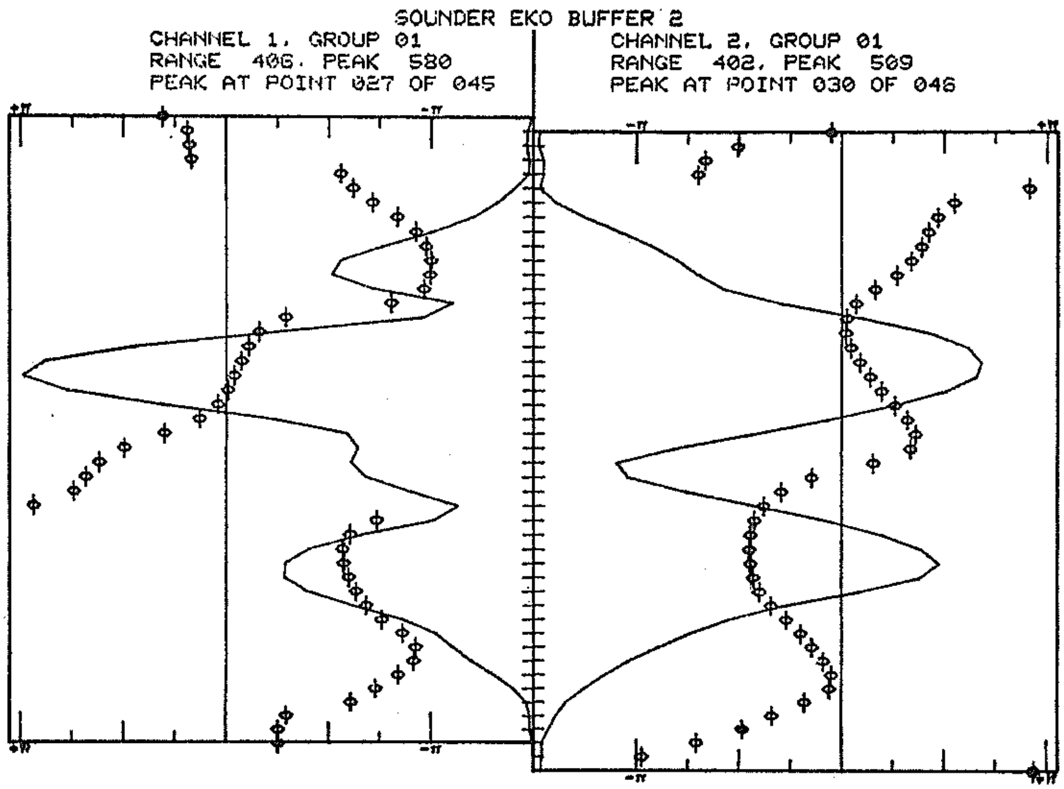


Figure 1Ø.4. PLOTEKO DETAIL DISPLAY.

DETAIL n,G1

Plot individual data groups G1 of EKO buffer n (1-4) for both receiver channels. (The data groups, as displayed by the PLOT command, are numbered from left to right, as range increases, counting from Ø, the calibration pulse.) If a group number G1 exists for either or both channels, it is plotted.

DETAIL n,G1,G2
Plot group G1 of channel 1 with a group G2 of channel 2, both from EKO buffer n. Data group numbering is described above. DETAIL plots are scaled vertically (range) to fit the screen; if the point density is high, a horizontal bar is plotted for phase, rather than the phi symbol. The ordinates are scaled against the largest amplitude found in all groups of both channels.

CONT — Continue with DETAIL plots of single groups from EKO buffer selected by the DETAIL command. CONT is ignored if all groups have been plotted.

POLAR — Plot DETAIL plots in polar coordinates. (Provided by R. I. Kressman, BAS.)

PHAMP — Plot DETAIL plots as phase, amplitude. PHAMP is the default DETAIL mode.

SCALE n — Set scale for polar DETAIL plot. (Provided by R. I. Kressman, BAS.)

LIST n — List EKO buffer n (1-4) contents at the console in signed decimal.

AMP — Compute amplitude as SQRT(X**2 + Y**2) (FORTRAN notation). This is the default computation and remains in effect until BARXY is typed.

BARXY — Compute amplitude as ABS(X) + ABS(Y). This is similar to the amplitude approximation used by PICKER for FEP SNDR processing. BARXY remains in effect until AMP is typed.

The present FEP amplitude calculation uses A + B/2, where,

A = max[ABS(X),ABS(Y)]; B = min[ABS(X),ABS(Y)]

TAPE — Read the next set (4) of PCT/EKO buffer records from magnetic tape written by CALL task TENUS. (See Section 10.3.1.) The sounding data are read into SOUNDER's EKO buffers and are available for plotting or listing.

<u>TAPE</u> f — Scan forward through the tape until the pulse set data for frequency f or the next higher frequency are found, and read the set into SOUNDER's EKO buffers.

<u>RW</u> — Rewind the magnetic tape.

<u>FF</u> — Forward space one tape file to the next sounding.

<u>END</u> — Terminate the PLOTEKO task.

1Ø.2.8. TPLOT 2-Ø1 [DCW]

(For details about changes made to TPLOT 2-ØØ, refer to Appendix A, dated 1983 November.)

Task TPLOT plots time series data in several different formats, from data recorded as T-, or W-class, that is, K-, P-, G-, or W-modes, or their variants. (See Figure 1Ø.5.) The high order bit of the MD.CTRL byte is checked to verify the sounding class. These data must be the standard output from the HF Radar. Data sets may be selected for plotting either from disc (for near real time review) or from magnetic tape. TPLOT 2-Ø1 can handle data sets of up to 6 pulses per set and 52 echoes per frequency. Notice that, in most circumstances, the disc will contain a collection of the most recently recorded data sets, written and referenced by MANAGER. The span of time covered by this collection depends upon the sizes of the data sets (or the lengths of the soundings that produced them) and the amount of disc storage available to MANAGER.

Disc recorded data sets are not available to TPLOT if the task is running stand-alone.

TPLOT maintains definitions of six focus, or plot definition levels. The levels are numbered Ø-5. Each focus level may be defined by the operator to call for a single plot, or for two plots superimposed on a single frame. Focus level Ø is predefined to plot phase and amplitude vs. time for frequency FØ (Chapter 8, command F), using the first pulse of echo number 2 from receiver 1. (Echo Ø is normally the noise record, flagged with a range of -1 (65535); echo 1 is the calibration pulse. Echo 2 is the first true echo.) By default, a newly selected data set (see TAPE and DISC below) is plotted at focus level Ø. However, the sequence

```
PLOT n
LOCK
```

locks the focus at level n for subsequent TAPE or DISC data set selection commands.

TPLOT commands are:

DEFINE n   Define focus level n (Ø-5). TPLOT leads the operator by displaying currently selected or default values, with the accepted range of values enclosed within parentheses. New values may be entered, or the old values will be retained if (cr) only is typed.

TPLOT requests the X-axis, or time, range in sets. (See command NSET, Chapter 8.) The minimum set number is 1, the maximum is 32767. The minimum range of sets is 10. (Because the X-axis is calibrated in pulse sets, the time scale may be compressed and does not account for any NOP intervals between pulse sets.)

TPLOT next requests the specification of the Y-axis for plot 1. Valid plot types are:

AMPLITUDE	Plot amplitude vs. time.
DAMPLITUDE	Pulse set to pulse set change in amplitude vs. time.
LOGAMPL	Log amplitude vs. time.
POWER	Log of amplitude squared vs. time.
PHASE	Phase vs. time.
DPHASE	Pulse set to pulse set change in phase vs. time.
HEIGHT	Virtual height vs. time.
DHEIGHT	Pulse set to pulse set change in height vs. time.

TPLOT next requests the range of the units for the Y-axis. The maximim range of Y-axis units is displayed enclosed within parentheses.

The frequency number (0-9) is next requested. The frequency number corresponds exactly to that specified by the F command. (See Chapter 8.)

The echo number corresponds to that displayed by EXOUT (Section 10.2.6). (Echo number 0 may be the noise record (range = -1), and if so, it will be incorrectly displayed, because phase and amplitude calculations are made on the mean and max values as if they were quadrature components.)

The pulse number is 1-6, and the data selected for plotting depends upon the delta F and antenna sequences specified by commands DFAC and ANT for the sounding being plotted.

Finally, TPLOT requests the receiver number (1-2).

Thus, plot 1 is fully specified. TPLOT begins the Y-axis specification for the 2nd, or superimposed, plot at this focus level. If no plot 2 is desired, type (cr) in response to

PLOT 2
Y-AXIS:

Otherwise, the sequence for defining plot 2 is exactly that for plot 1. (The plot 1 symbol is a vertical bar; the plot 2 symbol is a horizontal bar. See Figure 1Ø.5.)

TAPE [n] Select a new data set from magnetic tape and plot at focus level Ø, unless the focus is locked at another level. If given, n specifies the nth data set of the current tape file. (See command FORMAT, Chapter 8.) TPLOT scans forward for the nth data set. If n is not specified, the next data set on tape is selected. In either case, if the data set selected is not T- or W-class, the tape is scanned forward until a T- or W-class data set is found, or until a double EOF is sensed.

Note: When first started, TPLOT assumes the tape to be at the append position, that is, between the two EOFs at the current end-of-information, where MANAGER left it. To prevent the tape from running away, the tape cannot be read forward from this position. Therefore, after TPLOT start up the operator must either rewind (RW) or backfile (BF) the tape before selecting a data set from the tape.

DISC n Select a new data set from disc and plot at focus level Ø, unless otherwise locked. This command specifies a disc data set like GRAPHER's GPION n command: if n = Ø, the most recent data set is selected, etc. If the data set is not T- or W-class, an error message is displayed. (Another data set must be selected before the PLOT n command can be given.)

PLOT n — Plot currently selected data set at focus level n. If the currently selected data set is from tape, TPLOT assumes that the tape is positioned at the end of the data set and that the tape must be repositioned to the start of the data set before plotting begins. For this reason any tape positioning commands (see below) that intervene between the TAPE and PLOT n commands cause TPLOT to discard the data set currently selected from tape. TPLOT reads through the data for each TAPE, DISC, or PLOT n command. Unless the tape position is explicitly changed, TPLOT automatically positions the tape for each PLOT.

The PLOT n command ignores the state of the focus level lock, but leaves it unaltered for subsequent TAPE and DISC commands.

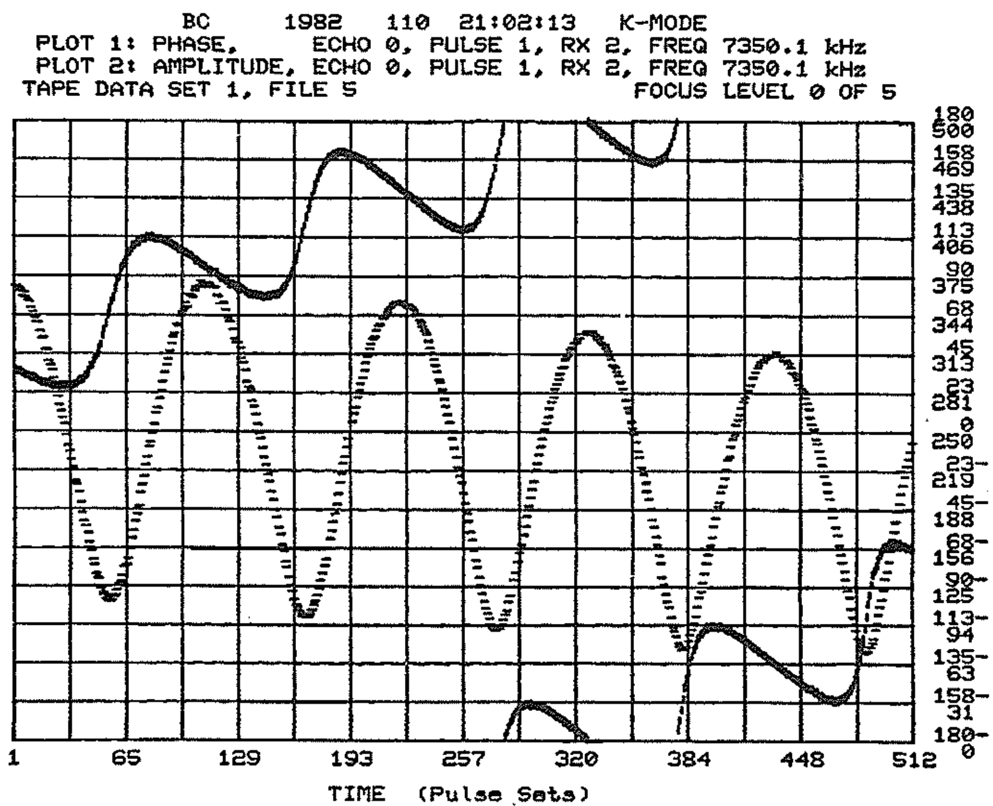


Figure 10.5. TPLOT DISPLAY.

10.2.8. TPLOT 2-01

LOCK — Lock the focus level at the level just PLOTted. Issued when the cursor reappears after a PLOT finishes, this command locks the focus level for subsequent TAPE and DISC commands until the focus is unlocked. Note that a focus level may be DEFINEd at any time; LOCK affects plotting with TAPE or DISC only.

UNLOCK — Unlock the focus, making focus level 0 the default. Until the LOCK command is given for the first time, newly selected data sets will also be plotted at focus level 0. (See TAPE and DISC, above.)

CT n,m — Get 32-bit hexadecimal time correction from console to correct the starting time of the sounding. The time correction is added to the value taken from SCT.STS; the corrected time is used for the plot only.

UT — Clear time correction. UT assumes the times recorded with the data sets are correct. UT is the default condition.

END — Terminate the TPLOT task.

In addition to the above commands, TPLOT recognizes a number of commands to position the magnetic tape. Remember that a physical tape file may contain one or many data sets, depending upon the form of the FORMAT command issued to MANAGER. The tape positioning commands follow.

RW — Rewind the tape to its load point. RW makes the magtape available for data set selection.

FF [n] — Skip forward n file(s). If n is not given, n = 1.

BF — Position the tape at the beginning of the previous or current tape file. If an EOF was the last tape record read, or if TPLOT has just been started, BF backs over 2 EOFs, then does a forward file; if the tape is not positioned at the beginning of a file, BF backs over a single EOF, then does a FF. Like RW, BF enables the TAPE command. Notice that BF may, or may not, be equivalent to backing up one data set.

FILE n — Scan forward to tape file n. The file number n must be greater than the current tape position.

APPEND — Position the tape at the current end-of-information, that is, between the double EOFs. APPEND leaves the tape positioned for MANAGER to append more data sets.

CAUTION: APPEND assumes that if FORMAT > 1 was given to MANAGER, a CLOSE tape was issued before TPLOT was RUN, i.e., that two EOFs signal the end-of-information. APPEND disables the TAPE command until RW or BF is again given.

10.2.9. SPLOT 2-01 [DCW]

(For details about changes made to SPLOT 2-01, refer to Appendix A, dated 1983 November.)

Task SPLOT is the Sweep-class counterpart of TPLOT. SPLOT plots all data from I-, B-, and Z-modes and their variants, provided the data are standard output from the HF Radar. (See Figure 10.6.) The high order bit of the MD.CTRL byte is checked to verify the sounding class. Currently, SPLOT ignores the log/linear bits in the MD.CTRL byte in the SCT (Section 7.1), and plots only by frequency index, not by linear frequency. The noise (mean, max) record is ignored, because its range (-1) causes it to fall off-scale.

SPLOT 2-01 can handle data sets of up to 4 pulses per set and 16 echoes per frequency. (These limits are parameterized in lines 31-35 of SPLOT.CAL. If enlarged, SPLOT cannot exceed GRAPHER in size if SPLOT is to run under SOUNDER.)

The command structure of SPLOT is almost identical to that of TPLOT. Only commands that are unique to SPLOT will be detailed in this section. These are:

DEFINE n — Define focus level n (0-5). SPLOT first requests the range of frequencies to be plotted. Enter the frequencies in kHz using any separator. If a single frequency is entered, SPLOT will assume a minimum range of frequencies. Normally the X-axis scaling is logarithmic in frequency. If the frequency range is less than 75 frequency index steps, however, the X-axis scaling will be linear in frequency index.

SPLOT next requests the virtual height range to be plotted. Enter the range in km. Again, if a single value is entered, SPLOT will assume a mimimum range. The maximum virtual height allowed is 3276 km; the greatest range plotted is 1047 km. Otherwise, ARITH ERRORS occur. (See RECOV, Section 10.2.22.)

Having input the plot limits, SPLOT plots a box around the area specified with lines radiating to the corners of the original plot for clarity, and asks for confirmation (Y or N). A response of N causes SPLOT to repeat the above sequence; a response of Y causes the cursor to be displayed in anticipation of another command. Another focus level may be defined or level n may be PLOTted.

NOCAL — Suppress plotting of the calibration pulse. NOCAL is the default condition.

CAL — Plot the calibration pulse as if it were a true echo.

n — The decimal number, n, is taken to be a frequency index. The corresponding frequency is displayed in kHz.

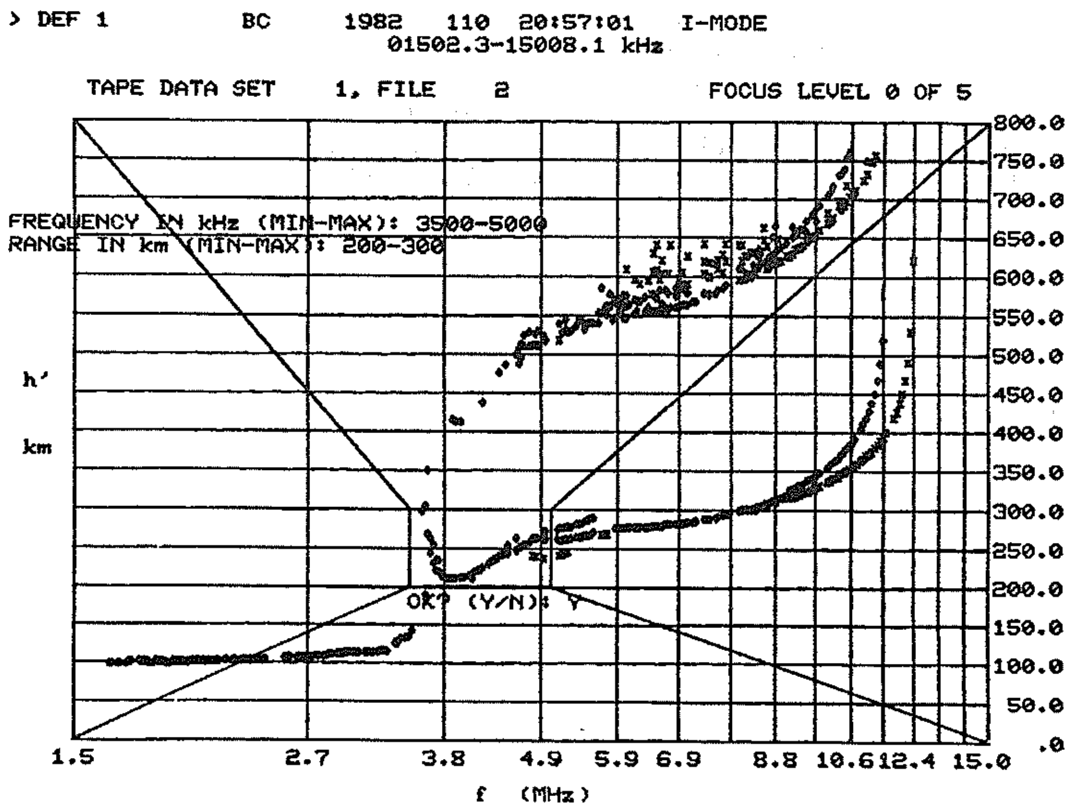


Figure 10.6. SPLOT DISPLAY.

1Ø.2.9. SPLOT 2-Ø1

SPLOT assumes the standard ANT and DELF sequences, and plots an "x" for extraordinary echoes and an "o" for ordinary echoes. SPLOT, like GRAPHER, uses SOUNDER subroutine XORO in SNDREENT for its X-O discrimination. (See XORO, Section 2.6.3.)

All SPLOT commands, except those listed above, are identical to those of TPLOT. Refer to Section 1Ø.2.8 for details. For quick reference, these commands are:

TAPE [n]	Select and plot data set from tape.
DISC n	Select and plot data set from disc.
PLOT n	Plot selected data set at focus level n.
LOCK	Lock the focus.
UNLOCK	Unlock the focus.
CT	Correct times from SCT.STS.
UT	Use times as recorded in SCT.
RW	Rewind magtape.
FF [n]	Skip [n] EOFs.
BF	Backfile magtape (cleverly).
FILE n	Forward file magtape to file n.
APPEND	Leave magtape in appending position for MANAGER.
END	Terminate the SPLOT task.

1Ø.2.1Ø. SNRIO 2-Ø1 [DCW]

(For details about changes made to SNRIO 2-ØØ, refer to Appendix A, dated 1983 November.)

Task SNRIO is used to display TSG timing scenarios as stored in zero-page images on TSGMEM disc files. The default zero-page is that of TSGMEM.TWO (Figure 1Ø.7).

M LSD D	0	2	4	6	8	A	C	E
0	0020	520B	4252	0F20	520F	4252	2320	5223
1	4252	1320	5213	0104	0810	0000	0000	0020
2	0303	83FF	83FE	9300	0103	9300	0A03	1311
3	1203	100A	1205	930F	9103	8107	83F7	83FF
4	9300	0A03	1311	1203	1006	1205	930F	9103
5	8103	83FB	83FF	9300	0000	0000	0000	0000
6	0B03	1311	1203	100A	1205	930F	9103	8107
7	83F0	810A	83FC	9300	0B03	1311	1203	1006
8	1205	930F	9103	8103	83F2	8106	83FE	9300
9	0B03	1303	1103	0103	030A	1203	1006	1205
A	930A	83FF	83F8	9300	0000	0000	0000	0000
B	0000	0000	0000	0000	0000	0000	0000	0000
C	0000	0000	0000	0000	0000	0000	0000	0000
D	0000	0000	0000	0000	0000	0000	0000	0000
E	0000	0000	00A3	F8EC	F81B	F960	F99F	F9BB
F	F9DD	F900	0000	0000	0000	0000	0000	0000

Figure 1Ø.7. TSGMEM.TWO ZERO-PAGE.

SNRIO commands are:

GET xxx — Get zero page from TSGMEM.xxx for plotting. If no GET command is issued, SNRIO uses the TSGMEM.TWO zero-page.

PLOT n — Plot scenario n (0-7) by vector state change (Figure 10.8). There will be no plot if the last GET command resulted in an error.

PLOT nT — Plot scenario n (0-7) in time scale (Figure 10.9). Major tick marks are 100 milliseconds apart.

END — Terminate the SNRIO task.

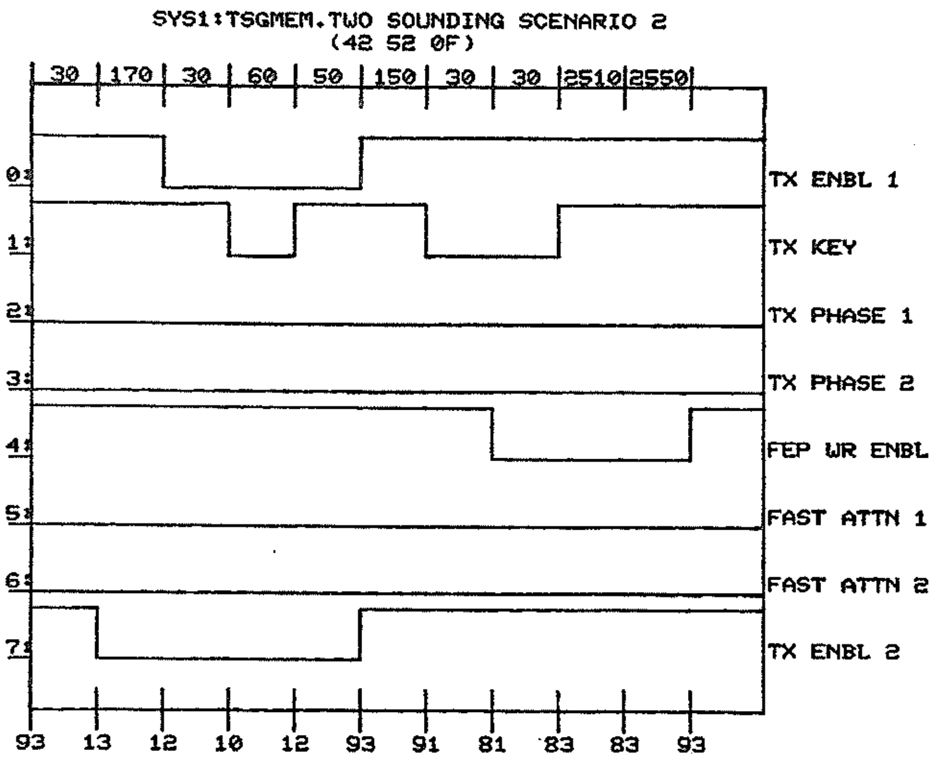


Figure 10.8. SNRIO PLOT 2.

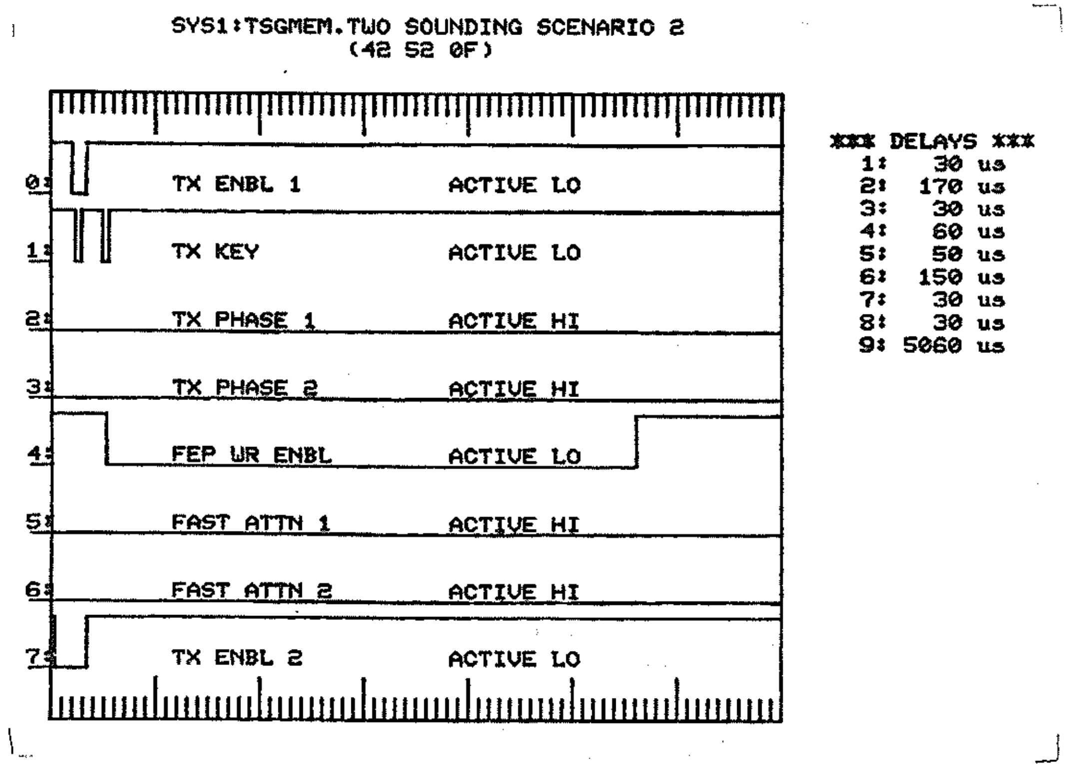


Figure 10.9. SNRIO PLOT 2T.

Currently, TSG scenarios are created or modified by DUTIL (Sections 10.4.1 and 12.4). An editor for TSG scenarios is designed, but not yet implemented.

10.2.11. SCLOOP 2-00 [DCW]

Task SCLOOP sets up read/write loops for checking the DIO with logic analyzer or oscilloscope. Loops can mix reads and writes or can be declared to be either exclusively read or write from any point of the loop to the end.

SCLOOP commands are:

LOOP — Set up a loop. The program displays the command number (1-100) and requests the command type, unless the loop has been previously declared read or write (RL or WL, below). Valid responses to the "READ/WRITE:"

- R — Make this command a read.
- W — Make this command a write.
- RL — Make this and all following commands reads.
- WL — Make this and all following commands writes.
- (cr) All commands for the loop have been entered. All that remains to be specified is the loop count.

After the operator has responded to the "READ/WRITE:" request with other than (cr), or if RL or WL has previously been typed, SCLOOP requests the DIO address that is to be read from or written to. The response is any hex number from 0 to 3F. If a number greater than 3F is entered, only the low order 6 bits will be used. (Not all addresses from 0 to 3F are used by the HF Radar hardware. See the DED STRUC, Section 7.9 and DIOGEN, Section 10.2.4.) If RL or WL has previously been typed, a (cr) typed in response to "DIO ADDRESS:" terminates the read/write command input, and the loop count is requested.

If the current command is a write, SCLOOP next requests "DATA:". The response is any hex number from 0 through FF. If a number greater than FF is typed, only the low order 8 bits will be written to the DIO address specified.

After the write data or the read DIO address has been entered, SCLOOP requests "DELAY:". The response is a decimal number 0-32767 and is roughly the number of milliseconds to delay after the current command has been executed in the loop. The delay timing loop is not accurate. A delay request of 0 results in a delay of about 20 microseconds.

When the operator terminates command input with (cr) in response to "READ/WRITE:" or to "DIO ADDRESS:", or if 100 commands have been entered, SCLOOP requests "LOOP COUNT:", the number of times each command in the loop is to be executed. The response is 0-32767 (decimal) or (cr) only, if looping is to continue until interrupted (below).

After the loop count is speicifed, the DIO read/write loop is executed. A loop may run to completion, given a finite loop count, or it may be interrupted at any time with a single (cr) typed at the console. An interrupted loop may be restarted with another (cr). All specified loop parameters remain intact until a new LOOP command is typed. Thus, a loop, once set up, may be started and stopped at will by typing a series of (cr)s.

DATA — Display the data read, if any, the last time through the loop. This command cannot be given unless the loop has been interrupted. After the read data has been displayed, the interrupted loop may be restarted with (cr).

END — Terminate the SCLOOP task.

1Ø.2.12. DIO 2-ØØ [DCW]

Task DIO sets and checks the radar receiver instruments and other devices on the DIO bus. A...A is the argument of a command. See the DED STRUC (Section 7.9) for a list of DIO devices by DIO address.

The DIO task commands are:

1ANT AA	Select channel 1 antenna input, 1-4.
2ANT AA	Select channel 2 antenna input, 1-4.
1RF AA	Select channel 1 RF attenuation, Ø-6Ø dB.
2RF AA	Select channel 2 RF attenuation, Ø-6Ø dB.
1IF AA	Select channel 1 IF attenuation, Ø-6Ø dB.
2IF AA	Select channel 2 IF attenuation, Ø-6Ø dB.
1FIL AA	Select channel 1 bandwidth filter, 1, 3, 1Ø, or 3Ø kHz.
2FIL AA	Select channel 2 bandwidth filter, 1, 3, 1Ø, or 3Ø kHz.
CAL AA	Select calibration attenuation, Ø-12Ø dB.
FREQ AAAAA.A	Select synthesizer frequency in kHz.
DIO DD,AA	Command DIO device DD (hex) with hex value AA.
	If any of the above commands are given without an argument AA, the current setting of the device is displayed.
HSK CC	Read Housekeeping ADC channel CC and display the reading in volts and in ADC counts (hex).

LOCAL — During a sounding, SOUNDER puts the receiver instruments under remote, or program, control, disabling the front panel manual controls. LOCAL removes the remote lock and enables the LOCAL CONTROL push switches, which, when depressed, enable the front panel controls.

RWCK — Check the DIO interface read-write/read-only state command. Both states are tested 10000 times and any failures are reported. Run this test if SOUNDER command SYS shows any DIO errors.

STATUS — Display the current receiver instrument settings.

END — Terminate the DIO task.

10.2.13. RDIOT 2-00 [DCW]

The RDIOT task is used to test the remote DIO hardware. The remote DIO allows devices on the DIO bus located up to 1 km from the HF Radar itself to be controlled by SOUNDER. The remote DIO hardware consists of a central unit, located with the HF Radar, and up to 8 remote units.

RDIOT task commands are:

DELAY n,m — Set delay factors for the remote DIO test loop. The remote DIO can be tested at different speeds. The defaults n,m are 60,0. The delay factors specify the number of times to loop through a delay routine. The delay loop time for an Interdata 7/16 is 5.00 microseconds and is 5.25 for an 8/16. The second delay (m) was found to be unnecessary as hardware development progressed and is normally set to 0. The first delay factor in SOUNDER's DIO loop is 60; the second delay has been eliminated from SOUNDER.

EXER [n] — Exercise remote unit n (28-2F hex). If n is not specified, remote unit 28 is tested. EXER runs through 65536 passes, and each pass tests all bit patterns in the 8-bit data byte. For each pattern, the data are sent to the remote unit, read back and compared to the test pattern. Errors are stored in an error table, which may be examined with ERRORS, below. The error table can hold 255 error entries, and when full, EXER halts. The EXER loop can also be stopped at any time by typing any character. When EXER is running, the pass number, modulo 256, is displayed in the high order two hex digits of the Interdata panel lights; the low order two digits display the error count.

ERRORS — Display the pass number, expected data, and data read from remote unit for errors detected during last EXER run.

END — Terminate the RDIOT task.

10.2.14. TSGINT 2-00 [DCW]

Test the TSG interrupt and associated software. At start up, TSGINT asks for the TSG paces as strapped on the TSG board. The response, if the TSG is standard, is

10,20,100,1000

The values are in milliseconds and correspond to the binary pace codes 00, 01, 10, and 11 in bits 1 and 2 of the TSG command byte (0=MSB). If pace code 11 invokes an external interrupt, enter a value of 0 for that pace, unless the external interrupt source is connected and operating.

TSGINT task commands are:

RUN [nn] Repeatedly start the TSG with TSG command byte nn. The value of nn is rejected unless the high order bit is on, indicating a timing sequence scenario request. (If the high order bit is off, the command byte is interpreted as a TSG page read/write request.) If nn is not given, use the last named command byte (default 80).

TSGINT enters an endless loop that starts the TSG by sending it the specified command byte via Supervisor Call 1 (SVC1), which invokes the TSG driver at the MT2 OS level. After starting the TSG, TSGINT reports errors, if there are any. In case the TSG start results in a FEP start, TSGINT clears the FEP DATA READY signal before waiting for the TSG DONE interrupt. If there are no errors sensed by the SVC1 TSG wait call, the loop counter is displayed in the hex display panel lights, and TSGINT loops again. The SVC1 TSG start and wait calls are identical to those made by SOUNDER; TSGINT issues no calls to the FEP or EKO drivers. The TSGINT loop may be stopped at any time by typing (cr).

PACE Test the TSG paces against the values entered at start up. The TSG paces are timed by counting TSG DONE interrupts/second for each pace in the testing sequence. The default testing sequence goes through all combinations of the four paces. Other sequences can be set with PSEQ, below. PACE runs until terminated with (cr) or until NPASS (below) is satisfied, at which time any errors detected are reported. If 0 was entered for binary pace code 11 at TSGINT start up, that pace is ignored to avoid filling the error table with TSG timeout errors.

ERRORS — Reexamine any errors detected by the most recent PACE test.

PSEQ <list> — Set a new sequence for the PACE command. The numbers in the argument list, <list>, are 0, 1, 2, or 3, corresponding to the binary pace codes 00, 01, 10, and 11. For example, PSEQ 2,1,3,2 instructs PACE to test the 100, 20, 1000, and 100 millisecond paces and to repeat the sequence NPASS times, or until (cr) is typed. If <list> is absent, the current sequence is displayed.

NPASS n — Set a pass count for the PACE command. PACE will go through its sequence table n times. The default count is infinite.

RESET — Reset the TSG. This command is included so that TSGINT need not be exited to BOOT TSG from SOUNDER, in case the TSG times out. If RESET is given, however, the TSG may still have to be BOOTed from SOUNDER, if other than the PROM zero page is needed, in order to sound with the correct TSG scenarios.

(cr) — Stop the loop. If a RUN or PACE loop is in progress, it is stopped; otherwise the (cr) is ignored.

END — Terminate the TSGINT task.

10.2.15. PIPLN 2-00 [DCW]

PIPLN tests the basic HF Radar TSG-FEP-EKO loop and associated driver software. PIPLN enters an endless loop that is a simplified version of SOUNDER's (P1.0) pipeline, with an identical sequence to the main events. The pipeline sequence in P2.0 eliminates the wait and start FEP calls in order to save some time in the pipeline loop. (If the FEP DONE, or FEP DATA READY, interrupt occurs before the SELCH is started for the EKO transfer, starting the SELCH starts the EKO transfer; if however the SELCH is started before the FEP DONE, the EKO transfer does not occur until the FEP DONE interrupt is sensed. In either case, no explicit control of the FEP from the SOUNDER task level is required.)

Running PIPLN can help clarify whether a TSG timout error (8237) from SOUNDER is in fact caused by the TSG or is caused by the FEP not signaling DONE to the TSG.

The PIPLN task test sequence is:

Start TSG
delay
Wait for EKO DONE
delay
Wait for FEP DONE
delay
Start FEP
delay
Start EKO
delay
Wait for TSG DONE
delay

The delays are operator-selectable and load the loop to simulate time loading in the HF Radar software. The device starts and waits are made with SVC1 calls, identical to SOUNDER's. SOUNDER and PIPLN SVC1 calls invoke the same TSG, FEP, and EKO drivers at the MT2 OS level. Start and wait errors are reported as they occur. The loop counter is displayed in the hex display panel lights. The PIPLN loop may be stopped at any time by typing (cr).

10.2.15. PIPLN 2-00

PIPLN task commands are:

RUN [nn] — Run the PIPLN loop using TSG command byte nn. The value of nn is rejected if it invokes a TSG page read/write request (high order bit set to 0). If nn is not given, use the last named command byte (default A5).

(cr) — Stop the loop, if in progress, else ignore the (cr).

SETD — Specify the loop delays (discussed above) in decimal, integer milliseconds. The delays are only approximate. The total of the six delays should not exceed the time value of the TSG interrupt selected. The operator is led through the SETD command; delays are associated with the specific start or wait that they follow.

DELAYS — Display the currently specified delays.

END — Terminate the PIPLN task.

10.2.16. FEPEKO 2-01 [DCW]

(For details about changes made to FEPEKO 2-00, refer to Appendix A, dated 1983 November.)

Task FEPEKO tests the FEP and the data transfer from the FEP to the Interdata via the EKO device. FEP programs can be debugged interactively in object (microcode) format from the console; the FEP output can be listed and plotted.

At start up, FEPEKO displays the decimal number of FEP locations it can store for a NEW FEP program. (See command, NEW, below.) FEPEKO's NEW program buffer is considerably larger when FEPEKO runs stand-alone and can hold a full 1024 (decimal), or 2000 (octal), FEP program locations.

Trouble with the DMA transfer of the FEP data can be at several different levels. FEPEKO provides a series of commands that effect the FEP data transfer in a variety of ways. They are listed below in most- to least-primitive order. (Before a transfer of FEP data can be initiated, it must be set up by a start to the TSG. (See STTSG, below.)

READ — Read data from the FEP directly, 16 bits per read. The SELCH and the system software drivers are bypassed by this most primitive of reads. READ's first request is: "SENSE STATUS BIT:". Valid responses are:

12	Check busy/done bit after read.
14	Check EOM bit after read.
15	Check device unavailable bit after read.
(cr)	Ignore status, but delay between reads.

If (cr) only is typed, READ requests, "DELAY IN MSECS.:". Respond with the decimal number of milliseconds to wait between 16-bit reads. READ next requests, "NO. OF HALF-WORDS:". Type 0-1024 (decimal); 0 means ignore the READ command. READ reads directly from device 77 (FEP/EKO) using the read 1/2-word (RH) instruction; the status is checked or the program waits between reads, as specified. The data are displayed when READ is satisfied.

BLOCK — Read a block of data from the FEP. BLOCK waits for FEP DATA READY true, then reads the first 16 bits of data with a read 1/2-word register (RHR). The first 16 bits read contain the number of 1/2-words to be transfered in the block. BLOCK continues to read with RH, checking device 77 status bits 14 (EOM) and 12 (busy/done), until EOM is true or until the 2048-byte buffer is full. The block read terminating condition (EOM SENSED or BUFFER FULL) is displayed and if EOM was sensed, the data are displayed.

SBLOCK — Read a block of data from the FEP through the SELCH. After stopping the SELCH, the SELCH status is read. If it is not 0, the error is reported and FEPEKO requests another command. If the SELCH is ready, SBLOCK sets up a transfer of 2048 bytes by writing its buffer start and end addresses to the SELCH. SBLOCK waits for FEP DATA READY true, then starts the SELCH, initiating the data transfer. SBLOCK then waits until SELCH DONE is true. The data transfer count as read from the SELCH and the data transfered are then displayed.

SVCRD — Read a block of data from the FEP through the SELCH via a call to the system EKO driver (SVC1). This is the manner in which SOUNDER reads the data from the FEP via the EKO device.

SVCRD requests, "NO. OF HALFWORDS:". The maximum transfer is 1024 (decimal) 1/2-words. The transfered data are displayed.

The following commands assume that the DMA EKO device is functioning properly. They are used to initialize the TSG and FEP programs, effect a data transfer, and to examine the data.

BOOT TSG[,xxx]
Boot the TSG. This command functions exactly like its counterpart in SOUNDER (Chapter 8), except that the TSGMEM files are assumed to be experimental and are kept on SYSC.

BOOT FEP[,xxx]
Load the FEP program memory. Like SOUNDER's BOOT FEP, except the FEPMEM files are on SYSC. If desired, program xxx becomes the active program

for interactive debugging. (See GET and PUT, below.)

FEP n	Set the FEP thresholding factor. (See FEP, Chapter 8.)
FEP n,c,k	Pass a count, c, and a constant, k, to FEP location n. The count and constant must be less than 1024, and n must be 2-4. (See Chapter 8 for standard usage.)
STTSG [nn]	Start the TSG with command byte nn, or if nn not given, with the last named command byte. The default initial command byte is 86.
EKO	Execute a full data transfer from the FEP via the SVC1 call.
RESET	Reset the FEP program location counter and clear the FEP DATA READY. This is exactly the FEP reset that SOUNDER and the FEP verify routines use.
LIST	List the last data transferred from the FEP as the result of READ, BLOCK, SBLOCK, SVCRD, or EKO. The complete data buffer is listed to the console, based on the 1/2-word count in 1/2-word 0 of the data transfered.
LIST n	List 1/2-words 0 through n-1, or n 1/2-words.
LIST n,m	List m 1/2-words, starting with 1/2-word n. A list terminates if the end of the transferred data is encountered before m is satisfied.
PLOT	Plot the last data read from the FEP. The buffer 1/2-word position is used as the X-coordinate; the value of the 1/2-word is the Y-coordinate. Thus, the data are plotted without regard to their format. In other words, this is a quick plot. Figure 10.10 shows an obvious problem with channel 2.
X n,m	Limit plot to 1/2-words n through m. The defaults and limits of X are 0-1023. The range, m-n must be at least 10.

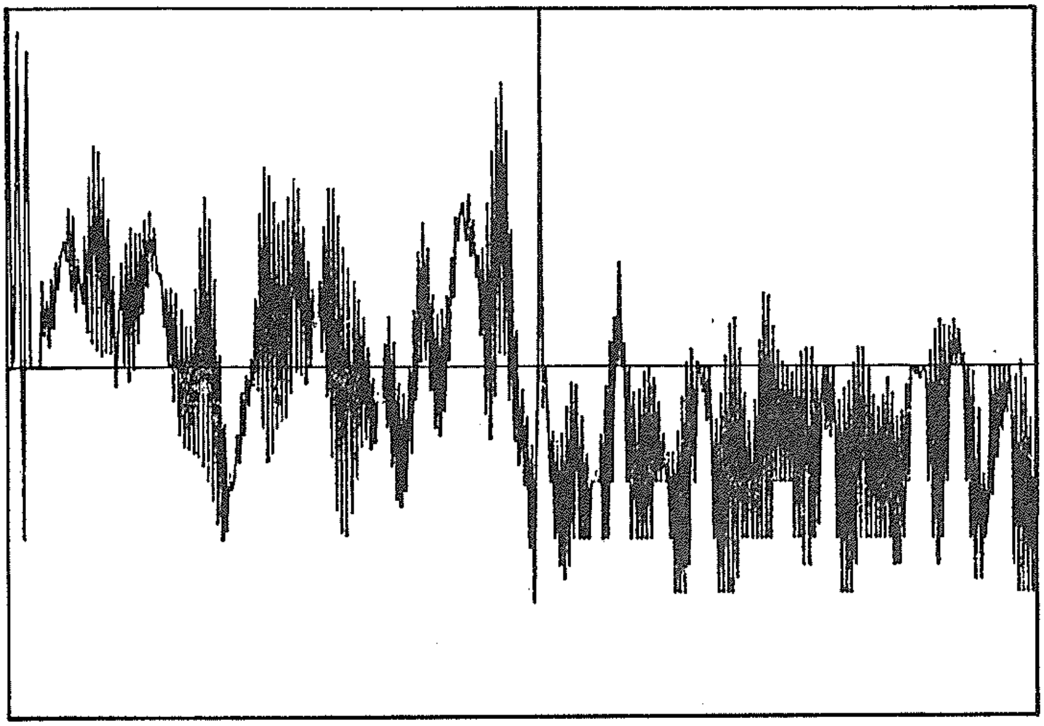


Figure 1Ø.1Ø. FEPEKO PLOT DISPLAY.

Y n,m — Limit plot to 1/2-words with values between n and m. The Y defaults are -2Ø48 through +2Ø47 (the FEP ADCs are limited to 12 bits); the Y limits are -32768 through +32767.

Off-scale values are plotted on the top or bottom borders; the horizontal zero-line floats between top and bottom.

The following commands are used to debug the FEP or FEP programs:

NEW — Initialize for a new FEP program or program segment. A NEW program is built in FEPEKO memory and can be the active program for interactive debugging. (See GET and PUT, below.)

NEW is overridden by another NEW or by BOOT FEP[,xxx]. NEW is terminated by SAVE, below.

SAVE POx — Append NEW program or program seqment to SYSC:FEPMEM.POx (x is any alphanumeric character). POx remains the active program and can be edited in the FEP and on the disc. (Only POx disc files can be changed by FEPEKO.)

GET n — Get location n (0-1777 octal) of the currently active program for editing. If the program has been made active by BOOT FEP, or if a NEW program has been SAVEd, location n is read from the disc. If a NEW program has not been SAVEd, location n is either fetched from or initialized in FEPEKO memory. In any case, location n is displayed in field, or object, format and can be edited with a command of the form:

<label> n

where <label> is the label, or field, name as displayed, and n is the octal value to be placed in the field.

A field edit command does not update the location in FEPEKO memory, in FEP program memory, or on the disc. The location is held in an editing buffer until it is PUT.

PUT — The last named, or active, location n (from GET n) is written to the FEP program memory. If the active program is still NEW (not yet SAVEd), location n is updated in FEPEKO memory. If the active program is on disc (from either BOOT FEP or SAVE) the location is updated on disc, but only if the FEPMEM extension is POx. The PUT location remains available for further editing until NEW, GET, EXAMINE, BOOT FEP, or SAVE is typed. In other words, after PUT, a STTSG-EKO-LIST sequence can be followed by <label> n, on the previously PUT location.

EXAMINE — The currently active program is displayed in field format. This list can be directed to a disc file or to a line printer, if available, with the SET LOG <fd> command to MT2.

EXAMINE n — Display location n (0-1777 octal) of the active program.

EXAMINE n,m — Display locations n-m of the active program. The locations need not be in ascending numerical order in the active program.

FEPEKO also recognizes:

PAUSE — Pause the FEPEKO task.

END — Terminate the FEPEKO task.

To summarize, a FEP program is made active for debugging by BOOT FEP[,xxx], or NEW. A NEW program that has not yet been SAVEd is edited in the FEP and in FEPEKO memory. (Remember to PUT the location.) A SAVEd program remains active, and is edited in the FEP and on disc as file SYSC:FEPMEM.POx. A BOOT program is edited in the FEP, but on disc only if BOOT FEP,POx was given.

Note that SAVE appends the NEW program or program segment to SYSC:FEPMEM.POx but does not write to the FEP as does BOOT or PUT. Therefore, all or part of program POx may be in the FEP after the SAVE command. To write the whole of program POx to the FEP, BOOT FEP,POx.

If called from SOUNDER, FEPEKO uses ET.ABUF through ET.ABUF+X'7FF' for its LIST/PLOT buffer and ET.ABUF+X'800' through ET.LBUF for its NEW buffer. If running stand-alone, FEPEKO's LIST/PLOT buffer is X'A000'-X'A7FF'; the NEW buffer is X'A800'-X'D7FF'.

Although SOUNDER limits the EKO transfer to 520 1/2-words (SNDR, Section 7.7), FEPEKO, for debugging purposes, allows a full 1024 1/2-words.

Experimental FEP programs created with SAVE must be copied from SYSC to SYS1 if they are to be made operational.

10.2.17. SXREF 2-00 and GXREF 2-00 [DCW]

Tasks SXREF and GXREF cross-reference the global symbols for multi-moduled sounding tasks SOUNDER and GRAPHER, respectively. These CALL tasks are useful for debugging and for finding one's way through the assembly listings. They read through all of the respective tasks' object (.OBJ) disc files, noting all ENTRY and EXTRN labels by PROG (module) label. SXREF and GXREF detect both unreferenced ENTRY labels and unresolved EXTRNs.

SXREF and GXREF are indentical except for their object file name tables, FNAME. If modules are added to or deleted from SOUNDER or GRAPHER, FNAME must be updated accordingly.

SXREF and GXREF commands are:

BUILD — Build global labels' table. This table is built by default when the task runs; it can be rebuilt if disc errors, etc., occur.

<entry> — Display the home module of ENTRY label, <entry>, and the PROG labels of all referencing modules.

UNREF — List all unreferenced ENTRY labels by PROG, or module name.

LIST [<prog>] — List all ENTRY labels for module, <prog>, and all PROG labels of modules that reference them; and list all EXTRN labels with their home module PROG names. If an EXTRN label is unresolved, the home PROG name is "******". This form of the LIST command writes to the console. If the argument, <prog>, is absent, LIST writes the entire cross-reference listing of the sounding task to the line printer. Installations with no line printer can modify the "LPR:" references in SXREF and GXREF to "CON:", the console.

END — Terminate the SXREF or GXREF task.

10.2.18. XREF 2-00 [JRW]

Task XREF, a modification of GXREF, above, is provided by J. R. Winkelman as a general-purpose cross-referencing program. It functions like the tasks described in Section 10.2.17 with the following exceptions:

1. Unlike SXREF and GXREF, XREF can read library files, including those from FORTRAN.

2. XREF reads the names of files to be cross-referenced during the BUILD command. In response to XREF's request, "FILE>", type any volume, file name, and extension. The default file extension is ".OBJ". If the volume is not specified, the default volume is assumed. After all file names have been entered, type (cr) in response to "FILE>".

3. COMMON definitions appear as ENTRYs, but no references to these COMMON blocks are listed.

All other XREF commands function exactly like those of SXREF and GXREF (Section 10.2.17). XREF can run under SOUNDER as a CALL task, but very nearly fills GRAPHER's memory partition. For this reason, XREF should probably be run stand-alone. (See Section 10.2.)

The user may wish to change NLBL EQU 880, the number of program labels which can be kept. NPRG EQU 175 defines the maximum number of references to any program. Neither table is checked for overflow, but have enough room for FTN3.LIB.

RUN CXREF runs a stand-alone version with the improbable limits of, NLBL EQU 3500 and NPRG EQU 500.

10.2.19. HKDIO 2-00 [DCW]

Task HKDIO displays the HKADC and the DIO readings from a selected disc data set. (See HKDIO, Sections 2.8.1, 4.3, and 4.9.)

Task HKDIO cannot run stand alone.

HKDIO task commands are:

DISC n	Get data set for previous sounding n. The data set number, n, is the same as for the GRAPHER command GPION (Chapter 8). 0 means the data set just recorded, 1 means the one before that, etc. DISC reads through the data set selected (via MANAGER) to find the HKDIO record. For long data sets this may take tens of seconds.
HKADC	Display the House-Keeping ADC readings. The pre-sounding channels (0-31) are displayed first, followed by the post-sounding channels (0-63), four channels/line. The readings are displayed both as signed voltages and 12-bit hexadecimal ADC counts.
DIO	Display the post-sounding DIO readings of channels 0-63 in hexadecimal.
END	Terminate the HKDIO task.

10.2.20. HKADC 2-00 [DCW]

Task HKADC 2-00 exercises the House-Keeping ADC unit and is fucntionally identical to HKADC 1-00.

The HKADC task commands are:

SELCH n	Select a channel, 0-63. (decimal).
CHAN	Display current channel number selected.
DATA	Read the currently selected channel.
RDADC n,m	Read channels n through m. For example, 10.,15. reads channels 10-15 (decimal), inclusively.
CHLOOP	Read specified list of channels. Each channel is read 10 times (round robin) following a delay in ms specified in response to the task's request.
END	Terminate the HKADC task.

All HKADC readings are displayed as signed voltages and ADC counts (hex).

10.2.21. LOG 2-00 [DCW]

Task LOG displays in various ways the disc file to which COUNSEL writes all messages that are logged to LU2 as described in Section 6.5. Task LOG assumes this file to be SYS1:LOG, and when run, reads the log file index and displays it by session number, a session being defined as everything that happens between HFRADAR and CANHF.

LOG task commands are:

INDEX — Display the log file index. This is the same display presented at LOG start up.

SELECT — Select messages to be displayed. The arguments to the SELECT command are task identifiers (C for COUNSEL, S for SOUNDER, etc.) and message weights (0-9). Zero to four arguments can be given in any order, with or without comma separators. The argument list, AC23, selects all A2, A3, C2, and C3 messages, if any; A0Z3 selects messages with weights 0-3 for all tasks. If no task IDs are given, all tasks are selected; if no weights are specified, all are selected. SELECT(cr) is equivalent to SELECT AZ09, and selects all messages. All messages are selected by default.

CLIST [n] — List selected messages for session n. If n is not specified, list all sessions. CLIST lists selected messages in chronological order.

BLIST [n] — List n selected messages in reverse chronological order. If n is not given, list selected messages from the entirety of the latest session. (The latest session is not the current session, if LOG is running under SOUNDER. The log index is only updated by COUNSEL when the command CANHF is typed to COUNSEL.)

CONT [n] — Continue with BLIST, listing n selected messages, or if n is not given, listing selected messages from the entirety of the next latest session.

END — Terminate the LOG task.

10.2.22 RECOV 2-00 [DCW]

Task RECOV recovers from a GRAPHER or CALL task pause. Such a pause is usually caused by an arithmetic error in trying to scale bad data in 16-bit, signed integer form. RECOV can also be used to get out of an interminable display or plot. The word RECOV is typed to MT2 (the * prompt must be seen) either after a pause or an escape. SYS1:RECOV.CSS cancels the currenty troubled task in GRAPHER's memory partition then loads and starts SYSC:RECOV.TSK. This task immediately requests to be canceled by SOUNDER. SOUNDER then swaps GRAPHER for RECOV and the COUNSEL prompt is seen.

It is left as an exercise for the user to rename SYS1:RECOV.CSS to something more imaginative. Your choice of a name might reflect your level of frustration at seeing "ARITH ERR 70DC", etc. Decency prevents me from making suggestions of my own.

RECOV should not be used in the middle of a sounding. In such a case GRAPHER might get queues out of order and go belly up with nothing gained.

Also, RECOV should not be used during the execution of a GPION or GPSKY display. Doing so would leave MANAGER with an opened data set.

10.3. Sounding-mode CALL Tasks [DCW]

Section 10.2 described Idle-mode CALL tasks. This section discusses those CALL tasks that can occupy GRAPHER's memory partition during an actual sounding. (See SOUNDER command, CALL, Chapter 8.)

If a task is placed on CALL to run at sounding time, SOUNDER loads the task into GRAPHER's paritition when the RUN command, initiating a sounding or a series of soundings, is given. The task remains in GRAPHER's partition for a single sounding or, if an SPH schedule has been specified, until SN is satisfied or an AB is typed. (See commands SPH and SN, Chapter 8.)

Note that SOUNDER, when the CALL command is given, does not check the argument of the command to see if the task exists on SYSC; a task existing at CALL time might have been deleted by RUN time. If SOUNDER cannot find the CALL task at RUN time, COUNSEL displays "ERR 9".

None of the sounding-mode CALL tasks described in this section can run stand-alone. All of them recognize CANHF, or the C04 queue, from COUNSEL. They also all recognize SOUNDER's sounding-start (S04), data buffer available (S02), and sounding-end (S06) queues.

10.3.1. TENUS 2-00 [DCW]

Task TENUS copies the EKO data exactly as received from the FEP, together with the PCT associated with the data, directly to magnetic tape. Tape files written by TENUS can be used as input for SOUNDER's TPR command (Chapter 8), and by CALL task PLOTEKO (Section 10.2.7).

A slower than usual pulse rate is required to allow TENUS to write tape in real time. If TENUS cannot keep up with the data at the specified pulse rate, SOUNDER will insert TSG NOPs between pulse sets, and will report "LOST PULSES" at the end of the sounding. Lost pulses do not mean that any data have been lost; only the specified time resolution is affected.

When first started, TENUS sends a TFF queue to SOUNDER to release the console and to request the use of all of SOUNDER's EKO buffers (TFF+4 = 5). TENUS then waits for an S04 queue from SOUNDER, signaling that a sounding has begun. After clearing the S04 queue block status, TENUS gets the extension of the TSGMEM disc file for the current sounding mode from SOUNDER's SCT and opens the TSGMEM file for input. (If in the unlikely event that TENUS can not find the TSGMEM file on SYS1, the task logs an error message and waits for the operator to type AB. If TENUS is ABorted, SOUNDER reinstates GRAPHER and everything returns to normal. Nothing is written to tape.)

TENUS then copies the SCT from SOUNDER into its tape write buffer, logically ORs hex 50 to SCT.MODE and writes the SCT to tape. (TPR will only accept tape files as input if the sounding mode in SCT.MODE is flagged with hex 50.)

TENUS next reads the zero-page from TSGMEM record 0, and writes it to tape. TENUS then gets the address of the DIO pulse-to-pulse data from ET.ADIO (Section 7.3) and writes DIO.PTP to tape twice, making a total of four control records at the start of the tape file. (Four records make it easy for PLOTEKO to position the tape.)

TENUS then waits for S02 queues from SOUNDER. When an S02 queue is received, TENUS gets the address of the PCT and EKO buffer from the S02 queue block and writes the data to tape. The size of the tape record is determined by the size of a PCT, currently 10 bytes, and the number of 1/2-words in the EKO data, found in the first 1/2-word of the EKO buffer. (The PCT and EKO buffer are contiguous in SOUNDER memory.)

At sounding end, indicated by the SØ6 queue from SOUNDER, TENUS writes two EOF marks to tape and backspaces over the second EOF, leaving the tape positioned for another TENUS or normal sounding. Thus, a TENUS-written tape contains one sounding per file with double EOFs indicating the end-of-information.

TENUS does not allow for multiple reels. Tape errors, if any, are logged at the end of each sounding. A TENUS sounding can be ABorted at any time.

Because the magnetic tape unit is attached to MANAGER during normal sounding, it is the operator's responsibility to avoid conflicts between MANAGER and TENUS. MANAGER and TENUS data sets can be mixed on a given tape, but the CLOSE tape command should be given before running TENUS, if FORMAT > 1 was given. (See commands TAPE, FORMAT, and CLOSE, Section 4.7.) In other words, an EOF mark must precede a TENUS tape file.

TENUS does not terminate itself at the end of a sounding, but remains in GRAPHER's memory partition until swapped out by SOUNDER. After each sounding, TENUS waits for another SØ4 queue. If an SPH schedule is active, SOUNDER will terminate TENUS if an AB is given, or when SN is satisfied. If no SPH schedule is in progress, TENUS will occupy GRAPHER's partition for one sounding only.

Notice that TENUS provides a mechanism, albeit indirect, for getting the TSGMEM timing scenarios to tape. A minimum length (single pulse) sounding mode can be set up with CALL TENUS and BOOT TSG,xxx that when run will guarantee that the zero-page from SYS1:TSGMEM.xxx will be written to tape.

1Ø.3.2. MEANS 2-ØØ [DCW]

Tasks MEANS and MENUS (Section 1Ø.3.3) were especially written for a partial reflection experiment attempting to see returns at very low ranges. These tasks were used in an attempt to rid the data of phase coherent echoes, or "ground clutter".

Task MEANS runs like task TENUS, but records no data. Instead, MEANS sums the X and the Y values for both receiver channels and all ranges over the course of a sounding; then at SØ6 time, MEANS divides these 32-bit sums by the number of SØ2 queues received during the sounding, to produce 16-bit means for the X and Y values for all ranges and for both channels. These 16-bit means are written to a contiguous disc file, SYS1:MEANS.SAV, which must exist before a MEANS sounding is run. The MEANS.SAV file is used as input by MENUS, below.

Task MEANS only computes means of data from sounding modes P or G, or their variants. (See MAKEM, Section 1Ø.2.1.)

1Ø.3.3. MENUS 2-ØØ [DCW]

Task MENUS runs exactly like TENUS (Section 1Ø.3.1), except that at start up, MENUS reads the table of means computed and written to disc file SYS1:MEANS.SAV by task MEANS (Section 1Ø.3.2). Then, when an SØ2 queue is received from SOUNDER, MENUS subtracts the means from the EKO data before writting the PCT/EKO buffer to tape.

Like MEANS, MENUS expects data from sounding modes P or G, or their variants. MEANS and MENUS should be run in sequence, and should use the exact same form of the FEP 3,Fc,Fk command. (See Chapter 8.)

Tape files written by MENUS can be played back through the HF Radar sounding tasks with command TPR to establish them as standard data sets on the disc. These data sets can in turn be examined by EXOUT (Section 1Ø.2.6) and by TPLOT (Section 1Ø.2.8).

10.4. Other Utility Tasks [DCW]

Sections 10.2 and 10.3 describe test and support tasks that are categorized CALL tasks, indicating that they can temporarily displace GRAPHER as one of the HF Radar tasks. Section 10.4 covers all other utility tasks provived with P2.0. These tasks can only run stand-alone, and therefore are not integral parts of the HF Radar software system.

10.4.1. DUTIL [DCW]

Disc utility program DUTIL enables the user to read, search, modify, and write disc sectors, and to examine the names of files, file indexes, and volume directories on the disc.

DUTIL bypasses all MT2 OS software disc protections and must be used with extreme caution.

To run DUTIL, type DUTIL to MT2. This command invokes file DUTIL.CSS, which sets partitions, loads, and starts DUTIL. If a message reveals that file DUTIL.CSS is unavailable, type:

```
SE PA 1/F000
SE PA 1/7000
LO DUTIL
T DUTIL
ST
```

All numerical input to DUTIL is hexadecimal; leading zeroes are not needed and are ignored. All values displayed by DUTIL are hexadecimal. Only devices DSC1: (top or removable disc) or DSC2: (bottom or fixed disc) may be specified as the current DEVICE.

DUTIL's sector and file commands are discussed below. Other DUTIL commands are:

*<text>	Enter comment or documention line. DUTIL ignores command lines beginning with the asterisk.
CURRENT	Display currently attached (current) device, sector number, volume name, volume directory start sector number, and the volume bit map start sector number.
DEVICE	Specify new current device. Type DSC1: or DSC2: in response to DUTIL's request.

10.4.1. DUTIL

PREAMBLE — For 7/16's only. PREAMBLE destroys all curre data on the device specified. DUTIL formats the device (DSC1: or DSC2:) specified in response to its request by writing each sector as 256 bytes of test pattern X'A5' with its 4-byte preamble. The device formatted becomes the current device.

PAUSE — Pause the DUTIL task. To reenter DUTIL, type:

T DUTIL
CONTINUE

END — Terminate the DUTIL task.

DUTIL's sector commands enable the user to read, search, modify, and write sectors on the disc specified by DEVICE, above.

The current sector is the last sector read, written, transferred, or zeroed, i.e., the sector currently in the memory buffer.

DUTIL's sector commands are:

DISC — Display the sectors used and the sectors free on the current device as indicated by the disc bit map. DISC also displays the size of the largest contiguous block of free sectors.

READ n — Read sector n and display its contents. Sector n becomes the current sector.

SEARCH <list>

Search the current sector for the 1/2-word values in the list, <list>. Up to 8 values may be specified. When a match is found, "(+DD)=HHHH" is displayed, where DD is the hex byte displacement of the matching 1/2-word value, HHHH. DD is always even. If no matching values are found, there is no display.

If <list> is not given, the entire contents of the current sector are displayed sequentially, one 1/2-word per line.

MODIFY DD,<list>

Modify the current sector with the hex bytes in the list, <list>, beginning with byte displacement, DD. The list may contain up to 16 bytes, separated with any non-hex character. When the modification is complete, the contents of the current sector are displayed.

EXAMINE — Display the contents of the current sector. The display is of the current sector as represented in memory; it is not changed on the disc until the WRITE command is given.

ZERO n — Write zeroes to sector n. Sector n becomes the current sector.

XFER n — Transfer (write) the current sector from memory to disc sector n. Sector n becomes the current sector.

WRITE — Write the current sector back to disc. If there is no current sector, nothing is written.

DUTIL's file commands enable the user to obtain file name and file parameters. In addtion, the hexadecimal editor provides means of modifying contiguous files, and NEXT finds file index and directory blocks.

DUTIL's file commands are:

FILES — List all files residing on the current device.

FILES name.xxx
List the parameters of the specified file.

FILES name.-
List the parameters of all files with the specified name and all existing extensions.

FILES -.xxx
List the parameters of all files with the specified extension.

FILES - — List all files with null extensions.

FILES -.- — Same as FILES(cr).

10.4.1. DUTIL

HEXEDIT name.xxx

Enable hexadecimal edit of the specified file, which must be contiguous and must reside on the current device. After the edit mode is entered, type:

HHHH

where HHHH is the byte address of the half-word cell to be modified. DUTIL responds by displaying:

+HHHH/ CCCC

where CCCC are the current contents of cell HHHH. To modify the cell, type

NNNN (1)

where NNNN are the desired new contents of cell HHHH. To leave cell HHHH as it is, type only a carriage return:

(cr) (2)

To open the next half-word cell for editing (byte address HHHH + 2), type line feed followed by carriage return instead of a carraige return at (1) or (2):

(lf)(cr)

DUTIL will indicate when a modified sector has been written on the disc. At any time (cr)(cr) is typed, the edit mode and HEXEDIT terminate.

NEXT Read consecutive sectors, starting with the next sector, until a file index or directory block is found, and display the block sector. Such a block is one that has the first two words zero. This routine was written by J. R. Winkelman to assist in the repair of directory chains.

Each 5-megabyte disc has a Volume Directory (VD) that contains the information required by the file manager (FMGR) of the operating system (OS) to access files on the disc. The VD itself is contructed somewhat like an indexed file in that each VD block contains the pointer to, or the sector number of, the next VD block in the chain. Sector number zero points to the first VD block, which is always 30. The last VD block in the chain points to zero. The VD is initialized to use the following sectors: 0 through 5A in steps of 6, with 5A pointing to 31; 31 through 5B in steps of 6; 32 through 5C by 6; 33 through 5D by 6; 34 through 5E by 6; 35 through 5F by 6, with 5F pointing to 0.

Should the VD become full, FMGR assigns additional blocks, as needed, and appends them to the end of the VD chain. These blocks are assigned as available from the disc bit map, so their sector numbers are unpredictdable. For example, if the bit map reveals that sector number 3B0E is the next availabe block, the end of the chain will have 5F pointing to 3B0E, which in turn points to 0. In this way the chain is lengthened as the number of files grows, with the last VD sector always pointing to 0.

The first 4 bytes of a VD block contain the pointer to the next block. The first two bytes should always be 0 because the largest sector number is 4C7F (19583 decimal). Occassionally, a VD block gets written back to the disc with a non-zero value in the first two bytes. When an attempt is made to access a file beyond such a bad VD block, an EOM error results. DUTIL can be used to correct the bad sector in order to reconstruct the VD block chain. If a file known to exist cannot be accessed or listed, DUTIL's NEXT command can be used to read through the disc until the break is located and repaired, a long and tedious process.

10.4.2. LION and TAPIR [JRW]

LION (LIst IONogram) lists to the printer and as such is not of general utility. It is, however, an often-used program. LION was written to determine what is on a tape for debugging. It was not conceived to be used by anyone else. No replacement program using more proper formats has been written.

The format is mixed decimal and hex to display a P02 record. This is a terrible format for the SCT. LION has been expanded to print compact formats also. The formats may be interspersed on tape. It has not been updated to read the two sector SCTs.

The first line of each record is a record number in decimal and the length of the tape record in bytes. The first number on the following line is the first word of the P02 block and is the length of usable data (normally shorter than the actual record length). It is used to determine the number of lines to print so as to suppress echoes that don't belong with this data.

The remainder of the second line of each record is the contents of the four PCTs. The first word in each PCT is decimal. The remaining lines within the record are the echoes with the X,Y pairs in hex and the range in decimal.

TAPIR is a printing program that simply lists any file, 24 hex words per line. The last line in each record has 24 words regardless of the actual length of the data.

The primary use of these two programs might be to provide the base for your own listing programs by replacing the printer subroutine with one which will run locally.

10.4.3. FEPM [JRW]

FEPM creates a disc file containing a FEP program to serve as input to FEP load routines. The output of FEPM is used by CALL task FEPEKO (Section 10.2.16) and by SOUNDER to load the FEP program memory. The output from the FEP assembler, FEPASM, is used as input to FEPM.

Each line of input contains 17 octal fields separated by spaces. The first field contains the FEP program location number. The remaining fields contain the data loaded into that FEP location. The relationship between the 17 fields and the 12 byte output record is complex and confusing. See the FEP Verify document for a description of this relationship. Each line of input supplies the information for loading one FEP program location. The input lines may be in any order, and any FEP program location can be loaded singly.

The output disc file is indexed and has 12 bytes per record. The output records are the 12 bytes expected by the FEP load routines. Input format and disc file errors are reported.

To run the FEPM task, invoke the disc command file FEPM.CSS by typing:

FEPM xxx

Input is from xxx.ASM, if it exists, otherwise from xxx.LSM. Output is to FEPMEM.xxx, the file invoked by the BOOT FEP,xxx command to either FEPEKO or SOUNDER.

There is no command input to FEPM.

10.4.4. LFIL [JRW]

LFIL produces alphabetical listings, to the line printer (LPR:), of the system disc files. The listing device is specified in LFIL.CSS and can be changed to suit a given installation. The general form for running LFIL is:

```
LFIL [voln:][filename][.xxx]
```

Any and all of the arguments are optional as for the MT2 OS command, FILES. For example, the form:

```
LFIL(cr)
```

will produce two alphabetical listings of all the files on the default volume. The first listing is ordered primarily by file name and secondarily by extension; the second is ordered primarily by file extension and secondarily by file name, with files with null extensions listed first.

```
LFIL <filename>
```

lists all files with name <filename> on the default volume.

```
LFIL .<xxx>
```

lists all files with extension <xxx> on the default volume.

```
LFIL SYS<x>:
```

lists all files on volume SYS<x>.

```
LFIL SYS<x>:<filename>
```

lists all files with name <filename> on volume SYS<x>.

```
LFIL SYS<x>:.<xxx>
```

lists all files with extension <xxx> on volume SYS<x>.

LFIL can sort and list up to 500 files. (A heavily used volume at SEL contains less than 400 files.) LFIL creates a file, FILES (defined in the CSS), which contains the directory used by the sort and list program.

10.4.5. TSKT1T3 [LDM]

TSKT1T3 is a CSS file which loads three tasks, then starts one of them. Normally, only one of the three tasks runs at a time. To change the task currently running, use pause, reset the task and start. TSKT1T3 must run stand-alone, since the partitions are not the standard partitions used by HFRADAR. The source code for these three tasks and other similar tasks is in several files MAT70x.CAL and MAT80x.CAL, where x is a small number. There is no command input.

MAT80x code uses direct I/O without using the SELCH or the SVC1 commands. It is used mainly for checking out the CRT hardware. MAT70x code uses various forms of the SVC1 call and is used for checking out the CRT driver, SELCH and hardware. TSKT1T3 is normally used only when there are problems or for maintenance. Most of the tasks have sufficient NOPs to allow easy patching. Some have extra parameter blocks for checking out alternative forms of I/O. Others allow easy patching to control output data rates.

1Ø.5. Subroutine WRCON [DCW]

The job of displaying formatted lines on the console from an assembly language program ranks somewhere between a time-consuming chore to a full-blown nightmare, depending upon what needs to be done. Subroutine WRCON was written to simplify this job. It is included here because all of the CALL and utility tasks make liberal use of it. WRCON is the only external subroutine in use by the CALL and utility tasks that is external to the HF Radar sounding software. It resides by itself on the disc as WRCON.CAL; command file TETRUN.CSS, used to build CALL tasks (Section 1Ø.2), includes WRCON.OBJ after the object file of the main task code. Programs or modules that make subroutine calls to WRCON require the source code statement:

```
EXTRN   WRCON
```

to permit TET, the task establisher, to provide the subroutine linkage. The EXTRN statement must precede any references to WRCON within the calling module's code. WRCON is not included in the HF Radar sounding software, although SNDRDISP has a modified version of WRCON within it.

WRCON outputs to logical unit 1, normally assigned to the console. LU1, however, can be assigned to any device (TTY:, MAG:), or disc file that accepts 76-byte ASCII records. If LU1 has not been properly assigned, or if any write error occurs, WRCON stops with the message:

```
END OF TASK 111
```

Other error conditions are discussed below.

WRCON writes with SVC1 using a function code of 29 (hex). The SVC1 call is suspended until the output device is free; the calling task is not rescheduled to run until the write is complete. Writing is in image format, suppressing the task ID that otherwise appears on every console line. The buffer in which WRCON composes output lines is 74 bytes long. WRCON appends (cr) and (lf) (required by image formatting) when appropriate so that output records are from 2 to 76 bytes long.

The basic calling sequence is:

```
 BAL    R15,WRCON              WRITE TO LOGICAL UNIT 1
 DC     A(format)              LINE FORMAT ADDRESS
[DC     ARG1,...,ARGn          ARGUMENT LIST]
 DC     15                     ARGUMENT LIST TERMINATOR
 (next statement)
```

Only the argument list is optional. Format or argument list programming errors that could kill the system cause

END OF TASK 15

The line composed up to the argument in error is written before the task stops. Errors that are not potentially fatal simply result in scrambled output; the task is allowed to continue. General registers 0-14 are returned to the user intact; R15 will contain the address of the argument list terminator plus 2.

A WRCON call argument can be either a register number (0-14) or a memory address; the register or memory 1/2-word can contain numerical data to be formatted, an integer count, or the address of an ASCII string or sub-format.

The WRCON format is an ASCII string in the calling task's memory, defined by a program statement, such as:

```
FMT      DB     C'TEXT',0
```

The above statement defines 5 bytes at displacement FMT in the module. The 5 bytes are the hex representations of the ASCII characters 'TEXT' followed by a null (0) byte. WRCON scans the format from left to right; the null byte flags the end of the format for a given WRCON call and is required. If a format is too long to fit comfortably on a line, an equivalent to the above is:

```
FMT      DB     C'T'                 STORE "T"
         DB     C'EXT',0             STORE "EXT" NULL
```

There is no practical limit to the number of continuation statements; the text is stored contiguously but must be terminated with a null byte. The label, FMT, need not be on a 1/2-word boundary. The assembly language sequence:

```
         BAL    R15,WRCON
         DC     FMT,15
CONT     NOP (or other executable statement)
```

writes the ASCII characters "TEXT" to logical unit 1 and then continues execution at CONT. If LU1 is the console (CON:), the writing begins at the cursor position. WRCON presets its composing or output buffer to spaces (20 hex). As it scans the format string, byte-by-byte, the character picked up is put unchanged into the next output buffer byte position unless it is the up-arrow character (↑, 5E hex). ↑ is the flag used to request a special function ("bell or whistle"). ↑ can be written, but only indirectly (see ↑C, below). The function mnemonic immediately follows the ↑ and is one or two upper case alphabetic ASCII characters, the virgule (/), or the left or right parenthesis.

Though most functions refer to the argument list, some do not. Those that do, advance an argument list pointer in WRCON that must, at format end, point to the argument list terminator, 15 (F hex). After the WRCON call is satisfied, program execution resumes just after the 1/2-word containing 15. For a given WRCON call, the number of ↑-flags in the format that invoke a reference to the argument list must equal the number of argument list entries.

In the detailed descriptions below, functions that refer to the argument list are flagged with "Arg. list". WRCON distinguishes between registers and memory addresses in the argument list by their magnitudes. WRCON output buffer character positions are referred to below as columns; columns are numbered 0-75. The numeric value or ASCII string passed to WRCON via an argument list entry is referred to as the object of the argument. For more detailed information about assembly language program statements, refer to the Common Assembler Language (CAL) User's Manaul and to the 16-bit Processor User's Manual.

(The examples on the following pages are not presented as examples of good coding practice, but to demonstrate the WRCON functions in simple form.)

↑D
Arg. list

The object of the argument is a numeric value to be written in decimal ASCII. Five column positions are used; leading zeros are suppressed. The following example produces the string:

____1st_day_of_JAN,_1977

where an underscore means a space.

```
            LIS    R3,1                 1 IN REG 3
            BAL    R15,WRCON            WRITE LINE
            DC     FMT,R3,YR,15
            . . .
YR          DC     1977                 CONSTANT 1977
FMT         DB     C'↑Dst day of JAN,↑D',Ø
```

↑L
Arg. list

The object is numeric to be written in decimal ASCII. Leading zeroes and leading blanks are suppressed. The number is left justified. In the above example, if the first function was ↑L instead of ↑D, the following string would be produced:

1st_day_of_JAN,_1977

↑RDnn
Arg. list

The object is numeric to be written in decimal ASCII. Only the rightmost (low order) nn digits are written; nn itself is Ø1 through Ø5. Leading zeroes are written. The following example produces "DAY_ØØ1,_1979".

```
            LIS    RØ,1                 +1 IN REG Ø
            BAL    R15,WRCON            WRITE LINE
            DC     FMT,RØ,YR,15
            . . .
YR          DC     9
FMT         DB     C'DAY ↑RDØ3,197↑RDØ1',Ø
```

If nn > Ø5, Ø5 is used; if nn = ØØ, no digits are written.

↑H
Arg. list

The object is numeric to be written in hexadecimal ASCII. Four columns are used; leading zeroes are written. The following code produces the line "ØØ64_HEX_is__1ØØ_DEC.".

```
              BAL    R15,WRCON
              DC     FMT,D100,D100,15
              . . .
D100          DC     100
FMT           DB     C'↑H HEX is↑D DEC.',0
```

↑G
Arg. list

The object is numeric to be written in hexadecimal ASCII. Leading zeroes are changed to blanks. If in the above example, ↑G were used instead of ↑H, the line "__64_HEX_is__100_DEC." would result.

↑RHnn
Arg. list

The object is numeric to be written in hex ASCII. Only the rightmost (low order) nn digits are written; nn is 01 through 04. Leading zeroes are written, so ↑RH04 is equivalent to ↑H. If nn > 04, 04 is used; if nn = 00, no digits are written. The example below writes "64_HEX=100_DEC".

```
              LH     R14,D100              100 IN REG 14
              BAL    R15,WRCON
              DC     FMT,R14,D100,15
              . . .
D100          DC     100
FMT           DB     C'↑RH02 HEX=↑RD03 DEC',0
```

↑C
Arg. list

The object is an ASCII string to be copied into the output. The object string is formed like a format statement and must be terminated with a null byte or (cr) (0D hex). The code produces:

The_↑_flag_can_be_written_indirectly.

```
              BAL    R15,WRCON
              DC     FMT,STR,15
              . . .
STR           DB     C'The ↑ flag can b',0
FMT           DB     C'↑Ce written indirectly.',0
```

The format string(s) and the argument object string(s) can be concatenated in any fashion.

↑W
Arg. list

The object is a two-byte (16-bit) ASCII string to be copied into the output. No terminator is required. The code for "ABCDEFG" is:

↑W

```
         LHI   R7,C'EF'          "EF" IN REG 7
         BAL   R15,WRCON
         DC    FMT,S1,R7,15
         . . .
FMT      DB    C'A↑WD↑WG',0
S1       DB    C'BC'
```

↑/ Terminate line with a carriage return and a line feed. This function has no argument object. Multiple ↑/ functions can appear in a format; multiple lines can be written with one call to WRCON. (In all of the preceding examples, the console cursor would be left at the column following the last character written.) To write:

```
LINE 1
LINE 2
```

use the following code.

```
         BAL   R15,WRCON
         DC    FMT,15
         . . .
FMT      DB    C'LINE 1↑/LINE 2↑/',0
```

WRCON clears its output buffer to blanks (20 hex) after each ↑/ and resets the column number to 0.

↑Tnn Tab to column nn, which becomes the next column used. (Remember that the first character position is column 0.) For example:

```
LINE 1
     LINE 2
          LINE 3
```

```
         BAL   R15,WRCON
         DC    FMT,15
         . . .
FMT      DB    C'LINE 1↑/↑T05LINE 2↑/↑T10LINE 3↑/',0
```

The column number, nn, is a two-digit decimal integer 00 - 74. Tabbing need not be forward. For example, "1234.5" can be written by:

```
                 BAL   R15,WRCON
                 DC    FMT,N,N,15
                 . . .
N                DC    12345
FMT              DB    C'↑D↑TØ4.↑RDØ1',Ø
```

"12345" is first copied to the output buffer; the 5 in column 4 is over-written with a period.

The line written is terminated at the last column number used. For example, "12", the high order two digits, can be written by:

```
                 BAL   R15,WRCON
                 DC    FMT,N,15
                 . . .
N                DC    12345
FMT              DB    C'↑D↑TØ2↑/',Ø
```

"12345" is first copied to the output buffer; the "34" in columns two and three is over-written with (cr) and (lf). The output record is four bytes long.

↑TA
Arg. list

The object of the argument is an integer column number. ↑TA works just like ↑Tnn except the column number is a variable that can be computed. For example:

```
LINE 1
     LINE 2
          LINE 3
```

```
                 LIS   R2,5                    TAB, LINE 2
                 LIS   R3,1Ø                   TAB, LINE 3
                 BAL   R15,WRCON
                 DC    FMT,R2,R3,15
                 . . .
FMT              DB    C'LINE1↑/↑TALINE 2↑/'
                 DB    C'↑TALINE 3↑/',Ø
```

Notice that a null byte terminates the format continuation statement only.

↑Snn

Skip over nn columns. For example, write the following pair of lines:

↑Snn

```
0123456789
*         *

        BAL    R15,WRCON
        DC     FMT,15
        . . .
FMT     DB     C'0123456789↑/*↑S08*↑/',0
```

Again, nn is a two-digit decimal integer from 00 through 74.

↑SA
Arg. list

The object is the number of columns to skip. ↑SA functions like ↑Snn except that the number of columns skipped is variable. For example, write the following pair of lines:

```
0123456789
*    *    *

        LIS    R5,4                    4 IN REG 5
        BAL    R15,WRCON
        DC     FMT,R5,R5,15
        . . .
FMT     DB     C'0123456789↑/*↑SA*↑SA*↑/',0
```

↑I
Arg. list

Ignore the argument. ↑I is useful if one call to WRCON is used for several purposes. For example, "TEXT" is produced by:

```
        LHI    R0,FMT2                 ADDRESS OF FMT2
CALL    STH    R0,FMTAD                TO ARG LIST
        BAL    R15,WRCON
FMTAD   DC     0,R13,15
        . . .
FMT1    DB     C'PAGE↑D',0
FMT2    DB     C'TEXT↑I',0
```

"PAGE 1" is produced by:

```
        LHI    R0,FMT1                 ADDRESS OF FMT1
        LIS    R13,1                   1 IN REG 13
        B      CALL                    WRITE PAGE NO.
```

↑Mnnc

Write the character c nn times, nn from 00 through 74. For example, produce the line "***** TITLE *****":

↑Mnnc

```
                BAL   R15,WRCON
                DC    FMT,15
                . . .
        FMT     DB    C'↑M05* TITLE ↑M05*↑/',0
```

↑MAc
Arg. list

Write the character c a variable number of times according to the argument object. For example, produce the same example as above:

```
                LIS   R0,5                5 IN REG 0
                BAL   R15,WRCON
                DC    FMT,R0,R0,15
                . . .
        FMT     DB    C'↑MA* TITLE ↑MA*↑/',0
```

The character c can be any printable non-control character except the single quote, which is used by CAL to delimit ASCII strings. ↑ is usable; (lf) might work (so far untested) but it won't reinitialize the output buffer as ↑/ does.

↑F
Arg. list

The object of the argument is another format string terminated by a null byte. When WRCON sees ↑F, it scans through the sub-format string to its null before continuing the scan of the format string in which the ↑F appears. Sub-formats can invoke their own sub-formats to a total depth of 5 levels. The following example produces "ABCDE":

```
                BAL   R15,WRCON
                DC    FMT,SUB1,SUB2,SUB3,15
                . . .
        FMT     DB    C'A↑FE',0
        SUB1    DB    C'B↑F',0
        SUB2    DB    C'C↑F',0
        SUB3    DB    C'D',0
```

Since SUB3 contains no ↑-flags, SUB2 could read:

```
        SUB2    DB    C'C↑C',0
```

↑Xnn

Execute a segment of the format nn times, nn from 00 through 99 (decimal). Since ↑Xnn (and ↑XA, below) must always be used with ↑(and ↑), which are used to delimit the format segment to be multiply executed, examples will follow the description of ↑).

↑XA Arg. list — The object of the argument is the repeat count for the multiple execution of a format segment. ↑XA functions like ↑Xnn except for how the repeat count is passed.

↑(— Begin multiply-executable format segment.

↑) — End multiply executable format segment.

Examples of ↑Xnn↑(...↑) and ↑XA↑(...↑) follow.

To write 15 blank lines:

```
         BAL    R15,WRCON
         DC     FMT,15
         . . .
FMT      DB     C'↑X15↑(↑/↑)',Ø
```

To write N blank lines:

```
         LHI    R9,N                 N = LINE COUNT
         BAL    R15,WRCON
         DC     FMT,R9,15
         . . .
FMT      DB     C'↑XA↑(↑/↑)',Ø
```

Sequences of multiply-executable format segments can be nested to 5 levels, but counting argument list references can get tricky. The following code produces these three lines:

```
XXXX__XXXX__XXXX__XXXX__
XXXX__XXXX__XXXX__XXXX__
XXXX__XXXX__XXXX__XXXX__
```

```
         BAL    R15,WRCON
         DC     FMT
         DC     ST,ST,ST,ST
         DC     ST,ST,ST,ST
         DC     ST,ST,ST,ST
         DC     15
         . . .
FMT      DB     C'↑XØ3↑(↑XØ4↑(↑C__↑)↑/↑)',Ø
ST       DB     C'XXXX',Ø
```

Notice that only ↑C refers to the argument list, but it does so 4x3 = 12 times. Thus the argument list contains 12 references to the string ST.

Finally, any meaningfull string of hex bytes can be sent as control to the Tektronix 4012 console. To clear the screen and home the cursor, for example:

```
         BAL    R15,WRCON
         DC     FMT,15
         . . .
FMT      DB     X'1B',X'C',0
```

Since WRCON writes to the console in image mode, one can also select the Tektronix 4012 GRAPH mode or the GIN (graphics input) mode. Refer to the Tektronix 4012 Computer Display Terminal User's Instruction Manual for specific details.

11. OS MODIFICATIONS and SYSGEN

James R. Winkelman

11.1. Introduction

This chapter describes the modifications that have been made to operating system OS16/MT2, Revision 3.

It was decided to use a standard operating system for the sounder software. The Interdata OS16/MT2 allowed concurrent tasks and had good communication among the tasks. However, the slower the operating system, the slower the soundings. The soundings are usually done at 10 millisecond intervals, unless conditions such as spread F or very noisy signals prevail.

Two approaches were taken to allow a sounding repetition time of 10 milliseconds: the FEP does more processing, which allows PICKER to do less, and OS16 itself was speeded up. The user can save processing time by typing: NOQUEUE 2 and [PE. These commands stop all queueing to GRAPHER and the scheduling of GRAPHER, as well as all memory references required by the refreshing of the display. GRAPHER and the display can be restarted by: [PS and GPION 0.

Revisions have been made to OS16 to allow it to run faster, to simplify operator interactions, and to help toward remote operation. The task priority sorting used 24% of the computer time, it now uses about 1%.

Setting SEL.ALL to one in the CUP file (see SYSGEN, Section 9.5) invokes most of the changes. The remainder of the chapter describes the changes in detail.

11.2. SEL.SAFE

SEL.SAFE EQU 0, default COMPROC*ST

This variable controls several conditions. It replaces and extends some testing made when CMPROC*ST EQU 0. In this situation, there is no command processor or the system is single-task and core-resident. Many tests are not made because there is little a user could do if an error were detected.

Tests made for parameter blocks or buffers starting on an even byte are examples of classes of tests that are deleted. A debugged program runs faster without these tests and that is the purpose of this selection.

11.3. SEL.DISK

SEL.DISK EQU 1, default SEL.ALL

When a disc is marked on, the directory block is not physically changed. In the original system, if a disc was marked on during a crash, it could not be marked on or off after the system was restarted, a level of safety required only by payroll files and the like, if then.

When the system is booted, all discs attached are marked ON or ON PROTECTED as indicated by hardware protection at the boot time. This is useful for remote operation and convenient to operators.

11.4. SEL.TIME

SEL.TIME EQU 1, default SEL.ALL

Time is displayed in the front panel lights in the format M DD HH MM SS. Only the last digit of the month is displayed.

11.5. SEL.IMAG

SEL.IMAG EQU 1, default SEL.ALL

The image mode to the console requires that the user provide the carriage return and line feed if they are desired. The system, however, provides prompts: an "*", the task name, and a ":" or ">". SEL.IMAG ≠ Ø requires that the user provide any prompts that may be wanted. This mode is useful for graphing and gives the user total control of the device while in image mode. There are no changes to the normal mode.

A binary read to the console device gives an absolute image mode. All eight bits are preserved, and NO checking is done for any special characters, including escape and carriage return. The read is completed only when the requested number of characters has been read.

11.6. SEL.FIL

SEL.FIL EQU 1, default SEL.ALL

The date and time of creation and last assignment are stored in the file directory. The file listing format is changed also.

The number of sectors used and free are printed as well as the longest string of contiguous sectors. A contiguous file longer than the number after CONT: cannot be allocated.

The format contains the file name, extension, type (INdexed or COntiguous), size (IN: bytes per record; CO: hex sector address of first sector; length is always 256 bytes per sector), keys, count of records (IN) or sectors (CO), directory address (the hex address of the sector which contains the directory of this file), creation date (MMDDYY HHMMSS), and date and time of last assignment while the disc was not protected.

11.7. SEL.CON

SEL.CON, default 72

The system assumes the length of a console line is either 72 characters or LOGLEN, whichever is longer. This version uses SEL.CON instead of the 72.

11.8. SEL.FEP

SEL.FEP EQU X'77', default Ø

A SINT SEL.FEP is given every 8.333 milliseconds (1Ø milliseconds for 5Ø hertz) if SEL.FEP is not zero. A sounding cannot be run in a system with SEL.FEP ≠ Ø. A set of test programs can be run, however, to measure the time used by OS16/MT2. SEL.FEP is only useful in special circumstances.

11.9. SEL.SVC

SEL.SVC EQU 1, default Ø

A default of SEL.ALL would have been preferred, but SEL.SVC must be defined long before SEL.ALL is available.

Two specialized SVC calls are used heavily by the system: SVC 9,- (wait for queue entry); and SVC 6,- (add to the queue of another task). Both of these calls are slow because they share numerous resources with other tasks.

SVC 8,- exactly replaces SVC 9,- and is about Ø.7 milliseconds faster. SVC 4,- replaces SVC 6,- (add to queue only) and is about Ø.8 milliseconds faster. SVC 6,- calls remain unchanged and must be used for any SVC 6 function except add-to-queue. Add-to-queue may be done by either an SVC 6,- or an SVC 4,-. The SVC 4 uses only the task name, the status, and the address of the Q-block.

11.1Ø. SEL.CMD

SEL.CMD EQU 1, default SEL.ALL

When the system is started, the command file STARTOSW.xxx is executed from SYS1 (default volume in the CUPS file) as a CSS file. The file extension xxx is the extension used to boot the system, typically .ØØ1 or .ØØ2. This CSS file can do anything from running a program to set the time of day clock from the TSG, to running a complete sounding using a file similar to HFRADAR.CSS. Remember that COUNSEL can be started with a command file ending with RUN or SRUN and CEND.

The command file is STARTOSW.CSS if the command processor is not overlayed by OVCMD, OVSIX, or OVSEVEN; or if there is no file manager by DELETE SVCSEVEN.

11.11. SEL.ALL

SEL.ALL EQU 1, default Ø

SEL.ALL is used as default for most of the other control parameters. It also controls some changes that are not separately controlled. Most of these changes are minor.

An additional SVC 2 command has been added. The parameter block is:

```
DB      OPTION,1Ø
DC      C'HH:MM:SS  MM/DD/YY'
DC      TIME,OFDAY
```

The 22 bytes must be in exactly the order and format given, including "/" and ":". The four bytes TIME,OFDAY are the second of the day as a full word integer. This integer should match the time given in ASCII. If OPTION is 1, the date given in this parameter block is the European format DD/MM/YY.

The system has been changed so that a PAUSE to a PAUSED or dormant task does not cause an error. The time and date are included in the heading line of FILES, useful in commands such as: FI .CAL,LPR:. There are three new operator commands:

DISPLAY REG Display the 16 registers of the currently selected task, four per line. The register memory address can be used by the MODIFY command to change a register.

ADDn is the memory address of register n
REGn is the contents of register n

D R					
Ø-3	ADDØ	REGØ	REG1	REG2	REG3
4-7	ADD4	REG4	REG5	REG6	REG7
8-B	ADD8	REG8	REG9	REGA	REGB
C-F	ADDC	REGC	REGD	REGE	REGF

ZAP file Delete a file, whether or not protected, independent of access keys! No sympathy will be offered for files unintentionally ZAPped.

XDELETE file Delete the file if it exists, but give no error if it does not exist. XDELETE file is equivalent to:

11.11. SEL.ALL

```
$IFX file
   DELETE file
$ENDC
```

11.12. SEL.SWAP

SEL.SWAP EQU 15, default 15*SEL.ALL; Ø for no SWAP code.

SEL.SWAP allows the swapping of all user memory, including tasks and partitions, between two states. The only difference to the user, who may wish to know which state he is in, is that the second state types only two-character task names for input and output to the console. For example, the VOLUME command types SYSVOL in one state and SYSV in the other.

Swaps occur for three reasons: the user types the command SWAP (see below); the program calls an SVC 2,11; or the previous SVC 2,11 timed out.

SWAP — This is a new command to the command processor. The current user memory is written onto the file SWAPFILE.ROL. If a swap has occurred previously, the user memory is read from the other half of SWAPFILE.ROL. If no previous swap had occurred, the user memory remains unchanged. Swap time-out is set to zero (no time out).

SYS1:SWAPFILE.ROL must be allocated as a contiguous file of 384 sectors on a 7/16 and 896 sectors on an 8/16 with extended memory supported. SYS1 may not be protected during a swap.

The new swap SVC 2 has the following parameter block:

```
DC      11,TIME.OUT
```

This SVC 2 causes an immediate swap and the SVC 2 does not exit until the user memory has been swapped back. TIME.OUT is the time in seconds until this user's memory is swapped back in. If TIME.OUT is zero, no swap back will occur until the operator types SWAP. A TIME.OUT of less than 6Ø seconds is not recommended.

The normal uses of this swapping would be a user (Fortran and editing) who is swapped out for a minute occasionally by an HFRADAR program that would use the SVC 2,11. If both users use the SVC 2,11, the time out has priority over the current user and might cause problems.

The user can tell which memory is being used by observing the system prompts. The prompts in the 'other' memory use only the first two characters of the task name.

The user is responsible for interactions between user memories in the use of I/O devices such as tape. The tape can be assigned to both users in an incompatible mode.

The system tries to allocate the console in a friendly manner. If no typing has been done recently, it is useful to give a carrige return to see what is going on. Any return within SEL.SWAP seconds, currently 15 seconds, of time to swap will cause an immediate swap. Any line taking less that 15 seconds to type will not be interrupted.

Care must be taken to close files in both user memories before protecting the disc using the memory. EDIT uses the files EDITTEMP.nnn, which may become numerous if EDIT crashes or is turned off without being cancelled. The user may tell which user memory he is in by observing the prompts given when tasks do input and output

11.13. SEL.COV

SEL.COV EQU 1, default SEL.ALL

The console software handlers are normally overlayed along with the other command processor (operator commands and CSS processing) overlays.

SEL.COV non-zero leaves the console handler programs in core memory between operator commands. A listing or graph on the console device will run much faster, because the handlers are not read from disc for every output line. The console is the only device which is not normally resident.

11.14. SYSGEN

System generation consists of defining the parameters for the system needed or wanted, and generating that system as OS16/MT2. A CUPS file (CUP Source) defines the system configuration.

The SYSGEN disc (SYSD) is nearly full and contains only those files that are necessary. The CSS files generally use SYS1 for scratch files. TET716 and TET816 put both SYSMAP and the OS16 file onto SYS1.

EXEC16.CAL, FMGR16.CAL, and CMDP16.CAL are the original files from Interdata. They may be deleted, yielding about 7ØØØ sectors. If SELS is used instead of SEL, files EXEC.CAL, FMGR.CAL and CMDP.CAL will have records of 7Ø rather than 8Ø bytes. These save 1Ø bytes*25ØØØ records, or about 9ØØ sectors; this reduction does not affect assemblies. If a listing is needed, one of the files may be made with SEL, assembled then deleted, and the next file made.

CUP xxx — CUP (Configuration Utility Program) starts with a source file CUPS and generates a CAL file CUP. No SELCH device is allowed for the display unit PLT:, because this results in an error (CUP doesn't run right). The command file CUP.CSS xxx converts the source file CUPS.xxx to the CAL file CUP.xxx, where xxx is the extension, normally an identifying number for the CUP file. Both files are on SYS1.

EDIT — The CUP file must be edited to include the SELCH control for PLT: and the busy flag for sharing the SELCH between PLT: and MAG:. The example is wordier than necessary for clarity. The line numbers are for this example only and will change for different configurations. The example uses Fn as the SELCH address and should be FØ for a 7/16 or F2 for an 8/16.

Operator responses are underlined. Note especially the underlining of required blanks.

```
*EDIT
.BG: OS/16 EDIT RØØ
.BG> GET CUP.7ØØ
.BG> TYPE \ PLT@\
```

```
.BG:    191      DC PLT@BSY,Ø,U42TRM,Ø
   Change busy flag from PLT to SELCH.
.BG> CHANGE \ PLT@\,\ FFFn\
.BG:    191      DC FFFnBSY,Ø,U42TRM,Ø
.BG>            (carriage return, only)
.BG:    192      DC X'4ØØØ',U42DVR
   Indicate that this device uses the SELCH.
.BG> CHANGE \4ØØØ\,\4ØØ1\
.BG:    192      DC X'4ØØ1',U42DVR
.BG> TYPE \PLT@BSY\
.BG:    2Ø2     PLT@BSY DC X'ØØ'
```

Insert SELCH address and device busy flag pointer.

```
.BG> INSERT
.BG> 2Ø2.Ø1    DC X'FnØØ',FF57BSY
.BG> 2Ø2.Ø2    (carriage return, only)
```

Insert the special SEL.xxx control parameters from Chapter 11, such as SEL.ALL and SEL.DISK . These are normally appended to the CUPS file after the END statement.

```
.BG> INCLUDE \CSS\,CUPS.7ØØ,\SEL.\-
```

Find the busy flag definitions, and insert device 57 (PLT).

```
.BG> TYPE \FFFnBSY\1
.BG:    282     FFFnBSY DC Ø
.BG> INSERT
.BG> 282.Ø1 FF57BSY DC Ø
.BG> 282.Ø2    (carriage return, only)
.BG> SAVE CUP.7ØØ
.BG> END
```

This CUP file is used to control the assembly of the systems programs.

CALSQ xxx — This file uses CUP.xxx and programs EXEC.CAL, FMGR.CAL, and CMDP.CAL to produce EXEC.xxx, FMGR.xxx, and CMDP.xxx. No listings are produced so as to reduce the assembly time from several hours to less than a half hour. Furthermore, there is not enough space on the disc for all of the listings.

TET716 xxx — EXEC.xxx, FMGR.xxx, CMDP.xxx and DSCDMA.OBJ (716 disc driver), SELLIB.OBJ (the radar drivers) and DLIB16.OBJ combine to produce system OS16Wxxx.xxx and map file SYSMAP.xxx.

TET816 xxx Like TET716 for the 8/16. It does not use DSCDMA or TRAPS for the floating point FIX, FLOAT, STME, and LME.

OS716 xxx This CSS does a complete system generation. It is similar to a combination of CALSQ and TET716. SYSD can remain protected, as all files except the source CAL files are on SYS1. Only files not already assembled are reassembled, making it easier to make changes in one file only.

The old file OS16Wxxx.xxx is not deleted before the new one is created. This saves the contiguous space, important if the disc is full. The new OS16W file is never shortened, even if it could be much shorter. If the new OS16W file needs to be longer, a TET error results and the object (.xxx) files are not deleted. Delete the OS16Wxxx.xxx and give another 'OS716 xxx,y,z'.

The second parameter is optional, but if not null skips the TET, similar to CALSQ. The third parameter, also optional, saves the object files which are otherwise deleted after a sucessful TET. Example: OS716 799,,SAVE generates OS16W799.799 and saves the object files for MAG716.

OS816 xxx See OS716 for a complete description. The two CSS files differ only in their TET command files. The 7/16 uses a different disc driver than the 8/16.

MAG716 MAG716, no arguments, generates a 7/16 bootstrap tape (see Section 11.15.) from the object files .799 generated by OS716 799,,SAV.

MAG816 MAG816 generates an 8/16 bootstrap tape in the same manner as MAG716.

The following is for people who want to do more systems work.

CALSYS CALSYS assembles a systems program as follows:

```
CALSYS name,xxx,output,options
```

where the name is normally EXEC, FMGR, or CMDP; and xxx is as described under CALSQ.

Output:
null uses name .LST, nonnull is the actual file or device. The listing may use as many as 8000 sectors.

Options:
assembly options, such as CROSS and NLSTC, separated by commas. An assembly with cross references may take 99 minutes.

Example:

```
CALSYS CMDP,718,,CROSS,NLSTC
```

CALSYS1 xxx The same as CALSYS except that the listing file is SYS1:NAME.LST. There is often no room on the system generation disc.

Editing files this large is difficult, slow, and requires a lot of disc space. The files are normally modified by use of "SOURCE UPDATER" which uses the sequence numbers for updating. Two local programs SEL and COMP simulate parts of "SOURCE UPDATER" and are described below.

SEL SEL does the SELECT function ten times faster than UPDATER. It has the following arguments:

```
SEL S,IN,OUT
```

S is the source operator file, normally small like E.035. IN is the input file like EXEC16.CAL. OUT is the output file like EXEC.CAL.

Method:

S and IN are read a record at a time. If the sequence number for IN is less than that for S, the record for IN is copied to OUT and a new record is read from IN; similarly for S less than IN. If the sequence numbers for S and IN agree, S is written, IN is forgotten, and new records are read from both S and IN.

S may be exactly a SELECT file for "SOURCE UPDATER". Any lines without sequence numbers are ignored.

Examples:

```
SEL  E.ØØ1,EXEC16.CAL,EXEC1.CAL
SEL  E.ØØ2,EXEC1.CAL,EXEC.CAL
```

This will result in the same EXEC.CAL file as:

```
SEL  E.ØØ1,E.ØØ2,E.Ø12
SEL  E.Ø12,EXEC16.CAL,EXEC.CAL
```

These examples reveal that the operation is associative:

(a op b) op c = a op (b op c)

The development disc will contain both EXEC16.CAL as received from Interdata and EXEC.CAL and E.Ø24 which produced it. New releases of changes will be to EXEC.CAL, that is, they will have to be applied to E.Ø24 and EXEC16.CAL or can simply be applied to EXEC.CAL. If you keep both versions, space will be at a premium.

COMP

COMP does the COMPARE function faster as well as keeping the files alligned by sequence number (an insertion does not result in all future lines being listed) and generating an output file that could be used in SEL to change one of the files to the other in some cases.

COMP has the following arguments:

```
COMP  IN1,IN2,OUT
```

IN1 and IN2 are compared and differences are printed with a 1 or 2 according to which file was listed. OUT is optional, but if it is present, it can be used in

```
SEL  OUT,IN1,IN2
```

to regenerate IN2 from IN1 except where lines of IN2 have actually been deleted. To delete using SEL, include a comment card with the sequence number to be deleted. The line remains but generates no code when assembled.

11.15. BOOTSTRAP, writing

A bootstrap tape is required to restart a system when all discs have been lost, or when you no longer want to risk that last disc.

The SYSD disc must be mounted and ready. The files used will be referred to as --.799, OS716, and MAG716 for booting the 7/16. Similar files named --.899, OS816, and MAG816 exist for the 8/16 and are used similarly.

CUPS.799 defines the configuration for the system generation (SYSGEN). The deletion of features results in a system which is small enough to reside entirely in core memory, with room enough to run most programs.

The CSS command: CUP 799 creates a CUP file, CUP.799, which corresponds to the configuration defined in CUPS.799. This CUP file needs only the 'INCLUDE /CSS/,CUPS.799,/SEL./-' editing before use. (See EDIT example, Section 11.14. If the system being generated does not support CSS files, edit with 'INCLUDE 1,CUPS.799,/SEL./-'.)

The command: OS716 799,,SAV assembles EXEC, FMGR, and CMDP to give EXEC.799, FMGR.799, and CMDP.799 on SYS1. OS716 only assembles those files which do not already exist. If the CUPS.799 file is changed, these files should be deleted so that the new versions will be assembled. OS716 then calls TET to generate the system OS16W799.799 . The assembled files are not deleted because the third argument (SAV) is not null.

Mount a tape, including the write ring. The command: MAG716 copies the relocatable loader RELLDR.OBJ to tape. It then assigns all the files required for system generation and calls the library loader LLDR16. The command file LLDR.CMD then copies all required files to tape and generates a load map on LU3, currently CON:. At the end of the map it will state: UNDEFINED: NONE. If no errors are found, the END OF TASK Ø message is given.

MAG716 then copies the file GETDISK.OBJ to tape and backspaces over the end-of-file mark. SAVDISK is loaded and the fixed disc assigned, equivalent to the command: SAVDISK 2. In response to the label request, enter a label of up to 4Ø characters and return. After the fixed disc (SYS1) has been saved, SAVDISK saves the removable disc (SYSD). Again, a label is required. After SYSD has been verified the program exits.

If you wish, mount SYSC and type SAVDISK 1 when the disc is ready. Continue as desired until all discs are saved.

11.16. CUPS.799 Description

A CUPS file defines the parameters for the system generation (SYSGEN). the CUPS file for the 7/16 Bootstrap will be described. The 8/16 file is similar. This set of parameters defines a system which is easier to use than the smallest possible usable system.

```
CONSOLE 16,2,CON:
```

This defines the teletype as the console in case the 4Ø12 is not working. 'CONSOLE 34,2Ø,CON:' defines the 4Ø12.

```
VOLUME SYS1:
```

The default volume is SYS1, as if 'V SYS1' had been typed after booting.

```
DEVICE 51,DØ,DSC1:,ØØ,C6
DEVICE 5Ø,D1,DSC2:,ØØ,C6
```

Define the two disc devices. Define the magnetic tape device next on SELCH FØ.

```
DEVICE 65,85,MAG:,FØ
UNITS 1Ø
PROC Ø
FLOAT Ø
```

Ten logical units are needed by the CAL assembler. Define the processor type and specify no floating point support in the system.

```
CLOCK 6D
FREQ 6Ø
```

Define a sixty Hertz clock, nice but not necessary.

```
CSS 2
FOREGRND 2
```

Two CSS levels will run most CSS files. One foreground program is needed, and two make CSS files more general.

```
DELETE SETLOG
DELETE PFREQ
```

Save some space by deleting logging and the logging buffers. Also delete the precision timer and the 'SVC 2,23' wait commands.

```
DELETE INITIAL
DELETE SVCSIX
DELETE SVCFIVE
```

Delete the start-up code, the system can't be restarted without rebooting. Delete overlay support, intertask communication, and the ability to load a program from a task.

```
END
```

End of the configuration definitions, the remaining definitions are edited into the CUP.799 file.

```
SEL.ALL  EQU 1
SEL.SWAP EQU Ø
SEL.SAFE EQU Ø
```

Include most of the SEL enhancements, but not SWAP or error checking.

```
* CUPS.799 JRW Ø9/28/82
```

11.17. BOOTSTRAP, using

Mount the BOOTSTRAP tape, it must be at load point.

Set the Boot Enable switch to 'disable', otherwise the disc is booted. Press INI on the front panel to reset all devices. I suspect that the console may occasionally have to be turned off and on also, but results are confusing.

SET-up for tape booting:

```
DATA  22   ADD  DATA  WRT
DATA  3Ø   ADD  DATA  WRT   WRT  WRT
DATA  5Ø   WRT
DATA  5Ø   ADD  DATA  D5ØØ  WRT
DATA  CF   WRT  DATA  43ØØ  WRT
DATA  8Ø   WRT
DATA  78   ADD  DATA  85A1  WRT
      use C5A1 instead of 85A1 on an 8/16
DATA  3Ø   ADD  DATA  RUN
      tape moved, or else it wasn't on line, or ? ? ? !
DATA FCØØ ADD  DATA  RUN
      the relocatable loader is started
```

The tape is loaded, several records read, and an OS start-up message typed on the console. Leave the tape positioned as is, if there is a start-up message, otherwise, you may have to start over. There is no computer clock running on the front panel. A 'D D' command will display devices to prove that the system is up. None of the SEL enhancements are is this version.

This system is stand-alone, not requiring any data on the disc drives. It uses no overlays and cannot use CSS files (back to basics).

11.17. BOOTSTRAP, using

Type the following commands to restore the system:

SE PA .SYS/FØØØ	sets partitions
LD MAG:	loads GETDISK from tape
O E	system in executive mode
AS 1,DSC2:	assign the fixed disc
AS 2,MAG:	assign the tape
AS 3,CON:	assign the console
ST	start GETDISK

GETDISK will copy a file from the tape to restore the disc. It will run the same as if the the GETDISK 2 command had been typed. The label for SYS1 is listed. If a new SYS1 is desired, unprotect the disc and type Y (yes). The console display will show the sector number being copied. Each record holds 3Ø (48.) sectors.

.BG: VERIFY	tape is backspaced to verify the disc
.BG: LABEL	label is retyped
.BG: END OF TASK Ø	good run, else non-zero

CL 1	close the fixed disc
AS 1,DSC1:	assign removable disc
ST	copy the next file, SYSD, if desired

Both discs have been restored. If there is a third file, a new disc can be mounted and written. The assignments remain the same, a START is all that is required.

After the last file has been copied from the BOOTSTRAP tape or any other SAVDISK tapes, type:

CL AL	close all assignments
O U	return to user mode

The system should be in the same state as when the BOOTSTRAP tape was written. Enable the Boot Enable switch, enter the '78' sequence on an 8/16, and boot it up.

11.18. Sample SYSGEN Definitions

Two sample CUPS SYSGEN definitions will be shown. Both can run soundings, but .7Ø2 supports floating point, is faster when not sounding, and starts just below address 6ØØØ. System .7Ø1 has everything overlayed, except those parts which are actually used during active sounding, and starts just below 4ØØØ, which leaves room for more analysis programs or bigger buffers.

```
     CONSOLE 34,2Ø,CON:             CONSOLE 36,2Ø,CON:
```

Define the console device. The 36 includes additional console graphics support for the 4Ø12.

```
     VOLUME SYS1:                   VOLUME SYS1:
```

The default volume after booting is SYS1. Next define the teletype and teletype reader (the reader adds six bytes to the system).

```
                                    DEVICE 16,2,TTY:
                                    DEVICE 81,2,ASR:
     DEVICE 51,DØ,DSC1:,ØØ,C6       DEVICE 51,DØ,DSC1:,ØØ,C6
     DEVICE 5Ø,D1,DSC2:,ØØ,C6       DEVICE 5Ø,D1,DSC2:,ØØ,C6
```

Define the two discs, using the DMA instead of a SELCH. Define the magnetic tape as a nine track 16ØØ bpi device on SELCH FØ.

```
     DEVICE 65,85,MAG:,FØ           DEVICE 65,85,MAG:,FØ
     DEVICE 24Ø,37,TSG:             DEVICE 24Ø,37,TSG:
     DEVICE 241,77,FEP:             DEVICE 241,77,FEP:
     DEVICE 242,57,PLT:             DEVICE 242,57,PLT:
     DEVICE 243,F1,EKO:             DEVICE 243,F1,EKO:
```

Define the four user devices. Note that PLT cannot define the use of SELCH FØ.

```
     CSS 2                          CSS 4
```

Two levels of CSS are enough for most CSS files. Use fewer and shorter buffers to save space for SET LOG.

```
     LOGLEN 4Ø                      LOGLEN 72
     LOGQUE 1Ø                      LOGQUE 12
     UNITS 1Ø                       UNITS 1Ø
     PROC Ø                         PROC Ø
     FLOAT Ø                        FLOAT 2
```

Ten logical units are required for each task, some of them, anyway. The processor is a 7/16 with high speed ALU (all the instructions) and has built in floating point.

```
     CLOCK 6D                       CLOCK 6D
     PCLOCK 6C                      PCLOCK 6C
     INTERVAL 1Ø                    INTERVAL 1Ø
     FREQ 6Ø                        FREQ 6Ø
```

Full clock support is required, with a wait interval of ten milliseconds, and a line frequency of 6Ø.

```
     FOREGRND 6                    FOREGRND 6
```

The sounding system requires six tasks plus background. SAFE tests that buffers start on even bytes and other such goodies. These tests are not needed in a fully debugged system, and take both space and time.

```
                                   SAFE
     HALTIO                        HALTIO
```

The HALTIO allows the console to be interrupted by a message from a task.

```
     DELETE INITIAL                DELETE INITIAL
```

The initialization code is only needed at boot time and is then overwritten by the background partition.

```
     OVTWO
```

Most SVC 2 subroutines are overlayed, making operator response and message formatting very slow. The SVC 6 overlays are almost never used, especially with the SEL.SVC EQU 1. SVC 7 subroutines are used by the file manager for assigning files and related activities. These subroutines are not used during a sounding, and require a lot of memory.

The command processor is overlayed whenever any overlay is specified, so no OVCMD specification is needed.

```
     OVSIX                         OVSIX
     OVSEVEN                       OVSEVEN
     END                           END
```

The END statement ends the CUPS file processing. The remaining instructions are edited into the CUP file, which is generated from this CUPS file. The same editing process adds SELCH support for the PLT. (See EDIT, Section 11.14.)

```
     SEL.SVC  EQU 1                SEL.SVC  EQU 1
```

Include the faster SVC 4 and SVC 8 commands.

```
     SEL.ALL  EQU 1                SEL.ALL  EQU 1
```

Include all the SEL enhancements not excluded below. The shorter system contains neither swapping capability nor error checking.

```
     SEL.SWAP EQU Ø                SEL.SWAP EQU 1
     SEL.SAFE EQU Ø                SEL.SAFE EQU 1
     * CUPS.7Ø1 JRW Ø9/28/82       * CUPS.7Ø2 JRW Ø9/28/82
```

12. TSG Programs

James R. Winkelman

12.1. Introduction

The Timing Sequence Generator (TSG) controls the timing and phase of the transmitted pulse and the timing of data recorded by the Front End Processor (FEP) for later processing. The TSG selects the type of processing to be done by the FEP.

The TSG acts as the master clock for the sounding system, keeping both time of day and accurate intervals for sequences of pulses.

12.2. Flow of Control

The operation of the TSG is entirely dependent upon control code contained in ROM or downloaded from the Interdata. The version described here is based upon the new ROM made in August, 1982; in combination with the TSGMEM files TOO or TWO. Earlier and later versions are similar, but not identical. The changes involve the use of page zero for subroutine addresses and the FEP start addresses used for repeat codes.

The TSG waits for a start command byte from SOUNDER of the following form:

```
80  Start command
 +    choose one (timing)
00  Start at the next 10 millisecond edge
20  Start at the next 20 millisecond edge
40  Start at the next 100 millisecond edge
60  Start at the next external timing edge
 +    choose one (repeat count)
00  Single pulse
08  Four pulses at above interval (in byte 17, TSG zero page)
10  Eight pulses at above interval (in byte 18, TSG zero page)
18  Sixteen pulses (actual value in byte 19, TSG zero page)
 +  N = scenario number (0 - 7)
```

The timing is to the next edge of the selected clock signal, not the time from the previous pulse. The selection of times available is made with jumpers on the TSG board. The 10, 20, and 100 millisecond intervals, mentioned above, are those most often used. The delay after a ten millisecond pulse before a 100 millisecond pulse, can be any multiple of ten milliseconds, depending on the number of ten millisecond edges since the last 1ØØ millisecond edge.

The repeat counts are defined on the zero page, see Section 12.4. and are under user control. The 4, 8, and 16 are settings commonly used.

Scenario number Ø is for timing only, no pulses or data control is done. Scenario 1 has been reserved for calculations of the X,Y system offsets. Scenarios 2 through 7 are defined by the user. Sounder treats scenarios 2, 4, and 6 as transmitting types; and scenarios 3, 5, and 7 as corresponding 'quiet' versions which are used for protected frequencies. The user requests scenario 4 from SOUNDER, but SOUNDER uses scenario 5 if the frequency is protected. This puts part of the responsibility of maintaining protected frequencies on the writer of TSG scenarios, but still prevents the casual user from sounding on forbidden frequencies. It is necessary to allow quiet scenarios if the sounder is used for frequency surveilance.

There are three defining bytes associated with each scenario number. The first byte is the location of the table controlling the RF vector. The second byte is the FEP vector which is output at the chosen timing edge, and then remains unchanged until the following timing edge. The third byte is the starting address for the FEP processing.

The bits in the FEP vector are used for several purposes, both input and output. Only four values are allowed for the user:

- 12 X,Y are sampled every 2Ø microseconds and stored in the FEP input buffer. Only PART type processing treats 2Ø microsecond data correctly.
- 1A X,Y are sampled every 2Ø microseconds, but the connections to the receivers are reversed, giving a 18Ø degree phase shift.
- 52 X,Y are sampled every 1Ø microseconds and stored in the FEP input buffer. This is the FEP vector used in almost every case.
- 5A X,Y are sampled every 1Ø microseconds with a 18Ø degree phase shift.

12.3. RF Vector Control

The TSG performs the following sequence of actions when it receives a start byte:

1. It waits for the required timing edge.
2. It sends the FEP vector.
3. It sends the RF vector bytes in the order and timing required by the scenario.
4. It waits for the FEP to finish processing the previous pulse.
5. It switches the input buffers with those of the FEP, and resets the buffer address.
6. The FEP start address plus one is sent to the FEP if this is the last pulse in a repeat sequence, or if it is the only pulse.
7. The FEP start address itself is sent if this is not the last pulse.

RF vector control consists of sending a series of RF vector bytes. The timing is done by a TSG subroutine with a loop exactly ten microseconds long. It takes thirty microseconds to output a new RF vector byte and restart the timing loop. This makes thirty microseconds the shortest interval which can be used, but any larger multiple of ten microseconds can be used.

The eight bit RF vector byte can define 256 different vectors, but only a few are useful because of the hardware configuration. The eight bits each control some functions. Some of the bits are active high, but others are active when low. (See SNRIO, Section 10.2.10.)

03 Enable the low power transmitter and record data in the FEP input buffer.
10 Key the transmitter with both transmitters enabled.
12 Enable the low power transmitter, enable the high power transmitter, and blank the receiver (disable the antenna multiplexer, set maximum attenuation, and reset the receiver overload switch).
13 Enable the low power transmitter. The low power transmitter must be enabled for 200 microseconds before it is keyed.
81 Key the calibrator and record data in the FEP buffer.
83 Record data in the FEP input buffer.
91 Key the calibrator only.
93 Wait for something to happen with everything turned off.

12.3. RF Vector Control

Phase shifts may be applied to the keyed signal. The following values should be added to vectors 1Ø, 81, or 91. Note the peculiar order.

ØØ No phase shift.
Ø4 Phase shift of 27Ø degrees.
Ø8 Phase shift of 9Ø degrees.
ØC Phase shift of 18Ø degrees.

A phase shift of 18Ø degrees (1C, 8D, and 9D) combined with a 18Ø degree receiver phase shift (FEP vector 1A or 5A) should result in the same receiver output as for unshifted pulses, but with opposite phase on interfering signals.

Additional attenuation may be required for strong E or F region echoes, if large gain is being used to look at the D region. The following values should be added to vectors 83 or 93.

ØØ No attenuation.
2Ø 2Ø dB fast attenuation.
4Ø unused, Ø dB.
6Ø 4Ø dB fast attenuation.

The RF vector control table contains an item count followed by several pairs of bytes; an RF vector byte, and a byte containing the number of ten microsecond intervals for the RF vector to be held.

12.4. Zero Page

The TSG memory is organized as 256 byte pages. The user can use pages Ø, 2, and 3. Page zero is the only normal user page. Pages 2 and 3 can contain special user subroutines, but have not been used except for debugging the TSG.

Figure 12.1 is a listing from the DUTIL utility program of a typical zero page. The first data line are bytes ØØ to ØF, the second line are bytes 1Ø to 1F, etc. The seven scenarios 1 through 7 are defined by three byte groups, starting at bytes Ø1, Ø4, Ø7, ØA, ØD, 1Ø, and 13.

	In Figure 12.1			In Figure 12.2		
Scenario	RF vector	FEP vector	FEP start	RF vector	FEP vector	FEP start
1	2Ø	52	Ø8	2Ø	52	Ø8
2	7Ø	52	2A	3Ø	52	1Ø
3	86	52	2A	46	52	1Ø
4	9Ø	52	48	5Ø	5A	ØB
5	BØ	52	48	66	5A	ØB
6	3Ø	52	ØD	3Ø	52	ØD
7	46	52	ØD	46	52	ØD

The bytes in 16 through 19 select four repeat counts, of which the first (in 16) must be Ø1. The program which does the pulse averaging works best with powers of two. This gives eight choices for the remaining three repeat counts: Ø2, 2.; Ø4, 4.; Ø8, 8.; 1Ø, 16.; 2Ø, 32.; 4Ø, 64.; 8Ø, 128.; or ØØ, 256.

The same repeat count should be used with both sets in complementary pulse coding, and this results in an effective doubling of the repeat count. Figure 12.1 has repeat counts of 1, 4, 8, and 16. Figure 12.2 has repeat counts of 1, 2, 4, and 8.

The table represented by Figure 12.1 is modified by the same program which was used to list it, DUTIL. The location of the zero page is given in the SIZE field of the FILES listing for TSGMEM.xxx . A new TSGMEM file can be made by CPY TSGMEM.xxx,TSGMEM.yyy . The commands required are: READ nnnn; MODIFY add,byte,byte,byte, . . .; and WRITE. HEXEDIT TSGMEM.xxx can also be used for this modification.

The example used READ 25EC. To change the repeat counts, type MODIFY 16,1,4,1Ø,4Ø to get counts of 1, 4, 16, and 64. The block will be updated on the screen after each MODIFY command, but the disk is not updated until a WRITE command is given.

Bytes below address 2Ø and above address DE have defined uses, and should only be changed by someone who understands these uses. Bytes 2Ø through DE can be used to define scenarios and as memory for programs in pages 2 and 3.

Pages Ø, 1, 2, and 3 are in consecutive disc sectors, starting with the sector given in the SIZE field. Page 1 is included for programming convenience, but is not loaded into the TSG memory.

12.4. Zero Page

M LSD D	Ø	2	4	6	8	A	C	E
Ø	ØØ2Ø	52Ø8	7Ø52	2A86	522A	9Ø52	48BØ	5248
1	3Ø52	ØD46	52ØD	Ø1Ø4	Ø81Ø	ØØØØ	ØØØØ	ØØ2Ø
2	Ø3Ø3	83FA	83FD	93ØØ	Ø1Ø3	93ØØ	ØØØØ	ØØØØ
3	ØAØ3	81Ø6	8312	1311	12Ø3	1ØØ6	12Ø5	931F
4	83FA	83E5	93ØØ	Ø4Ø3	81Ø6	83FA	83F7	93ØØ
5	ØAØ3	8DØ6	8312	1311	12Ø3	1CØ6	12Ø5	931F
6	83FA	83E5	93ØØ	Ø4Ø3	8DØ6	83FA	83F7	93ØØ
7	ØAØ3	81ØC	83ØC	1311	12Ø3	1ØØC	12Ø5	931F
8	83FA	83E5	93ØØ	Ø4Ø3	81ØC	83FA	83F1	93ØØ
9	ØCØ3	81Ø6	8DØ6	83ØC	1311	12Ø3	1ØØ6	1CØ6
A	12Ø5	931F	83FA	83E5	93ØØ	ØØØØ	ØØØØ	ØØØØ
B	Ø5Ø3	81Ø6	8DØ6	83FA	83F1	93ØØ	ØØØØ	ØØØØ
C	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ
D	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ
E	ØØØØ	ØØØØ	ØØ1D	F86C	F89F	F8D8	F8FE	F83C
F	F958	F9ØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ	ØØØØ

Figure 12.1

Pages 2 and 3 are loaded, but will only be used if the subroutine pointers are changed in page Ø.

The programmer can use pages 2 and 3 for programs and data as required. Subroutine links, least significant byte first, are given in page zero, starting at address E5. These addresses and the corresponding section number in chapter 5 of the TSG Manual, Volume 3 are:

F81D	PIN	5.4	PIA initialization
F86C	DSSP	5.6	Scenario set-up
F89F	RFST	5.7	Scenario run
F8D8	BRPT	5.8	unused
F8FE	NMIS	5.5	Decode command byte
F93C	IRQS	5.9	Service interrupts
F958	RUN	5.1Ø	RF vector timing

The pointers are to ROM pages F8 and F9. A user who changes one or more of the pointers to pages 2 or 3 can also use parts of the ROM, or insert some code and finish with the standard processing.

12.5. Simple Pulse Scenario

This example will use scenario 6, which is the same in both figures. Scenario 6 sends a 6Ø microsecond calibration pulse in a 24Ø microsecond calibration data recording interval. A 6Ø microsecond pulse is transmitted, followed by a short wait, and an additional 479Ø microseconds of data recording. The length of the data is 5Ø3 pairs in each receiver channel, for 5.Ø3 milli-seconds of received data.

Scenario 6 has its three byte definition pointer at zero page address 1Ø (3Ø, 52, ØD). The RF vector control sequence is at 3Ø, the FEP vector is 52, and the FEP starting address is ØD (ØE for each last pulse). This directs the FEP to process the pulse as uncoded, FEP PICKER.

The RF vector control sequence at address 3Ø is interpreted as follows:

ØA Ø3 Ten (ØA) word control table, 2Ø bytes excluding this length byte, wait 3Ø microseconds (Ø3).
81 Ø6 Sixty microsecond (Ø6) calibration pulse and record data (81).
83 12 Record (83) 18Ø microseconds (12=18.) of additional data. The filter delays are long enough to center this calibration pulse in the 24Ø microsecond interval.
13 11 Warm up the low power transmitter (13) for 17Ø micro-seconds (11=17.).
12 Ø3 Warm up the high power transmitter, finish warming the low power transmitter, and blank the receiver (12). Thirty microseconds are just long enough. This takes 2ØØ micro-seconds for the low power transmitter and 3Ø microseconds for the high power transmitter.
1Ø Ø6 Transmit a 6Ø microsecond (Ø6) pulse with no phase shift (1Ø).
12 Ø5 Leave transmitters on (12) for 5Ø more microseconds (Ø5) to allow for filter delays.
93 1F Wait 31Ø microseconds (1F=31.) with everything off (93) for the echo to return.
83 FA Record (83) 25ØØ microseconds of data (FA=25Ø.).
83 E5 Record (83) 229Ø microseconds more data (E5=229.).
93 Everything quiet until next time (93).

The transmit sequence, 13 11 through 12 Ø5, is common to all scenarios which transmit data. The 1Ø Ø6 command is changed to alter the pulse length and phase.

12.5. Simple Pulse Scenario

The protected frequency alternative RF vector control sequence (scenario 7) starts at 46. There is only a calibration pulse, no transmitted pulse.

Ø4 Ø3	Four commands (Ø4) in this sequence.
81 Ø6	Sixty microsecond (Ø6) calibration pulse and record some data (81).
83 FA	Record data (83) for 25ØØ more microseconds (FA=25Ø.).
83 F7	Record data (83) for 247Ø more microseconds (F7=247.). The 83 12 and 83 E5 from scenario 6 have been combined into an 83 F7.
93	All done, stop recording.

After either of these sequences, the FEP will be started at address ØD if the repeat count is not satisfied, or at address ØE if this was the last pulse.

12.6. Two Phased Pulse

The TSGMEM zero page for the two phased pulse is shown in Figure 12.2. The pulses are sent in pairs, or pairs with the same repeat count. The first pulse is scenario 4 (5Ø, 5A, ØB). This pulse inverts the phase for the keyed pulses and also inverts the phase at the receiver. This leaves the echo phase unchanged, but inverts the phase of any interference.

The RF vector control sequence at 5Ø is the same as the one at 3Ø described in Section 12.5, except that RF vectors 8D and 1C are used in place of vectors 81 and 1Ø for the inverted calibration and transmitter pulses.

The FEP vector 5A reverses the receiver inputs at the antenna multiplexer from what they would be with FEP vector 52.

The FEP start address of ØB does simple summing of the input buffers, but does not process the result. The PICKER task is informed that there is no data as this is part of a coded pulse set.

The second pulse uses scenario 6, described in Section 12.5. The input buffers are summed and divided by the accumulated number of pulses. The result is used as input to the FEP PICKER process (ØD).

Scenario 2 uses the same transmission as scenario 6, but uses EQR type processing on the final result.

M LSD D	0	2	4	6	8	A	C	E
0	0020	5208	3052	1046	5210	505A	0B66	5A0B
1	3052	0D46	520D	0102	0408	0000	0000	0020
2	0303	83FA	83FD	9300	0103	9300	0000	0000
3	0A03	8106	8312	1311	1203	1006	1205	931F
4	83FA	83E5	9300	0403	8106	83FA	83F7	9300
5	0A03	8D06	8312	1311	1203	1C06	1205	931F
6	83FA	83E5	9300	0403	8D06	83FA	83F7	9300
7	0A03	810C	830C	1311	1203	100C	1205	931F
8	83FA	83E5	9300	0403	810C	83FA	83F1	9300
9	0C03	8106	8D06	830C	1311	1203	1006	1C06
A	1205	931F	83FA	83E5	9300	0000	0000	0000
B	0503	8106	8D06	83FA	83F1	9300	0000	0000
C	0000	0000	0000	0000	0000	0000	0000	0000
D	0000	0000	0000	0000	0000	0000	0000	0000
E	0000	0000	001D	F86C	F89F	F8D8	F8FE	F83C
F	F958	F900	0000	0000	0000	0000	0000	0000

Figure 12.2

12.7. Coded Pulses

Scenarios 2 and 4 in Figure 12.1 define a complementary pair. The first pulse of the set, scenario 2 at address 70, sends two positive 60 microsecond pulses (one 120 microsecond pulse).

The processing at FEP address 2A does a deconvolution against two 60 microsecond positive pulses and saves the result (the data are summed to scratch memories). The data are not processed further.

The second pulse of the set, scenario 4 at address 90, sends a positive 60 microsecond pulse and a negative (180 degree phase shift) 60 microsecond pulse. The processing at FEP address 48 deconvolves the returning signal against a similar positive negative pulse pair, and sums the result with the deconvolution of the first pulse (double positive).

The resultant buffer is processed by the FEP PICKER program and sent to the Interdata computer.

13. FEP Programs

James R. Winkelman

13.1. Introduction

This chapter describes the FEP processing and the standard options available to the users.

The Front End Processor (FEP) is used to preprocess the data received by the sounder. Only one-fourth of the sampled data can be transferred from the FEP to a SOUNDER buffer. The programs range from programs which transfer a subset of the data, to programs which return fully processed data blocks which contain only the data which is normally recorded.

13.2. Header Block

The first sixteen words in a FEP buffer are common to all programs. An image of most of these words is kept in the top of Scratch Memory 1. These locations are used both for saving some of the data, and to tranfer the data to the output buffer in a block transfer. Addresses will be given for both the Scratch and Output Memories.

Scratch	Output	Contents
-	Ø	Total length of the buffer in bytes.
1761	1	Starting address of the second half of the buffer (in words, may have another use if the buffer is not divided).
1762	2	X-zero, channel 1, receiver offset.
1763	3	Y-zero, channel 1, receiver offset.
1764	4	Mean, channel 1.
1765	5	Maximum, channel 1.
1766	6	X-zero, channel 2, receiver offset.
1767	7	Y-zero, channel 2, receiver offset.
177Ø	8	Mean, channel 2.
1771	9	Maximum, channel 2.
-	1Ø	Peak of PICKED data or zero, channel 1.
-	11	Peak of PICKED data or zero, channel 2.
-	12	Amplitude of calibration pulses.
-	13	Revision date, MMDDY, in decimal.
-	14	Range, address of first data point, or zero.
-	15	Pointer to next range word, or zero.

13.3. Common Descriptions

All data used are first corrected for receiver offsets by subtracting the appropriate X-zero or Y-zero. The data output to the SOUNDER buffers has also been corrected. The X-zero and Y-zero used are recorded in the SCT for that sounding.

Amplitude: An estimate of the signal amplitude. For faster calculation, the program uses A + .5B, where A is the magnitude of the larger component and B is the smaller magnitude. The maximum error is six percent, and all 1000 amplitudes can be computed in about 1.2 milliseconds.

Maximum: The largest amplitude seen in that channel. The calibration pulse is not included in the maximum.

Mean: The average amplitude of the sixteen pulses following the calibration pulse. This data is expected to be the quietest within the pulse.

Threshold: The test value for rejecting data as uninteresting. The normal value is 2.5 times the mean plus 7.

Find peak: Find a local maximum in the data. The data around this local peak is likely to be of interest. The normal algorithm finds an amplitude larger than the amplitude on each side. This peak is ignored unless all these amplitudes are larger than the threshold. A peak is 'established' when the second amplitude after the peak is less than the peak amplitude. If either of the following amplitudes (third and fourth) is a peak and is greater than the first peak, it is saved instead of the first, but a peak is established without further search.

SCAN table: An intermediate table of up to eight X,Y pairs, including both a count of pairs and the average range associated with the pairs. The table is in order of increasing range, and is expanded as needed to insert a range from a later pulse, which was not already included. The current PICKER task has room for 28 ranges (assembly parameter). See PACK in the PICKER task for more information on selection of this data for output to tape.

13.4. XYZERO Processing

XYZERO sums the first 5ØØ Xs and Ys in each channel and divides the sum by 512. The four results, two from each channel, are small numbers so that division by the slightly larger number is unimportant, but division by 512 is easier than 5ØØ. These quotients are recorded as X-zero and Y-zero in the SCT. They are subtracted from every X and Y during processing to remove receiver offsets. The XYZERO program is always run as the first 'pulse' in a sounding, with all attenuations at maximum so that only the receiver and digitizer offsets should be measured. XYZERO sends a short buffer to SOUNDER to signal the end of processing.

13.5. SNDR Processing

SNDR is similar to the first program written for the Phase One NOAA sounder. It finds four amplitudes above threshold, and then saves the four previous X,Y pairs, all pairs above threshold, and finally saves four more X,Y pairs after the amplitude drops below threshold. The two channels are processed independently; channel 1 is saved in the first half of the buffer, channel 2 in the second half.

The first data block for each channel contains sixteen calibration X,Y pairs. The range (word 14) is zero, and the pointer (word 15) is 48. This combination of zero, non-zero, is used by PICKER to identify SNDR processing. After the calibration has been saved, the mean and threshold are computed. The rest of the data is scanned until either the end of input data is reached, or the output buffer is full. The first half of the buffer is full when 256 words have been filled, including the header block. The buffer is full when 512 words have been filled. Channel 2 can fill any words not used by channel 1. The pointer to the next data block is initially zero and is filled after the last X,Y pair is saved, when it becomes the address of the next output word. The pointer is left zero when the program reaches the end of that buffer half without reaching the end of data.

The PICKER task scans each data block to find and establish a peak. The range and data pair associated with each peak are saved in a SCAN table. The ranges and data pairs are combined whenever the ranges are within 6Ø microseconds of each other. The average range is saved in the SCAN table along with the count of pairs at that range. Pairs of data for channels and pulses for which a peak was not established are set to zero. The ranges for the two channels are averaged if they are within 6Ø microseconds, and this average is compaired to ranges already in the SCAN table. If these ranges match within 6Ø microseconds, a weighted average of the range is saved in the table.

13.6. PART Processing

The PART program was designed to output a set of data with no selection based upon peaks or thresholds, such as partial reflection data. The data is selected by a command to SOUNDER: FEP 3,pairs,address. The number of ranges is specified by pairs, such as 5Ø. The calibration pairs are followed by the 5Ø pairs requested. The address is an actual address in Input Memory, which will appear as the first range. This address must be an even number, since there are two words per X,Y pair. The range which appears in the tape record will have been converted to microseconds, and will have been adjusted by the TSG and analog delays. PART processing is the only type for which the PICKER task calculates the correct ranges when twenty microsecond sampling is used.

The data blocks are all six words long: the range, then a zero, then two X,Y pairs for channels 1 and 2. The range for the calibration is chosen near the center of the calibration interval. Words 14 and 15 are then non-zero, zero, which is recognised by the PICKER task as PART type processing.

13.7. EQUALREF Processing

The EQUALREF program does peak finding, so that the PICKER task has less work to do. In addition, the peak finding is done using the sum of the amplitudes of the two channels. This allows the recording of data at the same delay from the transmitted pulse for both channels. Three pairs from before the peak, and three pairs after the peak are also recorded. The data blocks are of fixed length: range, pointer, then the seven double pairs. The pointers are each 3Ø greater than the previous pointer. A double pair consists of the X,Y pair from channel 1 followed by the X,Y pair from channel 2. As always, the range is that of the first data pair. Processing continues until the calibration and 15 more data blocks have been saved, or until there are no more data. The combination of a non-zero range for the calibration, and a non-zero pointer is used by the PICKER task to signify EQUALREF type processing.

EQUALREF processing has become the most popular type. It finds more data sets, perhaps because fading in one channel does not stop data storage if the other channel has a strong signal.

At the same time, one channel barely above threshold will not cause recording unless the other channel is very near threshold. The eight pairs from four pulses can be used to solve for propagation properties, since they are at the same delay from the transmitted pulses.

The PICKER task combines the buffers in pairs, choosing the same range from each buffer. This is correct if both buffers are the same frequency, but not quite correct if one of the frequencies is offset. The range of the first pair of pulses is averaged with the range from the second pair, if the two range pairs are within 50 microseconds. The data are saved in the SCAN table for later packing.

13.8. PICKER Processing

The processing for each pulse and the data blocks and buffers generated are identical to those generated by EQUALREF. When EQUALREF finishes processing each of the first two pulses in a set, the buffer is copied to the upper half of the Output Memory. There are two Output Memories, one attached to the FEP program and one to the EKO device attached to the Interdata computer. These are alternated each time the FEP finishes a process to tranfer the results to the EKO buffer. The length of the first three pulse buffers is set to 14 words (short), and the actual length of the buffer is stored in word one, although the data are never transferred.

At the end of the fourth data pulse, data buffers are moved to the Scratch and Input Memories for the final coincidence checking. The FEP program is able to swap the Output Memories to get at those other two data buffers. The four buffers are scanned simultaneously for range coincidence. The smallest range is found. If no other range is within 55 microseconds of this range, the smallest range is dropped (KEEP = 4). The data for the range midway between this smallest and largest range is saved for all pulses with data in the range gate. Ranges for which data was saved are dropped, and a new smallest range is found. This processing continues until there are no more ranges, or until 25 range sets have been saved.

The data buffer sent to PICKER is identified by having both words 14 and 15 zero. The remainder of the buffer, starting with word 16 is exactly the form of a SCAN table, and can be packed accordingly. The maximum peaks for saved data with a KEEP of 6 or 8, are recorded in header words 10 and 11.

13.9. Pulse Averaging

All processing programs clear the data portions of the two Scratch Memories when they output a data buffer to SOUNDER. FEP start commands, other than the last start for a repeat command, start programs which add data to the existing Scratch Memories, but do not process that data or clear the memories. There are also start locations which return with a very short buffer but do no processing. These allow averaging under control of SOUNDER and may entail different TSG senarios.

Data (or a deconvolution of the data) are added to the Scratch Memory for each pulse. A Scratch Memory is used for each of the two Input Memories. After the data have been added to the Scratch Memories, register R8 is tested. R8 is set to zero for all but the last pulse within a repeat sequence. When R8 is zero, the program waits for the next pulse. If R8 is negative, a very short buffer (seven words) is sent, and the program waits for the next pulse. R8 is set to -1 for the last pulse of a sequence when pulses are to be combined before processing, such as complementary pairs. The repeat sequence is controlled from SOUNDER instead of from the TSG. The PICKER task completely ignores very short buffers.

Register R8 is set to one for the last pulse when processing is desired. The averaging program doubles R8 if Scratch Memory should be multiplied by 1.5, which is necessary for some weightings. Scratch Memory is divided by the power of two, greater or equal to the sum of weights, which is kept in register R4. The results are stored in Input Memory. The division program returns to the selected processing program. All processing programs clear Scratch Memory in preparation for more averaging.

13.10. Pulse Coding

A set of pulse coded routines are available. They are in two sets: Sn are for short (30 microsecond) pulses, but may work on systems with 60 microsecond filters; Ln are for longer (60 microsecond) pulses. All but S8 and S9 are Barker codes and side lobes are zero or one. 1 and 2, 4 and 6, and 8 and 9 are complementary codes. The sums of complementary codes have zero side lobes. They are used by specifying SUBL4 with no processing on the first pulse, and SUBL6 with the processing of your choice on the second pulse. The two pulses may have any repeat count, but it should be the same for both pulses. The transmitter pulse should be coded in the same manner as the chosen deconvolution.

13.11. FEP Start Addresses

The description in this section will refer to both the TSG ROM which has been in use for years (phase one) and a new ROM which has been tested (BWØ8), and is expected to come into general use. This new TSG ROM will be called phase two and is required for pulse coding and some repeat sequences.

	PULSE CODING	NONE	SNDR	PART	EQLR	PICK	PASS
NONE	PHASE ONE TSG	(TWO)	23	17	ØF	13	2B
NONE	PHASE ONE BASE	(TWO)	2Ø	14	ØC	1Ø	28
NONE	PHASE ONE TSG	(TOO)	15	18	12	ØF	1B
NONE	PHASE TWO TSG	(TOO)	14	17	11	ØE	1A
SUBØ	. . . .	ØB	13	16	1Ø	ØD	19
SUBL1	+ + ...	2A	32	35	2F	2C	
SUBL2	+ -	46	4E	51	4B	48	
SUBL3	+ + -	62	6A	6D	67	64	7Ø
SUBL4	+ + - + ...	84	8C	8F	89	86	92
SUBL6	+ + + -	B4	BC	BF	B9	B6	C2
SUBS1	+ + ...	1C	24	27	21	1E	
SUBS2	+ -	38	4Ø	43	3D	3A	
SUBS3	+ + -	54	5C	5F	59	56	
SUBS4	+ + - + ...	73	7B	7E	78	75	81
SUBS5	+ + + - + . . .	95	9D	AØ	9A	97	
SUBS6	+ + + -	A3	AB	AE	A8	A5	B1
SUBS7	+ + + - - + - .	C5	CD	DØ	CA	C7	D3
SUBS8	+ + + - + + - +	D6	DE	E1	DB	D8	E4
SUBS9	+ + + - - - + -	E7	EF	F2	EC	E9	F5

Start addresses for phase one are given as the last byte of the three bytes associated with the scenario number. This start address is used, as is, for repeat counts of 1 (in TSG zero page word 16). See Chapter 12 for descriptions of the TSG zero page. For repeat counts of more than 1, the base start address is given in byte 1F in the TSG zero page. The FEP is started at this base address for the first pulse of the repeat sequence. The FEP is started at this base address 'OR 2' for the last pulse of the repeat sequence. Note that the 2 is ORed and not added to the base address. The base address must be a multiple of four. All pulses except the first and last pulse start at the base address 'OR 1'. The type of processing is not specified by the start addresses associated with the scenario.

13.11. FEP Start Addresses

The first two lines of the start address table contain the addresses for phase one, single pulse (PHASE ONE TSG) and the base addresses for repeat sequences (PHASE ONE BASE). These addresses are for the file: FEPMEM.TWO, which runs with the old TSG ROM. The remaining addresses are for the file FEPMEM.TOO, which contains all the features described in this chapter. The line, PHASE TWO TSG, does not use the scratch memories and cannot have a repeat count. They behave exactly like phase one, processing the data directly from input memory to output memory. See Chapter 12 for examples on the use of these addresses.

13.12. Assembly Parameters

The FEP programs can be changed somewhat by changing a few parameters. The following symbols are in alphabetical order and appear in the first few lines of the program.

AVGC = XYZC+XYZC = 1000. AVGC is the number of words in the input buffer, twice the number of pairs.

CALX = SCNT = 24. CALX is the address of the peak of the calibration pulse. SCNT puts the peak in the center of the calibration interval.

EQRC = 15. EQRC is the maximum number of ranges to be saved for EQUALREF processing. EQRC cannot be larger than 15 without overflowing buffers.

FEPA = 30. FEPA is the number of words required for each range in EQUALREF. Seven double pairs plus a range and pointer is 30 words.

FEPB = 18. FEPB is the number of words in a SCAN table entry. Eight pairs pluse a range and count of pairs is 18.

FEPC = 12. Five times FEPC is the interval of range coincidence. Ranges closer than 55 microseconds are taken as coincident.

MEAN = SCNT+SCNT = 48. The quiet data starts at address MEAN. Deconvolution is done in two parts, above and below MEAN.

SCNT = 24. SCNT is the number of pairs in the calibration.

SHORT= 14. The short records output to make SOUNDER happy are SHORT words long.

SNDRC= 255. Each of the two halves of a SNDR process Output buffer are SNDRC words long.

XYZC = 500. XYZC is the number of data pairs in the input buffer. XYZC should not be made larger or the saved data in Scratch Memory might be overwritten.

XYZS = 9. XYZS is the shift count to divide the sum of inputs in calculating X-zero and Y-zero. XYZS is the base two logarithm of XYZC.

14. FEP VERIFICATION

James R. Winkelman

14.1. Introduction

A set of programs has been written to verify the proper operation of the Front End Processor (FEP) in the NOAA/SEL HF Radar. These programs all depend upon a certain level of correct operation. A program must run well enough to write interpretable results into the output memory and thence to the Interdata computer. This document assumes that the FEP is working well enough to report errors which are found by the verification programs. It also assumes that the FEP can be loaded from the Interdata via the DIO bus and that the EKO interface is working properly. The EKO interface can be tested with CALL task FEPEKO (Section 10.2.16). To check the FEP program load from the Interdata, load the HF Radar tasks and note the number of DIO errors on the startup display. If there are no DIO errors, the DIO bus can be assumed to be healthy, because it has been used extensively in writing programs to the TSG and FEP during SOUNDER initialization. Further checking can be done with alternate BOOT FEP and SYS commands. This is the only simple method to verify a good FEP program load.

Notice that the FEP can be tested without a functioning TSG, albeit laboriously, using FEPEKO and the TSG/FEP front panel. After BOOTing the FEP with the desired test program using FEPEKO, a manual FEP start can be given by depressing START on the panel. The starting addresses to the FEP verify routines are neither used nor checked. The data produced can then be transferred to the Interdata with a variety of FEPEKO commands. This may be necessary to indict or vindicate a suspected malfunctioning TSG.

FEP verification is divided into three overlapping domains:

1. Checking the validity of the microprogram memory which actually contains the programs.

2. Checking the data memories and their associated address logic.

3. Checking for proper operation of the CPU and sequencing operations.

14.1. Introduction

The descriptions that follow assume that only one FEP error is present, or that multiple errors do not interact. Multiple errors may cause more confusion than can be considered here. The user is advised to run the XFx block of verification programs first. (See Section 14.5.) If these run without error, the microprogram memory can be tested with the VFx and WFx programs (Section 14.3). Details about running the verification programs follow in Sections 14.2 and 14.2.1.

14.2. VERF

The main verify program is called VERF and is called by a CSS file of the same name. User interaction is through the computer console and the program communicates with the FEP through the FEP and EKO drivers (Section 2.11). Because the system TSG driver times out in three seconds and because some of the VERF tests run for more than twice this time, VERF has its own local TSG driver. (VFx, WFx, and YFx time out in 8Ø milliseconds, XFx in 8 seconds.)

If a hard copy of error reports is desired, a SET LOG device,C can be issued, or the hard copy device can be used. Volume SYSV should normally be mounted as the top, or removable, disc. The test programs can, however, be copied to and run from the bottom, or fixed disc.

After the operator has typed VERF, a brief reminder of the commands is written to the screen. Two types of commands are recognized: commands to change the mode of the verify program are preceeded by an @, tests to be run are identified by single letters or numbers. All commands are terminated with the carriage return, denoted by (cr). An alphanumeric string can run several tests without further operator intervention.

There are four mode changing commands to VERF. No blanks are allowed before the @ or between the @ and the letter. A letter other than A, B, L, or X following the @ terminates VERF. The normal termination is @(cr). (See the VERF example run, Section 14.2.1.)

@L n specifies the number of lines in an error report. Four lines is the default. Zero or blank both result in a single line. For all tests, the length of the error report is displayed in word Ø of the report. The test identifier, 1 for test A, 2 for B, etc., is in word 1. A test is 'GOOD' if the length of the buffer, which is the error report, is two, of if the length is three and word 2 is zero. If neither of these conditions is satisfied, the 'BAD' test results are displayed on the console.

@X n specifies the number of times each test is to be run. This is often used to run many passes of a memory test to find low probability errors or to run scope loops. Error reporting for the first pass through the test is normal. The remaining tests only print in case of errors, and then only a single line. Most of the tests run about 2Ø milliseconds, so @X 1ØØØØ may require 2ØØ seconds to run each test.

@A n is used to run the microprogram memory tests with a known bit error to determine the error reports generated if that bit changes in microprogram memory. The FEP memory location to be modified is specified by n (decimal). The valid microprogram line is displayed followed by pairs of lines for each error. The first line is the modified microinstruction; the second line is the error report which that bit change would generate.

@B n chooses the set of tests to be run, where n is Ø-3.

@B Ø tests microprogram memory with the test line in locations 2 through 926 (tests VFx).

@B 1 tests microprogram memory with the test line in locations 98 through 1Ø22 (tests WFx). See MVER (Section 14.3) for easy generation of user-written VFx and WFx tests.

@B 2 tests functions other than microprogram memory (tests XFx). See Section 14.5.

@B 3 tests programs in the YFx block, which often produce error reports even if correct. These tests may be in addition to the XFx tests, or may be simple programs designed to see what the FEP actually does in the tested circumstance. The YFx tests can be used for development, e.g., to see how a counter actually counts and tests for zero. Again, these tests may often produce 'BAD' results, even though the FEP is operating correctly. They are concentrated in @B3 so as not to confuse the casual user.

14.2. VERF

Running with @BØ, @B1, and @B2 should verify that the FEP is functioning properly. If many errors are detected in the VFx and WFx blocks (@BØ and @B1), the FEP should be debugged until the XFx block (@B2) runs correctly. Errors in the FEP logic or data memories are more likely than errors in the microprogram memory.

Each of the @Bn blocks allows for 36 programs. If this number is exceeded by user-written tests, the extra tests can be added to the YXx (@B3) block.

Typing (cr) only, runs all the tests A through Z and Ø through 9 for a given VERF mode, or block of tests. AXB3 runs tests xFA, xFX, xFB, and xF3 in that order. The actual tests run depend on the last @Bx command. @Xn specifies the number of times each test is run.

14.2.1. VERF Example

The following example of a VERF run shows how to run specific tests, change the VERF mode, and how to terminate VERF.

VERF>ABECCD(cr)

Run tests VFA, VFB, VFE, VFC twice (VFC is not reloaded before the second run), then run VFD.

VERF>@B 2(cr)
VERF>@X 9(cr)
VERF>AEC(cr)

Run XFA nine times, XFE nine times, and XFC nine times, reporting errors in detail for the first run and 'BAD TEST' for errors in subsequent runs.

VERF>@X 2(cr)
VERF>(cr)

Run tests XFA through XFZ and XFØ through XF9, twice each. Many of the XFx tests check data memory. Running them twice tests both banks of the input and output memories.

VERF>@(cr)

Terminate VERF.

I would recommend looking through the error reports for several references to the same address reporting an error. Isolate the bit in error and replace that memory chip. Look at error reports in XFx which usually indicate a failure in some function, such as an ALU. Errors found by XFx will usually have to be fixed before VFx and WFx will give meaningful results. It is often necessary to read the test program, either from a listing or by using the EDIT programs. Write-ups of individual programs are included as the first page of all verify programs.

14.3. MVER: Make VFx and WFx

The CSS call, MVER x, is used to make tests VFx and WFx. (The CSS call, FEPAM XFx, is used to assemble tests XFx and YFx. See Section 14.5.)

Tests XFx time out in eight seconds, about 4Ø million instructions, enough for exhaustive testing of bit combinations. The other tests time out in 8Ø milliseconds.

The microprogram line image on the disc is twelve bytes long. For historical and hardware reasons, the bytes in these records are in reverse order. The programs have always used copies of a standard load program for the complex loading proceedure. In running VERF, the user might notice the strange ordering of the bits for the first time. The ordering of these bits is not well documented elsewhere. The @A test modifies the bits, one at a time, from left to right in this twelve byte load vector. The error reports show the correct octal representation of the modified line, but a knowledge of the bit order can help find the changed bit.

Byte zero holds the lower eight bits of the microprogram memory address. Bytes eleven through one, in that order contain the 84 bits in order. Byte eleven has two zeros, the top two bits of the address, and then bits 83 to 8Ø. Bit 83 is the MSB, and bit Ø is the LSB. Byte one has bits 7 to Ø, byte two has bits 15 to 8, . . ., byte ten has bits 79 to 72. This means that @A modifies the bits in the order: 7-Ø, 15-8, 23-16, 31-24, 39-32, 47-4Ø, 55-48, 63-56, 71-64, 79-72, and then 83-8Ø.

The actual load procedure will not be described here, except to say that five words are sent over the DIO bus for every byte loaded into FEP memory. (See LDFEP, Section 2.9.2.1, for details about the FEP load procedure.) Individual lines, or microprogram instructions, can be loaded into the FEP, but not individual bytes.

The microprogram memory test programs contain more than 1ØØØ lines, 925 of which are identical within each program. The program called by the MVER CSS file makes the test program files FEPMEM.VFx and FEPMEM.WFx from an input file, Vx.VER. The call is MVER x. MVER x,S saves all intermediate files for checking or debugging MVER.

The Vx.VER file has a rigid format. Several comment lines with * in column one are followed by the test line which is to be replicated throughout memory. This test line is followed by the test program which can have any format, but must contain labels TEST and DONE, but must not contain an END directive.

The comments are listed followed by the test line. The test line is printed again at the top of the next page, followed by the test program proper (no END card).

Two programs are constructed: VFx.FEP and WFx.FEP. Both have a wait for the FEP start command in location zero and a GOTO TEST in location one. The test program is loaded into upper FEP program memory for VFx and into lower memory for WFx. The test program does the initialization and goes to START. The test line is replicated 925 times. The first line is at START and a GOTO DONE follows the last line.

The user must have a label DONE in the program which issues an error report and exits to transfer the report to the Interdata. Most test programs have a CHECK subroutine which checks the results of the execution of each test line.

LVER contains only the listing portion of MVER for additional listings. MVER runs from 15 to 35 minutes, depending upon the complexity of the test line.

The individual programs are not described in detail here. For details about any VFx and WFx, LIS Vx.VER,1,2Ø. This command, given to the OS, will display the comment lines and the beginning of the program. Most of these programs test many features and will respond to errors other than microprogram memory. These extra error tests will either confuse the user or corroborate errors reported by other tests. Many of the bits in the SH, COND, and SEQ fields must be deduced, as they interact to give ambiguous results in any one test. The user may have to change a suspicious chip and rerun the tests. All single bit errors can be identified, using the full set of tests, except bit 83, FEPDONE. No FEPDONE bit in the last line of the FEP program will cause an EKO timeout, as the FEP data will never be ready. A false FEPDONE will cause trouble only when it occurs early enough that the EKO transfer of the data begins before the data are ready, probably leaving word zero of the buffer at the value it was given at the start of the FEP program.

The error reports use 2, 3, 4, or 6 words per report so that an integral number of reports occur on each line. Only a few reports are written into the buffer. Later reports often reflect errors other than those intended, because the program didn't continue properly after the error. The user may have to read a program to properly interpret the results of a test.

The description of Vx will apply to both VFx and WFx. If the error is truely in the overlapped test region, 98 to 926 (62 to 39E hex) then there should be a similar error report from both VFx and WFx. If both tests start reporting errors at their lowest addresses, 2 and 98 (2 and 6Ø), the error is most likely to be another feature being tested, not microprogram memory. Errors between 2 and 98, or 927 and 1Ø23 may affect the test programs themselves and will cause some of the tests to fail completely.

Error report format:

```
LENGTH  ID  TWELVE  REPORT  WORDS
            TWELVE  MORE    REPORT  WORDS
            TWELVE  MORE    REPORT  WORDS
```

All error reports are hexadecimal and require knowledge of the specific test to interpret.

14.4. Expected Error Occurences, VFx and WFx

Tables 14.1 and 14.2 show expected error occurences for the VFx and WFx blocks of tests for bits ØØ-82 of the FEP microprogram memory. Tables 14.3 through 14.29 show actual error reports that may be displayed by the VFx and WFx tests, where x is A-Z, Ø-1. The two groups of tables are used together to attempt to resolve error ambiguities. For example, if the third line of Table 14.3 is seen as an error report, bit 56 or 58 has been erroneously turned on in test A, but the bit number is not uniquely identified by test A. Referring to Table 14.2, bit 56, we see that test C also reports an added bit 56, but this report is not ambiguous. (See the legend, below.) The test C report is shown in the first line of Table 14.5 and should be seen if bit 56 is indeed turned on. Notice that the bit 58 ambiguity may not be so easy to unravel.

In Tables 14.1 and 14.2, an attempt has been made to separate errors which immediately isolate a bit error (Ø and 1) from those tests which are more ambiguous (+ and -). The symbols + and 1 are used to show detection of a bit being added, changed from Ø to 1. Symbols - and Ø show a dropped bit, the change from 1 to Ø. These two tables summarize expected errors from each test. Other errors may be seen, but are not reported consistently. Tests including the 'X' (A, D, F, and H) can have an error in bits 69 to 78 masquerading as any other. Tests I, J, and R test these bits explicitly.

Some of the error reports are weak. The FEP will somtimes make extraneous reports or fail to report an error. This is especially true for errors in the branching control (bits 69-78).

The legend for Tables 14.1 and 14.2 follows:

+: A bit was changed from a zero to a one (added), but the bit number is not unique. Another bit being wrong could generate this same error report.

-: A one bit was changed to a zero (dropped), but another bit in error could have generated this same report.

1: A bit was added, or changed from zero to one. This error report is unique to this bit change.

Ø: A bit was dropped from a one to a zero. This report is unique to this bit change.

X: The bit change often causes an error report. Little more can be proven. The reports are variable. Believe those tests which produce a Ø or 1 from changes in this bit.

14.4. Expected Error Occurences, VFx and WFx

bit	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Ø	1	bit
ØØ															Ø	1		Ø									Ø	1	ØØ
Ø1	+		1	1		+	+	1				+	+	+	1	Ø						1	1	1	1	1	1	Ø	Ø1
Ø2	-		Ø	Ø		-	Ø	Ø				Ø	-	Ø	1	Ø						Ø	Ø	Ø	Ø	Ø	1	Ø	Ø2
Ø3	+		+	+		+	+	+				+	+	+	1	Ø		1				-	+	+	+	+	1	Ø	Ø3
Ø4	+		1	1		+	1	1				1		Ø	1	Ø		-				+	-	-	1	1	1	Ø	Ø4
Ø5	+		+	+		+	1	+				1	-	-	1	Ø		-				+	+	+	+	+	1	Ø	Ø5
Ø6							Ø					Ø			Ø	1		-									Ø	1	Ø6
Ø7	-		-	-		-	-	-				-	-	Ø	Ø	1						Ø	Ø	1	-	1	Ø	1	Ø7
Ø8	+		1	1		+	1	1				1		1	1	Ø						Ø	Ø		1		1	Ø	Ø8
Ø9	-		Ø	Ø		-	Ø	Ø				Ø	1	-								+	+	1	Ø	1			Ø9
1Ø	-		-	-		-	-	Ø				-	1	Ø	1							Ø	Ø	Ø	Ø	-	1		1Ø
11	-		Ø	Ø		-	1	Ø				1	1	Ø	Ø							1	1	1	Ø	1	Ø		11
12	+		+	+		+	1	+				1	1	Ø	1							1	1	1	+	1	1		12
13	Ø		-	-		-	-	-				-	Ø	1	Ø	-						Ø	Ø	Ø	-	Ø	Ø	-	13
14	-		-	-		-	Ø	-				Ø	Ø	1	1	+									-		1	+	14
15	-		-	-		-	1	-				+	Ø	1	1	+		Ø							-		1	+	15
16	+		+	+		+	+	+				+	Ø	1	Ø	-									+		Ø	-	16
17		-							Ø	Ø							1	-							-				17
18		-							Ø	Ø							1	-							-				18
19		Ø							Ø	Ø							1	-							Ø				19
2Ø		-							1	+							Ø	-							Ø				2Ø
21		Ø							Ø	1								-											21
22		Ø							Ø	1								-											22
23		1							Ø	1								+											23
24		1							Ø	1								1											24
25		1							Ø	1								1											25
26		1							Ø	1								1											26
27		1							Ø	1								1											27
28		1							Ø	1								1											28
29		1							Ø	1								1											29
3Ø		1							Ø	1								1											3Ø
31		Ø		+		+			Ø	-																			31
32		Ø		+		1																			+				32
33		Ø		+																									33
34		+																-							1				34
35	+		+		+									+	+										-		+	+	35
36									Ø	1																			36
37							+		Ø	1		1																	37
38									Ø	1																			38
39									Ø	1																			39
4Ø	+	1	Ø	1		1	1					Ø						1								Ø			4Ø
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Ø	1	

Table 14.1. Expected Error Occurences, Bits ØØ-4Ø

14.4. Expected Error Occurences, VFx and WFx

bit	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Ø	1	bit
41	1	Ø	Ø	-		-	1					Ø						1								Ø			41
42	1	1	1	Ø		-	1					Ø						Ø								1			42
43	1				1		-					Ø																	43
44	1		Ø			1	1					Ø												Ø	1				44
45	1		Ø			Ø	1					Ø												Ø	-				45
46	1		1			-	1					Ø												1	Ø				46
47	1				1		Ø					Ø			Ø												Ø		47
48		1		1		1	1					Ø						1				Ø			1				48
49		Ø		-		Ø	1					Ø						1				Ø			1				49
5Ø		1		Ø		-	1					Ø						Ø				1			1				5Ø
51					1		Ø					Ø																	51
52	1	1	Ø	1			1					Ø																	52
53	1	Ø	Ø	-			1																						53
54	1	1	1	Ø			1					Ø																	54
55	1						Ø					Ø																	55
56	+		1	Ø		-		Ø							-	-					-						+	+	56
57	1	1		-		-		-							-	-	1		1	1	Ø	1	1	1	1	+			57
58	+		-	-		-		-							-	-											-	-	58
59	-		-		+																-		-		-		-	-	59
6Ø				-		-		+					-	-							-		-		-				6Ø
61				+		+		-						-							-								61
62			+	+		+							-	-		-												-	62
63	-		-	+	+	+		+					+	+	+	+					-		-		+		-	-	63
64																			1	+	Ø								64
65																			1	1	Ø								65
66																			1	1	Ø								66
67																			Ø	+	1								67
68																			Ø	1	1								68
69	X			X		X		X	Ø	1									1										69
7Ø	X			X		X		X	Ø	1									1										7Ø
71	X			X		X		X	Ø	1									Ø										71
72	X			X		X		X	Ø	1									1										72
73	X			X		X		X	Ø	1									1										73
74	X			X		X		X	Ø	1									1										74
75	X			X		X		X	Ø	1									1										75
76	X			X		X		X	Ø	1									1										76
77	X			X		X		X	Ø	1									1										77
78	X			X		X		X	Ø	1									1										78
79																1						-	+	Ø		-		1	79
8Ø																						+	-			+			8Ø
81	-		-	-		-	-	-				-			1							+			-	-	1		81
82																						+	-						82
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Ø	1	

Table 14.2. Expected Error Occurences, Bits 41-82

ØØØ8	ØØØ1	ad+2	addr	ad+1	ØØ11	ØØ11	ØØ11	13-
ØØØ8	ØØØ1	ØØØØ	-addr	-ad-1	addr	ad+1	ØØØ1	57+
ØØØ8	ØØØ1	ØØØ1	-addr	ad+1	ØØ11	ØØ11	ØØ11	56+ 58+
ØØØ8	ØØØ1	ØØØ1	ØØØØ	ØØ11	addr	FFFF	FFFF	55+
ØØØ8	ØØØ1	ØØØ1	ØØØØ	ØØ11	FFFF	addr	FFFF	43+
ØØØ8	ØØØ1	ØØØ1	ØØØØ	ØØ12	FFFF	FFFF	addr	47+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØØ3	ØØ11	ØØ11	41+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ1Ø	ØØ11	ØØ11	42+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØØ3	ØØ11	53+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ1Ø	ØØ11	54+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ11	ØØØ3	45+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ11	ØØ1Ø	46+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ11	ØØ11	1+ 2- 3+ 4+ 5+ 7- 8+ 9- 1Ø- 11- 12+ 14- 15- 16+ 81-
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ11	ØØ13	44+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ11	ØØ13	ØØ11	52+
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ12	ØØ12	ØØ12	35+ 59- 63-
ØØØ8	ØØØ1	ØØØ1	addr	ad+1	ØØ13	ØØ11	ØØ11	4Ø+

Table 14.3. Expected Error Reports for VFA and WFA

0004	0002	-ad+1	ad-1					57+
0008	0002	0002	addr	0000	0000	0003	0003	40+
0008	0002	0002	addr	0000	0001	0001	0001	22-
0008	0002	0002	addr	0000	0002	0002	0002	21-
0008	0002	0002	addr	0000	0003	0000	0003	48+
0008	0002	0002	addr	0000	0003	0003	0000	52+
0008	0002	0002	addr	0000	0003	0003	0004	54+
0008	0002	0002	addr	0000	0003	0003	0007	33-
0008	0002	0002	addr	0000	0003	0003	0008	53-
0008	0002	0002	addr	0000	0003	0004	0003	50+
0008	0002	0002	addr	0000	0003	0007	0003	32-
0008	0002	0002	addr	0000	0003	0008	0003	49-
0008	0002	0002	addr	0000	0004	0003	0003	42+
0008	0002	0002	addr	0000	0007	0003	0003	31-
0008	0002	0002	addr	0000	0007	0007	0007	23+
0008	0002	0002	addr	0000	0008	0003	0003	41-
0008	0002	0002	addr	0000	000B	000B	000B	24+
0008	0002	0002	addr	0000	0013	0013	0013	25+
0008	0002	0002	addr	0000	0023	0023	0023	26+
0008	0002	0002	addr	0000	0043	0043	0043	27+
0008	0002	0002	addr	0000	0083	0083	0083	28+
0008	0002	0002	addr	0000	0103	0103	0103	29+
0008	0002	0002	addr	0000	0203	0203	0203	30+
0008	0002	0002	addr	0003	0003	0003	0003	20- 34+
0008	0002	0002	addr	0007	0003	0003	0003	19-
0008	0002	0002	addr	????	0003	0003	0003	17- 18-

Table 14.4. Expected Error Reports for VFB and WFB

0008	0003	-ad-2	-addr	addr	ad+2	addr	-addr	56+
0008	0003	-ad-1	-addr	addr	ad+1	addr	-addr	35+ 58- 59- 62+ 63-
0008	0003	addr	-addr	ad+1	0000	0000	0000	4+
0008	0003	addr	addr	ad+1	0000	0000	0000	3+ 7- 10- 13- 14- 15- 16+ 81-
0008	0003	addr	ad+1	ad+1	0000	0000	0003	44-
0008	0003	addr	ad+1	ad+1	0000	0000	000F	46+
0008	0003	addr	ad+1	ad+1	0000	0000	0013	45-
0008	0003	addr	ad+1	ad+1	0000	0003	0000	52-
0008	0003	addr	ad+1	ad+1	0000	000F	0000	54+
0008	0003	addr	ad+1	ad+1	0000	0013	0000	53-
0008	0003	addr	ad+1	ad+1	0003	0000	0000	40-
0008	0003	addr	ad+1	ad+1	000F	0000	0000	42+
0008	0003	addr	ad+1	ad+1	0013	0000	0000	41-
0008	0003	addr	ad+4	ad+1	0000	0000	0000	2-
0008	0003	addr	2ad+2	ad+1	0000	0000	0000	8+
0008	0003	addr	0000	ad+1	0000	0000	0000	5+ 12+
0008	0003	addr	0001	ad+1	0000	0000	0000	11-
0008	0003	addr	0004	ad+1	0000	0000	0000	1+
0008	0003	addr	6667	ad+1	0000	0000	0000	9-

Table 14.5. Expected Error Reports for VFC and WFC

14.4. Expected Error Occurences, VFx and WFx

0004	0004	-ad-1	ad+1					57- 58-
0008	0004	-addr	ad+1	ad+1	0005	0005	0005	60- 61+ 62+ 63+
000A	0004	-addr	ad+1	ad+1	0005	0005	0005	56-
0008	0004	addr	-addr	ad+1	0005	0005	0005	4+
0008	0004	addr	addr	ad+1	0005	0005	0005	3+ 7- 10- 13- 14- 15- 16+ 81-
0008	0004	addr	ad+1	ad+1	0003	0005	0005	42-
0008	0004	addr	ad+1	ad+1	0005	0003	0005	54-
0008	0004	addr	ad+1	ad+1	0005	0005	0003	50-
0008	0004	addr	ad+1	ad+1	0005	0005	0007	48+
0008	0004	addr	ad+1	ad+1	0005	0005	0008	32+ 49-
0008	0004	addr	ad+1	ad+1	0005	0007	0005	52+
0008	0004	addr	ad+1	ad+1	0005	0008	0005	33+ 53-
0008	0004	addr	ad+1	ad+1	0007	0005	0005	40+
0008	0004	addr	ad+1	ad+1	0008	0005	0005	31+ 41-
0008	0004	addr	ad+5	ad+1	0005	0005	0005	2-
0008	0004	addr	2ad+2	ad+1	0005	0005	0005	8+
0008	0004	addr	0000	ad+1	0005	0005	0005	5+ 12+
0008	0004	addr	0005	ad+1	0005	0005	0005	1+
0008	0004	addr	0006	ad+1	0005	0005	0005	11-
0008	0004	addr	6667	ad+1	0005	0005	0005	9-

Table 14.6. Expected Error Reports for VFD and WFD

0006	0005	-ad-1	FFFE	addr	ad+1	35+ 59+ 63+
0006	0005	ad-1	0000	0000	ad+1	47+
0006	0005	ad-2	ad+1	0000	0000	43+
0006	0005	ad-2	0000	ad+1	0000	51+

Table 14.7. Expected Error Reports for VFE and WFE

0007	0006	-addr	ad+1	0005	0005	0005	60- 61+ 62+ 63+
0009	0006	-addr	ad+1	0005	0005	0005	56-
0007	0006	addr	ad+1	0003	0005	0005	42-
0007	0006	addr	ad+1	0005	0003	0005	46-
0007	0006	addr	ad+1	0005	0005	0003	50-
0007	0006	addr	ad+1	0005	0005	0005	1+ 2- 3+ 4+
							5+ 7- 8+ 9-
							10- 11- 12+ 13-
							14- 15- 16+ 81-
0007	0006	addr	ad+1	0005	0005	0007	48+
0007	0006	addr	ad+1	0005	0005	0008	49-
0007	0006	addr	ad+1	0005	0007	0005	44+
0007	0006	addr	ad+1	0005	0008	0005	45-
0007	0006	addr	ad+1	0005	0008	0008	32+
0007	0006	addr	ad+1	0007	0005	0005	40+
0007	0006	addr	ad+1	0008	0005	0005	31+ 41-
0004	0006	ad+1	-ad-1				57- 58-

Table 14.8. Expected Error Reports for VFF and WFF

0032	addr	ad+1	addr	0000	ad+2	ad+1	14-
000E	0007	ad-1	addr	0000	addr	ad+1	54+
000E	0007	ad-1	addr	1000	addr	ad+1	46+
000E	0007	ad-1	addr	2000	addr	ad+1	50+
000E	0007	ad-1	addr	3000	addr	ad+1	42+
0032	0007	addr	-ad+1	0000	ad+1	-ad+2	4+
000E	0007	addr	ad-1	0000	addr	ad-1	6-
0032	0007	addr	ad-1	0000	ad+1	addr	3+ 81-
0032	0007	addr	ad+7	0000	ad+1	ad+8	2-
0005	0007	addr	0000	0000			55-
000E	0007	addr	0000	0000	ad+1	addr	52+
0032	0007	addr	0000	0000	ad+1	0001	5+
0005	0007	addr	0000	3000			37+ 43-
000E	0007	addr	0000	3000	ad+1	addr	40+
0032	0007	addr	0002	0000	ad+1	0003	15+
0032	0007	addr	0003	0000	ad+1	0004	11+
0032	0007	addr	0008	0000	ad+1	0009	1+ 10-
0032	0007	addr	0223	0000	ad+1	0224	9-
0005	0007	addr	1000	1000			47-
000E	0007	addr	1000	1000	ad+1	addr	44+
0005	0007	addr	1000	2000			51-
000E	0007	addr	1000	2000	ad+1	addr	48+
0032	0007	addr	FBBC	0000	ad+1	FBBD	12+
0032	0007	ad+1	addr	0000	ad+2	ad+1	7- 13- 16+
0032	0007	ad+1	2ad+1	0000	ad+2	2ad+2	8+
000E	0007	0002	addr	0000	0003	ad+1	53+
000E	0007	0002	addr	1000	0003	ad+1	45+
000E	0007	0002	addr	2000	0003	ad+1	49+
000E	0007	0002	addr	3000	0003	ad+1	41+

Table 14.9. Expected Error Reports for VFG and WFG

0005	0008	-ad-1	ad+1	ad+1				60+ 61- 63+
0008	0008	-ad-1	ad+1	ad+1	addr	-ad-1	addr	56-
0005	0008	addr	-ad-1	addr				57- 58-
0005	0008	addr	-addr	-addr				4+
0005	0008	addr	addr	addr				3+ 7- 13- 14-
								15- 16+ 81-
0005	0008	addr	ad+9	ad+9				2-
0005	0008	addr	2ad+2	2ad+2				8+
0005	0008	addr	0000	0000				5+ 12+
0005	0008	addr	0009	0009				1+
0005	0008	addr	3334	3334				11-
0005	0008	addr	5556	5556				10-
0005	0008	addr	6667	6667				9-

Table 14.10. Expected Error Reports for VFH and WFH

14.4. Expected Error Occurences, VFx and WFx

ØØØ5	ØØØ9	addr	ØØØØ	ØØ8Ø	39-
ØØØ5	ØØØ9	addr	ØØØØ	ØØCØ	38-
ØØØ5	ØØØ9	addr	ØØØØ	ØØEØ	37-
ØØØ5	ØØØ9	addr	ØØØØ	ØØFØ	36-
ØØØ5	ØØØ9	addr	ØØØØ	ØØF8	78-
ØØØ5	ØØØ9	addr	ØØØØ	ØØFC	77-
ØØØ5	ØØØ9	addr	ØØØØ	ØØFE	76-
ØØØ5	ØØØ9	addr	ØØØØ	ØØFF	75-
ØØØ5	ØØØ9	addr	ØØØA	ØØØØ	18-
ØØØ5	ØØØ9	addr	ØØ16	ØØØØ	19-
ØØØ5	ØØØ9	addr	ØØ39	ØØØØ	2Ø+
ØØØ5	ØØØ9	addr	Ø1BA	ØØØØ	17-
ØØØ5	ØØØ9	addr	8ØØØ	ØØFF	74-
ØØØ5	ØØØ9	addr	CØØØ	ØØFF	73-
ØØØ5	ØØØ9	addr	EØØØ	ØØFF	72-
ØØØ5	ØØØ9	addr	FØØØ	ØØFF	71-
ØØØ5	ØØØ9	addr	F8ØØ	ØØFF	7Ø-
ØØØ5	ØØØ9	addr	FCØØ	ØØFF	69-
ØØØ5	ØØØ9	addr	FC39	ØØFF	31-
ØØØ5	ØØØ9	addr	FEØØ	ØØFF	3Ø-
ØØØ5	ØØØ9	addr	FFØØ	ØØFF	29-
ØØØ5	ØØØ9	addr	FF8Ø	ØØFF	28-
ØØØ5	ØØØ9	addr	FFCØ	ØØFF	27-
ØØØ5	ØØØ9	addr	FFEØ	ØØFF	26-
ØØØ5	ØØØ9	addr	FFFØ	ØØFF	25-
ØØØ5	ØØØ9	addr	FFF8	ØØFF	24-
ØØØ5	ØØØ9	addr	FFFC	ØØFF	23-
ØØØ5	ØØØ9	addr	FFFE	ØØFF	22-
ØØØ5	ØØØ9	addr	FFFF	ØØFF	21-

Table 14.11. Expected Error Reports for VFI and WFI

ØØØ5	ØØØA	addr	ØØØØ	ØØØ1	75+
ØØØ5	ØØØA	addr	ØØØØ	ØØØ2	76+
ØØØ5	ØØØA	addr	ØØØØ	ØØØ4	77+
ØØØ5	ØØØA	addr	ØØØØ	ØØØ8	78+
ØØØ5	ØØØA	addr	ØØØØ	ØØ1Ø	36+
ØØØ5	ØØØA	addr	ØØØØ	ØØ2Ø	37+
ØØØ5	ØØØA	addr	ØØØØ	ØØ4Ø	38+
ØØØ5	ØØØA	addr	ØØØØ	ØØ8Ø	39+
ØØØ5	ØØØA	addr	ØØØ1	ØØØØ	21+
ØØØ5	ØØØA	addr	ØØØ2	ØØØØ	22+
ØØØ5	ØØØA	addr	ØØØ4	ØØØØ	23+
ØØØ5	ØØØA	addr	ØØØ8	ØØØØ	24+
ØØØ5	ØØØA	addr	ØØØA	ØØØØ	18-
ØØØ5	ØØØA	addr	ØØ1Ø	ØØØØ	25+
ØØØ5	ØØØA	addr	ØØ15	ØØØØ	19-
ØØØ5	ØØØA	addr	ØØ2Ø	ØØØØ	26+
ØØØ5	ØØØA	addr	ØØ38	ØØØØ	2Ø+ 31-
ØØØ5	ØØØA	addr	ØØ4Ø	ØØØØ	27+
ØØØ5	ØØØA	addr	ØØ8Ø	ØØØØ	28+
ØØØ5	ØØØA	addr	Ø1ØØ	ØØØØ	29+
ØØØ5	ØØØA	addr	Ø1B9	ØØØØ	17-
ØØØ5	ØØØA	addr	Ø2ØØ	ØØØØ	3Ø+
ØØØ5	ØØØA	addr	Ø4ØØ	ØØØØ	69+
ØØØ5	ØØØA	addr	Ø8ØØ	ØØØØ	7Ø+
ØØØ5	ØØØA	addr	1ØØØ	ØØØØ	71+
ØØØ5	ØØØA	addr	2ØØØ	ØØØØ	72+
ØØØ5	ØØØA	addr	4ØØØ	ØØØØ	73+
ØØØ5	ØØØA	addr	8ØØØ	ØØØØ	74+

Table 14.12. Expected Error Reports for VFJ and WFJ

ØØ32	addr	ad+2	ad+1	ØØØØ	ad+3	14-
ØØØE	ØØØC	ad-4	ad+1	1ØØØ	ad-3	45-
ØØØE	ØØØC	ad-4	ad+1	2ØØØ	ad-3	49-
ØØØE	ØØØC	ad-4	ad+1	3ØØØ	ad-3	41-
ØØ32	ØØØC	ad+1	-ad+2	ØØØØ	ad+2	4+
ØØØE	ØØØC	ad+1	addr	ØØØØ	ad+1	6-
ØØ32	ØØØC	ad+1	addr	ØØØØ	ad+2	3+ 81-
ØØ32	ØØØC	ad+1	ad+D	ØØØØ	ad+2	2-
ØØ32	ØØØC	ad+1	ØØØ1	ØØØØ	ad+2	5+
ØØØ5	ØØØC	ad+1	ØØØ2	ØØØØ		55-
ØØØE	ØØØC	ad+1	ØØØ2	ØØØØ	ad+2	52-
ØØØ5	ØØØC	ad+1	ØØØ2	1ØØØ		47-
ØØØE	ØØØC	ad+1	ØØØ2	1ØØØ	ad+2	44-
ØØØ5	ØØØC	ad+1	ØØØ2	2ØØØ		51-
ØØØE	ØØØC	ad+1	ØØØ2	2ØØØ	ad+2	48-
ØØØ5	ØØØC	ad+1	ØØØ2	3ØØØ		43-
ØØØE	ØØØC	ad+1	ØØØ2	3ØØØ	ad+2	4Ø-
ØØØ5	ØØØC	ad+1	ØØØ2	ad+3ØØØ		37+
ØØ32	ØØØC	ad+1	ØØØE	ØØØØ	ad+2	1+ 1Ø-
ØØ32	ØØØC	ad+1	Ø224	ØØØØ	ad+2	9-
ØØ32	ØØØC	ad+1	Ø779	ØØØØ	ad+2	11+
ØØ32	ØØØC	ad+1	FBBD	ØØØØ	ad+2	12+
ØØ32	ØØØC	ad+2	ad+1	ØØØØ	ad+3	7- 13- 15+ 16+
ØØ32	ØØØC	ad+2	2ad+2	ØØØØ	ad+3	8+
ØØØE	ØØØC	ØØØ3	ad+63	ØØØØ	ØØØ4	54-
ØØØE	ØØØC	ØØØ3	ad+63	1ØØØ	ØØØ4	46-
ØØØE	ØØØC	ØØØ3	ad+63	2ØØØ	ØØØ4	5Ø-
ØØØE	ØØØC	ØØØ3	ad+63	3ØØØ	ØØØ4	42-

Table 14.13. Expected Error Reports for VFL and WFL

ØØØ8	ØØØD	addr	addr	-addr	-addr	addr	-addr	6Ø- 62- 63+
ØØØ8	ØØØD	addr	ØØØ1	FFFF	FFFF	FFFF	FFFF	9+
ØØØ8	ØØØD	addr	ØØØ2	FFFF	FFFF	FFFF	FFFF	1Ø+
ØØØ8	ØØØD	addr	ØØØ4	FFFF	FFFF	FFFF	FFFF	11+
ØØØ8	ØØØD	addr	ØØØ8	FFFF	FFFF	FFFF	FFFF	12+
ØØØ8	ØØØD	addr	ØØØD	FFFF	FFFF	FFFF	FFFF	1+ 2-
ØØØ8	ØØØD	addr	FFFF	ØØØØ	FFFF	FFFF	FFFF	13-
ØØØ8	ØØØD	addr	FFFF	FFFF	ØØØØ	FFFF	FFFF	14-
ØØØ8	ØØØD	addr	FFFF	FFFF	FFFF	ØØØØ	FFFF	15-
ØØØ8	ØØØD	addr	FFFF	FFFF	FFFF	FFFF	ØØØØ	16-
ØØØ8	ØØØD	addr	FFFF	FFFF	FFFF	FFFF	FFFF	3+ 5- 7-

Table 14.14. Expected Error Reports for VFM and WFM

0008	000E	ad-1	ad-1	-ad+1	-ad+1	ad-1	-ad+1	60-	61-	62-	63+
0008	000E	addr	0000	FFFF	FFFF	FFFF	FFFF				4-
0008	000E	addr	0001	FFFF	FFFF	FFFF	FFFF				2-
0008	000E	addr	0007	FFFF	FFFF	FFFF	FFFF				12-
0008	000E	addr	000B	FFFF	FFFF	FFFF	FFFF				11-
0008	000E	addr	000D	FFFF	FFFF	FFFF	FFFF				10-
0008	000E	addr	000E	FFFF	FFFF	FFFF	FFFF			1+	9-
0008	000E	addr	001E	FFFF	FFFF	FFFF	FFFF				8+
0008	000E	addr	FFF0	FFFF	FFFF	FFFF	FFFF			3+	5-
0008	000E	addr	FFFF	000F	FFFF	FFFF	FFFF				13+
0008	000E	addr	FFFF	FFFF	000F	FFFF	FFFF				14+
0008	000E	addr	FFFF	FFFF	FFFF	000F	FFFF				15+
0008	000E	addr	FFFF	FFFF	FFFF	FFFF	000F				16+
0008	000E	addr	FFFF	FFFF	FFFF	FFFF	FFFF				7-

Table 14.15. Expected Error Reports for VFN and WFN

0008	000F	addr	-addr	addr	0000	-addr		35+	63+
0008	000F	addr	000C	0000	000C	0000			5+
0008	000F	addr	00EC	0000	00EC	0000			3+
0008	000F	addr	00FC	0000	00FC	0000			1+
0008	000F	addr	00FC	0000	010B	0000			7-
0008	000F	addr	00FC	0000	011A	0000			16-
0008	000F	addr	00FC	0000	8897	0088			13-
0008	000F	addr	00FC	0000	BBCA	00BB			14+
0008	000F	addr	00FC	0000	DDEC	00DD			15+
0008	000F	addr	010B	0000	000F	0000			6-
0008	000F	addr	010B	0000	00FC	0000			47-
0008	000F	addr	010C	0000	010C	0000			81+
0008	000F	addr	011A	0000	011A	0000	56-	57-	58-
0008	000F	addr	0207	0000	0207	0000			11-
0008	000F	addr	020E	0000	020E	0000			2+
0008	000F	addr	0216	0000	010B	0000			8+
0008	000F	addr	0F0F	0000	0F0F	0000			0-
0008	000F	addr	7873	0077	7873	0077			10+
0008	000F	addr	DED9	00DD	DED9	00DD			12+
0008	000F	addr	FF12	00FF	FF12	00FF			4+

Table 14.16. Expected Error Reports for VFO and WFO

14.4. Expected Error Occurences, VFx and WFx

0008	0010	addr	-addr	addr	0000	-addr	00FF	35+ 62- 63+
0008	0010	addr	000F	00FC	0000	0780	0000	13- 14+ 15+ 16-
0008	0010	addr	000F	00FC	0000	F100	00FF	8-
0008	0010	addr	000F	0700	0000	0780	0000	4-
0008	0010	addr	000F	077F	0000	0780	0000	3-
0008	0010	addr	000F	07FF	0000	0780	0000	5-
0008	0010	addr	000F	E200	00FF	1E00	0000	7+
0008	0010	addr	000F	F87F	007F	0780	0000	2-
0008	0010	addr	000F	F880	007F	0F00	0000	6+
0008	0010	addr	000F	F880	00FF	0780	0080	79+
0008	0010	addr	000F	FC3F	007F	03C0	0000	56- 57- 58-
0008	0010	addr	000F	FF00	007F	0780	0000	0+
0008	0010	addr	000F	FFF8	007F	0780	0000	1-

Table 14.17. Expected Error Reports for VFP and WFP

0004	0011	-ad+1	ad-1	57+
0004	0011	addr	ad-1	20-
0004	0011	addr	0003	17+
0004	0011	addr	0005	18+
0004	0011	addr	000F	19+

Table 14.18. Expected Error Reports for VFQ and WFQ

0008	0012	addr	0000	0007	0007	0001	0001	4-	6-	34-
0008	0012	addr	0001	0007	0007	0001	0001		19-	22-
0008	0012	addr	0002	0007	0007	0001	0001		5-	21-
0008	0012	addr	0003	0000	0007	0001	0001			41+
0008	0012	addr	0003	0007	0000	0001	0001			49+
0008	0012	addr	0003	0007	0007	0003	0001			15-
0008	0012	addr	0003	0007	0008	0001	0001			50-
0008	0012	addr	0003	0007	000E	0001	0001			48+
0008	0012	addr	0003	0008	0007	0001	0001			42-
0008	0012	addr	0003	000E	0007	0001	0001			40+
0008	0012	addr	0007	0007	0007	0001	0001		20-	23+
0008	0012	addr	000B	0007	0007	0001	0001			24+
0008	0012	addr	0011	0007	0007	0001	0001			0-
0008	0012	addr	0013	0007	0007	0001	0001			25+
0008	0012	addr	0023	0007	0007	0001	0001			26+
0008	0012	addr	0043	0007	0007	0001	0001			27+
0008	0012	addr	0083	0007	0007	0001	0001			28+
0008	0012	addr	0103	0007	0007	0001	0001			29+
0008	0012	addr	0203	0007	0007	0001	0001			30+
0008	0012	addr	03FC	0007	0007	0001	0001			3+
0008	0012	addr	????	0007	0007	0001	0001		17-	18-

Table 14.19. Expected Error Reports for VFR and WFR

0004	0013	-addr	addr			57+
0006	0013	addr	0006	0008	0005	64+
0006	0013	addr	0006	0008	0008	68-
0006	0013	addr	0007	0005	0008	67-
0006	0013	addr	0007	0007	0005	65+
0006	0013	addr	0007	0008	0001	71-
0006	0013	addr	0007	0008	0004	66+
0006	0013	addr	0007	0008	0006	69+
0006	0013	addr	0007	0008	0007	70+
0006	0013	addr	0007	0008	000D	72+
0006	0013	addr	0007	0008	0015	73+
0006	0013	addr	0007	0008	0025	74+
0006	0013	addr	0007	0008	0045	75+
0006	0013	addr	0007	0008	0085	76+
0006	0013	addr	0007	0008	0105	77+
0006	0013	addr	0007	0008	0205	78+

Table 14.20. Expected Error Reports for VFS and WFS

14.4. Expected Error Occurences, VFx and WFx

0004	0014	-addr	addr			57+
0006	0014	addr	0006	0008	0008	64+ 67+
0006	0014	addr	0007	0005	0008	68+
0006	0014	addr	0007	0007	0008	65+
0006	0014	addr	0007	0008	0007	66+

Table 14.21. Expected Error Reports for VFT and WFT

0004	0015	-addr	addr			57-
0006	0015	addr	0005	0007	0007	67+
0006	0015	addr	0006	0004	0007	68+
0006	0015	addr	0006	0007	0008	66-
0006	0015	addr	0006	0008	0007	65-
0006	0015	addr	0007	0007	0007	64-
0026	0015	ad+1	0005	0007		56- 59-
0006	0015	ad+1	0005	0007	0007	60- 61- 63-

Table 14.22. Expected Error Reports for VFU and WFU

0004	0016	-ad+1	ad-1			57+
0005	0016	addr	0000	0000		7-
0005	0016	addr	0001	0000		8-
0005	0016	addr	0002	0000		79- 80+
0005	0016	addr	0003	0002		48-
0005	0016	addr	0003	0015		50+
0005	0016	addr	0003	0018		49-
0005	0016	addr	0005	0000		3- 4+ 5+ 9+
						81+ 82+
0005	0016	addr	0027	0000		2-
0005	0016	addr	002B	0000		1+
0005	0016	addr	5553	0000		12+
0005	0016	addr	CCCB	0000		11+
0005	0016	addr	FFFD	0000		10-
001A	0016	ad+1	0005	0000	ad+2	13-

Table 14.23. Expected Error Reports for VFV and WFV

0004	0017	-ad+1	ad-1		57+
0004	0017	addr	0000		10-
0004	0017	addr	0004		3+ 5+
0004	0017	addr	0006		4- 9+
0004	0017	addr	3335		11+
0004	0017	addr	AAAC		12+
0004	0017	addr	FFD3		1+
0004	0017	addr	FFD7		2-
0004	0017	addr	FFFB		59- 60- 63- 82-
0004	0017	addr	FFFC		79+ 80-
0004	0017	addr	FFFE		8-
0004	0017	addr	FFFF		7-
001A	0017	ad+1	0006	ad+2	13-

Table 14.24. Expected Error Reports for VFW and WFW

0004	0018	-ad+1	ad-1			57+
0006	0018	addr	0000	0080	0000	10-
0006	0018	addr	0001	0080	0000	9+
0006	0018	addr	0003	0080	0000	3+ 4- 5+
0006	0018	addr	AAAA	00AA	0000	12+
0006	0018	addr	CCCC	00CC	0000	11+
0006	0018	addr	FFF1	00FF	0000	7+
0006	0018	addr	FFF3	00FF	0000	1+
0006	0018	addr	FFF7	00FF	0000	2-
0006	0018	addr	FFFC	007F	0000	79-
0006	0018	addr	FFFC	00FF	0007	44-
0006	0018	addr	FFFC	00FF	0015	46+
0006	0018	addr	FFFC	00FF	001D	45-
0022	0018	ad+1	0001	0080	0000	13-

Table 14.25. Expected Error Reports for VFX and WFX

14.4. Expected Error Occurences, VFx and WFx

ØØØ4	ØØ19	-ad-1	ad+1				57+
ØØØ8	ØØ19	-ad-1	addr	ØØØØ	ØØ1A	ØØ14	4+
ØØØ8	ØØ19	ad-1	addr	ØØØØ	ØØ1A	ØØ14	3+ 7- 13- 14- 15- 16+ 81-
ØØØ8	ØØ19	addr	addr	ad+1	ØØ1A	ØØ14	34+
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØØ5	ØØ14	49+
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØ19	ØØ14	5Ø+
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØ1A	ØØØ5	46-
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØ1A	ØØ18	44+
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØ1A	ØØ19	32+ 45-
ØØØ8	ØØ19	addr	addr	ØØØØ	ØØ1E	ØØ14	48+
ØØØ8	ØØ19	addr	addr	ØØØ1	ØØ1A	ØØ14	19-
ØØØ8	ØØ19	addr	addr	Ø3FF	ØØ1A	ØØ14	2Ø-
ØØØ8	ØØ19	addr	addr	????	ØØ1A	ØØ14	17- 18-
ØØØ8	ØØ19	ad+1	addr	ØØØØ	ØØ1B	ØØØF	35- 59- 6Ø- 63+
ØØØ8	ØØ19	ad+19	addr	ØØØØ	ØØ1A	ØØ14	2-
ØØØ8	ØØ19	2ad+1	addr	ØØØØ	ØØ1A	ØØ14	8+
ØØØ8	ØØ19	ØØØØ	addr	ØØØØ	ØØ1A	ØØ14	1Ø-
ØØØ8	ØØ19	ØØ14	addr	ØØØØ	ØØ1A	ØØ14	11-
ØØØ8	ØØ19	ØØ19	addr	ØØØØ	ØØ1A	ØØ14	1+
ØØØ8	ØØ19	6666	addr	ØØØØ	ØØ1A	ØØ14	9-
ØØØ8	ØØ19	FFFF	addr	ØØØØ	ØØ1A	ØØ14	5+ 12+

Table 14.26. Expected Error Reports for VFY and WFY

ØØØ4	ØØ1A	-ad+1	ad-1			57+
ØØØ6	ØØ1A	addr	ØØØØ	ØØ8Ø	ØØØØ	5+ 1Ø-
ØØØ6	ØØ1A	addr	ØØØ2	ØØCØ	ØØØØ	9+
ØØØ6	ØØ1A	addr	ØØØ3	ØØ8Ø	ØØØØ	3+ 81-
ØØØ6	ØØ1A	addr	ØØØ4	ØØØØ	ØØØØ	79- 8Ø+
ØØØ6	ØØ1A	addr	ØØØ4	ØØ8Ø	ØØØ7	4Ø-
ØØØ6	ØØ1A	addr	ØØØ4	ØØ8Ø	ØØ15	42+
ØØØ6	ØØ1A	addr	ØØØ4	ØØ8Ø	ØØ1D	41-
ØØØ6	ØØ1A	addr	ØØØD	ØØ8Ø	ØØØØ	1+
ØØØ6	ØØ1A	addr	ØØ11	ØØØØ	ØØØØ	7+
ØØØ6	ØØ1A	addr	ØØ11	ØØ8Ø	ØØØØ	2-
ØØØ6	ØØ1A	addr	3333	ØØB3	ØØØØ	11+
ØØØ6	ØØ1A	addr	5555	ØØD5	ØØØØ	12+
ØØØ6	ØØ1A	addr	FFFC	ØØFF	ØØØØ	4+
ØØ22	ØØ1A	ad+1	ØØØ2	ØØCØ	ØØØØ	13-

Table 14.27. Expected Error Reports for VFZ and WFZ

ØØØ8	ØØ5Ø	addr	-addr	addr	ØØØØ	-addr	35+ 56+ 58- 59- 63-
ØØØ8	ØØ5Ø	addr	ØØØC	ØØØØ	ØØØC	ØØØØ	5+
ØØØ8	ØØ5Ø	addr	ØØEC	ØØØØ	ØØEC	ØØØØ	3+
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	ØØFC	ØØØØ	1+
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	Ø1ØB	ØØØØ	7-
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	Ø11A	ØØØØ	16-
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	8897	ØØ88	13-
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	BBCA	ØØBB	14+
ØØØ8	ØØ5Ø	addr	ØØFC	ØØØØ	DDEC	ØØDD	15+
ØØØ8	ØØ5Ø	addr	Ø1ØB	ØØØØ	ØØØF	ØØØØ	6-
ØØØ8	ØØ5Ø	addr	Ø1ØB	ØØØØ	ØØFC	ØØØØ	47-
ØØØ8	ØØ5Ø	addr	Ø1ØC	ØØØØ	Ø1ØC	ØØØØ	81+
ØØØ8	ØØ5Ø	addr	Ø2Ø7	ØØØØ	Ø2Ø7	ØØØØ	11-
ØØØ8	ØØ5Ø	addr	Ø2ØE	ØØØØ	Ø2ØE	ØØØØ	2+
ØØØ8	ØØ5Ø	addr	Ø216	ØØØØ	Ø1ØB	ØØØØ	8+
ØØØ8	ØØ5Ø	addr	ØFØF	ØØØØ	ØFØF	ØØØØ	Ø-
ØØØ8	ØØ5Ø	addr	7873	ØØ77	7873	ØØ77	1Ø+
ØØØ8	ØØ5Ø	addr	DED9	ØØDD	DED9	ØØDD	12+
ØØØ8	ØØ5Ø	addr	FF12	ØØFF	FF12	ØØFF	4+

Table 14.28. Expected Error Reports for VFØ and WFØ

ØØØ8	ØØ51	addr	-addr	addr	ØØØØ	-addr	ØØFF	35+ 56+ 58- 59- 62- 63-
ØØØ8	ØØ51	addr	ØØØF	ØØFC	ØØØØ	Ø78Ø	ØØØØ	13- 14+ 15+ 16-
ØØØ8	ØØ51	addr	ØØØF	ØØFC	ØØØØ	F1ØØ	ØØFF	8-
ØØØ8	ØØ51	addr	ØØØF	Ø7ØØ	ØØØØ	Ø78Ø	ØØØØ	4-
ØØØ8	ØØ51	addr	ØØØF	Ø77F	ØØØØ	Ø78Ø	ØØØØ	3-
ØØØ8	ØØ51	addr	ØØØF	Ø7FF	ØØØØ	Ø78Ø	ØØØØ	5-
ØØØ8	ØØ51	addr	ØØØF	E2ØØ	ØØFF	1EØØ	ØØØØ	7+
ØØØ8	ØØ51	addr	ØØØF	F87F	ØØ7F	Ø78Ø	ØØØØ	2-
ØØØ8	ØØ51	addr	ØØØF	F88Ø	ØØ7F	ØFØØ	ØØØØ	6+
ØØØ8	ØØ51	addr	ØØØF	F88Ø	ØØFF	Ø78Ø	ØØ8Ø	79+
ØØØ8	ØØ51	addr	ØØØF	FFØØ	ØØ7F	Ø78Ø	ØØØØ	Ø+
ØØØ8	ØØ51	addr	ØØØF	FFF8	ØØ7F	Ø78Ø	ØØØØ	1-

Table 14.29. Expected Error Reports for VF1 and WF1

14.5. Description of the XFx Tests

The CSS call, FEPAM XFx, is used to assemble tests XFx and YFx. To allow the process to be under CSS control, even though running in the foreground, FEPAM uses a small program in the background (.BG) partition to cause the CSS file to wait until the FEP assembler finishes. This program waits until the task in partition 1 is done and then returns the same END OF TASK status.

The XFx tests time out in eight seconds, about 4Ø million instructions, enough for exaustive testing of bit combinations.

Tests XFA through XFK test the FEP data memories. These tests read and write every third word with either zeros or ones and will detect all common addressing errors within the memory chips. Tests XFA, XFB, and XFC test the memories one at a time. Six tests are run on each memory, every third word for zero or one. Only the first error in each test is identified. Tests XFD, XFE, and XFF are similar, except that the memories are written in pairs, such as IM1=SM2=Ø. Errors in these tests, but not the single memory tests, imply problems with noise or loading. Tests XFG, XFH, and XFI test the memory in triples, and should be even more sensitive. Test XFJ makes sure that the word on each side of the test word is different and was written after writing the test word. XFK is like XFJ, except that four memories are written at once.

Notice that the FEP programs used for sounding only write to the FEP memories in pairs, never in triples or quadruples. Therefore, failures in XFG-XFK do not necessarily preclude successful sounding. Few FEPs can run a series of XFK without a few error reports.

Tests XFL, XFM, XFN, and XFO test the addressing logic for SA1, SA2, IA, and OA respectively. All increments are tested, both positive and negative.

Tests XFP, XFQ, and XFR test the logical functions in the CPU. All combinations of ten bits are tested. The three programs test the top ten bits in the CPU, the middle ten bits, and the lower ten bits. These three tests overlap and test some of the 24 bits twice. The tests are done by using the logical instructions to calculate the EXCLUSIVE OR function in many ways and comparing the results.

Tests XFS, XFT, and XFU test the arithmetic instructions in the CPU in a way similar to that in which the logic is tested, except that the tests use positive and negative sets of nine bits. These CPU function tests run several seconds to test the million different combinations of bits.

Tests XFV and XFW test all combinations of conditional tests involving the lower and upper twelve bits respectively. Test XFX is a simple sum of the first N integers, using all sixteen registers and Q. The largest positive number, and the largest 24 bit sum are tested. A good result probably means that all registers are good, but a bad result cannot be translated into a specific problem.

Tests XFY and XFZ are memory and continuity tests which are probably done better by tests XFA through XFO. Test XF5 tests all possible twelve bit products for correct results. XF6 tests the shift functions; the test is not exhaustive, but should be adequate. Test XF9 is a minimum program which reports: GOOD TEST Ø. It is used for all tests which have not been written yet. XF9 permits blocks of tests to be run without regard to whether all of the tests exist.

More tests will be needed as experience with the FEP continues. The user is encouraged to write his/her own tests, and to share them with other users. Writing tests, especially in YFx, is a good way to learn the FEP.

I. APPENDIX I

David C. Walden

I.1. Introduction

Section I.2 of this Appendix describes one class of error codes that may result when a command is given to the SOUNDER task. These codes are from the ERR n messages that are logged through the COUNSEL task. The error code, n, is taken by COUNSEL from the status field of the CØ2 Q-block (Section 7.8.1). The condition n > Ø signals COUNSEL that an error was detected by the command execution routine, and the message, ERR n, is logged. All logged messages can be reviewed with CALL task LOG (Section 1Ø.2.21).

The Scheduler ERR codes are described in Chapter 8 under command SRUN xxx,n. The other HF Radar tasks have relatively straightforward command sets and report only ERR 1.

Errors of the form ERROR nnnn AT aaaa are discussed in Section 2.6.2 under ERROR and ERRCO. Some ERRORs are discussed in this Appendix, however. (See ERR 11 and ERR 13.)

With the descriptions of the ERR codes that follow, first-order remedies are suggested. Each description is followed by a list of the affected commands enclosed in parentheses.

I.2. SOUNDER Task ERR Codes

ERR Description/Remedies

1 General purpose typo trap: a required command argument was missing or illegal. Refer to Chapter 8 and specific sections of Chapter 2 for the exact form of the command arguments. Table 2.3 lists the specific sections by SOUNDER command. Some commands have additional, more specific ERR codes for illegal arguments.

(AGC, ANT, BDW, BOOT, CALA, CDN, DELF, DFAC, DIO, EFRQ, EPF, F, FEP, HPAT, HPE, IFAT (IF), KEEP, LPAT, MDEL, MGET, MSAV, NFIL, NO, NSET, PKHI, PKLO, PSEQ, RAMP, RFAT (RF), RNGE, RUN, RXANT, SFRQ, SN, SPH, STEP, TSGC, TXANT, UDIO)

I.2. SOUNDER Task ERR Codes

ERR	Description/Remedies
2	The specified frequency or delta F was out of range. The frequency limits for CALA, EFRQ, F, SFRQ, and TXANT are 1ØØ-3ØØØØ kHz; the limits for DELF, RAMP, and STEP are Ø-3275 kHz. (CALA, DELF, EFRQ, F, RAMP, STEP, SFRQ, TXANT)
3	Insufficient time was allowed for the countdown of a time-specific RUN. The time specified is in minutes (and seconds) past the start of the current hour. If it is currently 1Ø:45 and you wish to run a sounding at 11:Ø5, RUN 65. RUN 5 is taken as an attempt to run a sounding at 1Ø:Ø5. (RUN mm, RUN mm:ss)
4	An odd-numbered TSG scenario was specified for other than a W-class sounding. For all but W-class soundings, the odd-numbered scenarios (3, 5, and 7) are listen-only scenarios used at protected frequencies. Scenario 3 invokes the same FEP processing as does 2, 5 as 4, and 7 as 6. Scenario 1, reserved for the receiver calibration cycle, should never be used as part of a pulse set, although no specific test is made for scenario 1 in the TSGC command routine. (TSGC)
5	An attempt was made to run a T- or W-class sounding before any frequencies were specified. Use command F nF <freq>. P-mode is the only permanent mode with a pre-specified frequency (2666 kHz, the frequency at which the partial reflection array at Boot Lake, CO, is tuned). Also, frequencies specified to permanent modes endure only for the immediate sounding session, that is, until CANHF is typed. Frequencies specified to user-defined modes are retained until changed. (CONT, RUN)

ERR	Description/Remedies
6	The specified PIK <task> file, <task>.TSK, was not found on volume SYS1. If the file exists on SYSC, it can be copied to the bottom disc. A new PICKER task must be assembled with CAL and TETed. (See Sections 3.1 and 9.4.) (CONT, MGET, PIK, RUN)
7	A reference was made to an inactive frequency table entry (SCT.FTBL, Section 7.4). F(cr) displays the currently active entries; if none are active, F(cr) results in ERR 7. (An attempt to delete an inactive frequency table entry with F nFX does not result in an error.) (F(cr), F nF+, F nF-, RNGE nF)
8	No TXON was given, the transmitter filament warm-up period (about 9Ø seconds) is still in progress, or the timer in the transmitter controller has failed. The high voltage cannot be turned on as long as the HV HOLD indicator is lit. (HVON)
9	A RUN or CALL task file, <task>.TSK, was not found on volume SYSC, or GRAPHER.TSK was not found on SYS1 after a RUN or CALL task terminated. The latter should never happen; GRAPHER.TSK must be on SYS1 in order for HFRADAR to succeed. The RUN <task> error is immediate; the CALL error does not occur until RUN or CONT is given. (The CALL task name may be valid, but the wrong disc may be mounted.) (CONT, RUN, RUN <task>)
1Ø	A RUN <task> command was given while another RUN task was still active. This can only happen if a RUN/CALL task releases control of the console while SOUNDER is in the IMA loop. None of the tasks described in Section 1Ø.2 allow this to happen. User-written tasks should also respect this convention if they are to be RUN. The tasks described in Section 1Ø.3 release the console before they terminate, but if properly used, they occupy GRAPHER's memory partition only while SOUNDER is in the SPC loop. (RUN <task>)

I.2. SOUNDER Task ERR Codes

ERR Description/Remedies

11 A tape read error was sensed, or the tape mounted was not written in standard TENUS format (Section 1Ø.3.1). Read errors also display an ERROR message. ERROR 82xx is a read parity error (interpreted by the OS as a timeout error after 5 tries); 88xx means an immediate EOF was sensed when the TPR command was given, indicating a double EOF, or EOI; and AØxx means the tape was not ready.

(TPR)

12 A tape replay was requested with a RUN task active. A conflict over tape use could result. This cannot happen with any of the tasks described in Section 1Ø.2.

(TPR)

13 The sounding mode revision (MD.MREV) of the referenced sounding mode was not compatible with the currently running SOUNDER task (SCT.MREV). If detected during SOUNDER task initialization, this condition is reported as ERROR ØØ13. Get a different SMODES.xxx file with HFRADAR xxx or run a different revision of SOUNDER. All modes in all SMODES files provided by SEL are compatible with SOUNDER 2.Ø2.

(RUN, MGET)

14 The command given was inappropriate for the class of the currently active sounding mode. See Sections 2.5 and 1Ø.2.1.

(S-class: F; T-class: EFRQ, SFRQ, STEP; W-class: DELF, EFRQ, RAMP, SFRQ, STEP)

15 The sounding mode directory in file SMODES.xxx is full. Delete an unused mode (MDEL id) or make a new, empty directory with MAKEM (Section 1Ø.2.1). Each directory has room for 43 user-defined modes, in addition to a maximum of 1Ø permanent modes.

(MSAV)

ERR	Description/Remedies
16	There was an error in the pulse set sequence as specified. The palette indexes must be 1-8, and the repeat counts can not be greater than 256. The table that holds these values (SCT.PSEQ) is 32 bytes long. Parentheses cannot be nested. ERR 16, first seen after the PSEQ command, is again sensed by RUN or CONT, if the error is left uncorrected. (CONT, PSEQ, RUN)
17	An error was detected in the AGC command argument list. The number of a given class of arguments cannot exceed the number of receiver channels under Automatic Gain Control. This number is Ø for AGC Ø, 1 for AGC 1 or AGC 2, and 2 for AGC 3. For example, AGC 2,16,12 is illegal; AGC 3,16,12 is accepted. The Pn argument should not be given, because it is not fully implemented. Pn defaults to the number of pulses per pulse set, typically 4. (AGC)
18	The referenced DIO device has not been configured into the system by DIOGEN (Section 1Ø.2.4). Command ANT will fail with ERR 18 unless both receiver inputs are configured to be pulse-to-pulse devices. If one receiver input is configured into the system and the other is not, ANT will accept a specification for the former only. Either BOOT DIO[,xxx] or RUN DIOGEN to correct the problem. (ANT: receiver input or receiving antenna multiplexor; HVON; TXPH; TXON)
19	The referenced DIO device has not been defined to be changed pulse-to-pulse, whether or not configured into the system. Make correction as described under ERR 18. Remember that UDIO changes the palette extension in SOUNDER memory only. This allows for quick experimentation with different sequences of DIO commands, but a permanent change must be made with DIOGEN (Section 1Ø.2.4). (UDIO)

I.2. SOUNDER Task ERR Codes

ERR	Description/Remedies
2Ø	The specified FEP program file, FEPMEM.xxx, was not found on volume SYS1. It may be on SYSC. A new FEPMEM file is created with FEPM (Section 1Ø.4.3.) (BOOT FEP, CONT, MGET, RUN)
21	The specified TSG program file, TSGMEM.xxx, was not found on SYS1. Check SYSC. To create a new TSGMEM file, CPY TSGMEM.yyy,TSGMEM.xxx, and edit TSGMEM.xxx as required with DUTIL (Section 1Ø.4.1). (BOOT TSG, CONT, MGET, RUN)
22	The specified DIO tables file, DIO.xxx, was not found on SYS1. The file may exist on SYSC. A new DIO file is created with DIOGEN (Section 1Ø.2.4). (BOOT DIO, CONT, MGET, RUN)
23	The specified protected frequency map, xPROT.TWO, was not found on SYS1. The file may be on SYSC, or a new xPROT.TWO file may be created with PROTF (Section 1Ø.2.3). The map identifier, x, is specified in MD.MAP (Section 7.1). MD.MAP must be set by MAKEM (Section 1Ø.2.1), because there is no SOUNDER command for this purpose. (CONT, EFRQ, F, MGET, RUN, SFRQ, VUF)
24	A CONT was specified, but no SPH (Soundings Per Hour) schedule was interrupted with AB. There is nothing with which to continue. Remember that if a RUN command is given after an SPH schedule has been interrupted, only one sounding results. The SPH schedule can be resumed only with CONT. A new SPH schedule is started with the command sequence, AB, SPH n, RUN. (CONT)
25	You didn't believe that command VUF was not implemented. Try V-mode (Section 1Ø.2.1). (VUF)

II. APPENDIX II

David C. Walden

II.1. Introduction

As for I-mode in Product One, the default for all of the Sweep-class sounding modes in Product Two is to sweep upwards in frequency, using a frequency index step that is linear in the logarithm of frequency. Thus the frequency, as an index, is specified in 12 bits, and the logarithmic sweep is automatic. (See PCT.FRQ1, Section 7.2.)

If command STEP (Chapter 8) specifies a frequency index, the sweep-limiting frequencies specified by SFRQ and EFRQ are adjusted to the nearest log step equivalent, e.g., 15ØØ.Ø kHz is taken to be 1499.8 kHz. The logarithmic sweep is then conducted from SFRQ to EFRQ, stepping STEP units of frequency index between pulse sets for the base frequency selection (Section 2.5).

If command STEP specifies a linear frequency, SFRQ and EFRQ are taken as specified. The linear sweep goes from SFRQ to EFRQ, stepping STEP kHz between pulse sets for the base frequency selection. The nearest frequency index equivalent of the base frequency is put into PCT.FRQ1. This frequency index is the base of a log ramp, if so specified by RAMP. If a linear ramp is specified on a log base frequency, the base frequency index is converted to frequency and used as the base of the ramp. In any case, the actual frequency of any given pulse, including the delta F adjustment, is saved in the PCT in Hz/1ØØ as described in Section 7.2. This applies to sounding modes of all classes.

T- and W-class frequencies are always taken exactly as specified by the F nF <freq> command. Sometimes it is desirable to select a T- or W-class frequency that exactly corresponds to a frequency index. Section II.2 provides a table of frequencies between LOFR and HIFR (Section 7.7) by frequency index. This table was constructed using subroutine NTOFR (Section 2.6.3). Notice that subroutine FRTON is not always an exact inverse of NTOFR. The frequencies in Section II.2 are listed in kHz and may be given to SOUNDER as listed.

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
270	99.7	99.9	100.1	100.3	100.4	100.6	100.8	100.9	101.1	101.3
280	101.5	101.7	101.8	102.0	102.2	102.4	102.5	102.7	102.9	103.1
290	103.2	103.4	103.6	103.8	104.0	104.2	104.3	104.5	104.7	104.9
300	105.1	105.2	105.4	105.6	105.8	106.0	106.2	106.3	106.5	106.7
310	106.9	107.1	107.3	107.5	107.6	107.8	108.0	108.2	108.4	108.6
320	108.8	109.0	109.1	109.3	109.5	109.7	109.9	110.1	110.3	110.5
330	110.7	110.9	111.1	111.3	111.4	111.6	111.8	112.0	112.2	112.4
340	112.6	112.8	113.0	113.2	113.4	113.6	113.8	114.0	114.2	114.4
350	114.6	114.8	115.0	115.2	115.4	115.6	115.8	116.0	116.2	116.4
360	116.6	116.8	117.0	117.2	117.4	117.6	117.8	118.0	118.2	118.4
370	118.6	118.8	119.0	119.3	119.5	119.7	119.9	120.1	120.3	120.5
380	120.7	120.9	121.1	121.3	121.6	121.8	122.0	122.2	122.4	122.6
390	122.8	123.0	123.2	123.5	123.7	123.9	124.1	124.3	124.5	124.7
400	125.0	125.2	125.4	125.6	125.8	126.0	126.3	126.5	126.7	126.9
410	127.1	127.4	127.6	127.8	128.0	128.2	128.5	128.7	128.9	129.1
420	129.4	129.6	129.8	130.0	130.3	130.5	130.7	130.9	131.2	131.4
430	131.6	131.9	132.1	132.3	132.5	132.8	133.0	133.2	133.5	133.7
440	133.9	134.2	134.4	134.6	134.9	135.1	135.3	135.6	135.8	136.0
450	136.3	136.5	136.7	137.0	137.2	137.5	137.7	137.9	138.2	138.4
460	138.7	138.9	139.1	139.4	139.6	139.9	140.1	140.3	140.6	140.8
470	141.1	141.3	141.6	141.8	142.1	142.3	142.5	142.8	143.0	143.3
480	143.5	143.8	144.0	144.3	144.5	144.8	145.0	145.3	145.5	145.8
490	146.0	146.3	146.5	146.8	147.1	147.3	147.6	147.8	148.1	148.3
500	148.6	148.8	149.1	149.4	149.6	149.9	150.1	150.4	150.7	150.9
510	151.2	151.4	151.7	152.0	152.2	152.5	152.8	153.0	153.3	153.6
520	153.8	154.1	154.4	154.6	154.9	155.2	155.4	155.7	156.0	156.2
530	156.5	156.8	157.0	157.3	157.6	157.9	158.1	158.4	158.7	159.0
540	159.2	159.5	159.8	160.1	160.3	160.6	160.9	161.2	161.5	161.7
550	162.0	162.3	162.6	162.9	163.1	163.4	163.7	164.0	164.3	164.6
560	164.9	165.1	165.4	165.7	166.0	166.3	166.6	166.9	167.2	167.4
570	167.7	168.0	168.3	168.6	168.9	169.2	169.5	169.8	170.1	170.4
580	170.7	171.0	171.3	171.6	171.8	172.2	172.4	172.8	173.1	173.3
590	173.7	173.9	174.3	174.6	174.9	175.2	175.5	175.8	176.1	176.4
600	176.7	177.0	177.3	177.6	177.9	178.2	178.5	178.8	179.2	179.5
610	179.8	180.1	180.4	180.7	181.0	181.3	181.7	182.0	182.3	182.6
620	182.9	183.2	183.6	183.9	184.2	184.5	184.8	185.2	185.5	185.8
630	186.1	186.4	186.8	187.1	187.4	187.7	188.1	188.4	188.7	189.1
640	189.4	189.7	190.0	190.4	190.7	191.0	191.4	191.7	192.0	192.4
650	192.7	193.0	193.4	193.7	194.0	194.4	194.7	195.0	195.4	195.7
660	196.1	196.4	196.7	197.1	197.4	197.8	198.1	198.5	198.8	199.2
670	199.5	199.8	200.2	200.6	200.9	201.2	201.6	201.9	202.3	202.7
680	203.0	203.4	203.7	204.0	204.4	204.8	205.1	205.5	205.8	206.2
690	206.5	206.9	207.3	207.6	208.0	208.4	208.7	209.1	209.4	209.8
700	210.2	210.5	210.9	211.3	211.6	212.0	212.4	212.7	213.1	213.5
710	213.9	214.2	214.6	215.0	215.3	215.7	216.1	216.5	216.8	217.2
720	217.6	218.0	218.3	218.7	219.1	219.5	219.9	220.2	220.6	221.0
730	221.4	221.8	222.2	222.6	222.9	223.3	223.7	224.1	224.5	224.9
740	225.3	225.7	226.0	226.4	226.8	227.2	227.6	228.0	228.4	228.8
750	229.2	229.6	230.0	230.4	230.8	231.2	231.6	232.0	232.4	232.8
760	233.2	233.6	234.0	234.5	234.9	235.3	235.7	236.1	236.5	236.9
770	237.3	237.7	238.1	238.6	239.0	239.4	239.8	240.2	240.6	241.1
780	241.5	241.9	242.3	242.7	243.2	243.6	244.0	244.4	244.8	245.3
790	245.7	246.1	246.5	247.0	247.4	247.8	248.2	248.7	249.1	249.5
800	250.0	250.4	250.8	251.3	251.7	252.1	252.6	253.0	253.4	253.9
810	254.3	254.8	255.2	255.7	256.1	256.5	257.0	257.4	257.9	258.3
	0	1	2	3	4	5	6	7	8	9

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
820	258.8	259.2	259.7	260.1	260.6	261.0	261.5	261.9	262.4	262.9
830	263.3	263.8	264.2	264.7	265.1	265.6	266.1	266.5	267.0	267.4
840	267.9	268.4	268.8	269.3	269.8	270.2	270.7	271.2	271.7	272.1
850	272.6	273.1	273.5	274.0	274.5	275.0	275.4	275.9	276.4	276.9
860	277.4	277.8	278.3	278.8	279.3	279.8	280.3	280.7	281.2	281.7
870	282.2	282.7	283.2	283.7	284.2	284.7	285.1	285.6	286.1	286.6
880	287.1	287.6	288.1	288.6	289.1	289.6	290.1	290.6	291.1	291.6
890	292.1	292.6	293.1	293.7	294.2	294.7	295.2	295.7	296.2	296.7
900	297.2	297.7	298.3	298.8	299.3	299.8	300.3	300.9	301.4	301.9
910	302.4	302.9	303.5	304.0	304.5	305.0	305.6	306.1	306.6	307.2
920	307.7	308.2	308.8	309.3	309.8	310.4	310.9	311.4	312.0	312.5
930	313.1	313.6	314.1	314.7	315.2	315.8	316.3	316.9	317.4	318.0
940	318.5	319.1	319.6	320.2	320.7	321.3	321.9	322.4	323.0	323.5
950	324.1	324.7	325.2	325.8	326.3	326.9	327.5	328.1	328.6	329.2
960	329.8	330.3	330.9	331.5	332.0	332.6	333.2	333.8	334.4	334.9
970	335.5	336.1	336.7	337.3	337.9	338.4	339.0	339.6	340.2	340.8
980	341.4	342.0	342.6	343.2	343.7	344.4	344.9	345.6	346.2	346.7
990	347.4	347.9	348.6	349.2	349.8	350.4	351.0	351.6	352.2	352.8
1000	353.4	354.0	354.7	355.3	355.9	356.5	357.1	357.7	358.4	359.0
1010	359.6	360.2	360.9	361.5	362.1	362.7	363.4	364.0	364.6	365.3
1020	365.9	366.5	367.2	367.8	368.4	369.1	369.7	370.4	371.0	371.7
1030	372.3	372.9	373.6	374.2	374.9	375.5	376.2	376.8	377.5	378.2
1040	378.8	379.5	380.1	380.8	381.4	382.1	382.8	383.4	384.1	384.8
1050	385.4	386.1	386.8	387.5	388.1	388.8	389.5	390.1	390.8	391.5
1060	392.2	392.9	393.5	394.2	394.9	395.6	396.3	397.0	397.7	398.4
1070	399.1	399.7	400.5	401.2	401.8	402.5	403.3	403.9	404.6	405.4
1080	406.0	406.8	407.5	408.1	408.9	409.6	410.3	411.0	411.7	412.5
1090	413.1	413.9	414.6	415.3	416.0	416.8	417.5	418.2	418.9	419.7
1100	420.4	421.1	421.8	422.6	423.3	424.1	424.8	425.5	426.3	427.0
1110	427.8	428.5	429.2	430.0	430.7	431.5	432.2	433.0	433.7	434.5
1120	435.3	436.0	436.7	437.5	438.3	439.0	439.8	440.5	441.3	442.1
1130	442.9	443.6	444.4	445.2	445.9	446.7	447.5	448.3	449.1	449.8
1140	450.6	451.4	452.1	452.9	453.7	454.5	455.3	456.1	456.9	457.7
1150	458.5	459.3	460.1	460.9	461.7	462.5	463.3	464.1	464.9	465.7
1160	466.5	467.3	468.1	469.0	469.8	470.6	471.4	472.2	473.1	473.9
1170	474.7	475.5	476.3	477.2	478.0	478.8	479.7	480.5	481.3	482.2
1180	483.0	483.8	484.7	485.5	486.4	487.2	488.0	488.9	489.7	490.6
1190	491.4	492.3	493.1	494.0	494.8	495.7	496.5	497.4	498.2	499.1
1200	500.0	500.8	501.7	502.6	503.4	504.3	505.2	506.1	506.9	507.8
1210	508.7	509.6	510.5	511.4	512.3	513.1	514.0	514.9	515.8	516.7
1220	517.6	518.5	519.4	520.3	521.2	522.1	523.0	523.9	524.8	525.8
1230	526.7	527.6	528.5	529.4	530.3	531.3	532.2	533.1	534.0	534.9
1240	535.9	536.8	537.7	538.7	539.6	540.5	541.5	542.4	543.4	544.3
1250	545.2	546.2	547.1	548.1	549.0	550.0	550.9	551.9	552.9	553.8
1260	554.8	555.7	556.7	557.7	558.6	559.6	560.6	561.5	562.5	563.5
1270	564.5	565.4	566.4	567.4	568.4	569.4	570.3	571.3	572.3	573.3
1280	574.3	575.3	576.3	577.3	578.3	579.3	580.3	581.3	582.3	583.3
1290	584.3	585.3	586.3	587.4	588.4	589.4	590.4	591.5	592.5	593.5
1300	594.5	595.5	596.6	597.6	598.6	599.7	600.7	601.8	602.8	603.8
1310	604.9	605.9	607.0	608.1	609.1	610.1	611.2	612.2	613.3	614.4
1320	615.5	616.5	617.6	618.7	619.7	620.8	621.9	622.9	624.1	625.1
1330	626.2	627.3	628.3	629.5	630.5	631.6	632.7	633.8	634.9	636.0
1340	637.1	638.3	639.3	640.4	641.5	642.7	643.8	644.9	646.0	647.1
1350	648.3	649.4	650.5	651.6	652.7	653.9	655.0	656.2	657.3	658.5
1360	659.6	660.7	661.9	663.0	664.1	665.3	666.5	667.6	668.8	669.9
	0	1	2	3	4	5	6	7	8	9

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
1370	671.1	672.3	673.4	674.6	675.8	676.9	678.1	679.3	680.5	681.6
1380	682.8	684.0	685.2	686.4	687.5	688.8	689.9	691.2	692.4	693.5
1390	694.8	695.9	697.2	698.4	699.6	700.8	702.0	703.3	704.4	705.7
1400	706.9	708.1	709.4	710.6	711.8	713.0	714.3	715.5	716.8	718.0
1410	719.3	720.5	721.8	723.0	724.2	725.5	726.8	728.1	729.3	730.6
1420	731.8	733.1	734.4	735.7	736.9	738.2	739.5	740.8	742.1	743.4
1430	744.7	745.9	747.3	748.5	749.9	751.1	752.5	753.7	755.1	756.4
1440	757.7	759.0	760.3	761.6	762.9	764.3	765.6	766.9	768.3	769.6
1450	770.9	772.3	773.6	775.0	776.3	777.7	779.1	780.3	781.7	783.0
1460	784.4	785.8	787.1	788.5	789.9	791.2	792.7	794.1	795.4	796.8
1470	798.2	799.5	801.0	802.4	803.7	805.1	806.6	807.9	809.3	810.8
1480	812.1	813.6	815.0	816.3	817.8	819.2	820.7	822.0	823.5	825.0
1490	826.3	827.8	829.2	830.7	832.1	833.6	835.0	836.5	837.9	839.4
1500	840.8	842.3	843.7	845.2	846.7	848.2	849.6	851.1	852.6	854.1
1510	855.6	857.0	858.5	860.0	861.5	863.1	864.5	866.0	867.5	869.0
1520	870.6	872.0	873.5	875.1	876.6	878.1	879.7	881.1	882.7	884.2
1530	885.8	887.3	888.9	890.4	891.9	893.4	895.0	896.6	898.2	899.7
1540	901.3	902.9	904.3	905.9	907.5	909.1	910.7	912.3	913.9	915.5
1550	917.1	918.7	920.3	921.8	923.4	925.0	926.6	928.2	929.8	931.5
1560	933.1	934.7	936.3	938.0	939.6	941.2	942.9	944.5	946.2	947.8
1570	949.4	951.1	952.7	954.4	956.1	957.7	959.4	961.0	962.7	964.4
1580	966.0	967.7	969.4	971.1	972.8	974.4	976.1	977.8	979.5	981.2
1590	982.9	984.6	986.3	988.0	989.7	991.4	993.1	994.8	996.5	998.2
1600	1000.0	1001.7	1003.4	1005.2	1006.9	1008.7	1010.4	1012.2	1013.9	1015.7
1610	1017.5	1019.2	1021.0	1022.8	1024.6	1026.3	1028.1	1029.9	1031.7	1033.5
1620	1035.3	1037.1	1038.9	1040.7	1042.5	1044.3	1046.1	1047.9	1049.7	1051.6
1630	1053.4	1055.2	1057.1	1058.9	1060.7	1062.6	1064.4	1066.2	1068.1	1069.9
1640	1071.8	1073.7	1075.5	1077.4	1079.3	1081.1	1083.0	1084.9	1086.8	1088.6
1650	1090.5	1092.4	1094.3	1096.2	1098.1	1100.0	1101.9	1103.8	1105.8	1107.7
1660	1109.6	1111.5	1113.4	1115.4	1117.3	1119.2	1121.2	1123.1	1125.1	1127.0
1670	1129.0	1130.9	1132.9	1134.8	1136.8	1138.8	1140.7	1142.7	1144.7	1146.7
1680	1148.6	1150.6	1152.6	1154.6	1156.6	1158.6	1160.6	1162.7	1164.7	1166.7
1690	1168.7	1170.7	1172.7	1174.8	1176.8	1178.8	1180.9	1183.0	1185.0	1187.0
1700	1189.1	1191.1	1193.2	1195.2	1197.3	1199.5	1201.5	1203.6	1205.7	1207.7
1710	1209.8	1211.9	1214.1	1216.2	1218.2	1220.3	1222.4	1224.5	1226.7	1228.8
1720	1231.0	1233.1	1235.2	1237.4	1239.5	1241.6	1243.8	1245.9	1248.2	1250.3
1730	1252.4	1254.6	1256.7	1259.0	1261.1	1263.3	1265.5	1267.7	1269.9	1272.1
1740	1274.3	1276.6	1278.7	1280.9	1283.1	1285.4	1287.6	1289.8	1292.0	1294.3
1750	1296.6	1298.8	1301.1	1303.3	1305.5	1307.8	1310.1	1312.4	1314.6	1317.0
1760	1319.2	1321.4	1323.8	1326.1	1328.3	1330.7	1333.0	1335.3	1337.7	1339.9
1770	1342.2	1344.6	1346.9	1349.2	1351.6	1353.9	1356.2	1358.7	1361.0	1363.3
1780	1365.7	1368.0	1370.5	1372.8	1375.1	1377.6	1379.9	1382.4	1384.8	1387.1
1790	1389.6	1391.9	1394.5	1396.8	1399.2	1401.7	1404.0	1406.6	1408.9	1411.5
1800	1413.9	1416.2	1418.8	1421.2	1423.7	1426.1	1428.7	1431.1	1433.7	1436.1
1810	1438.7	1441.1	1443.7	1446.1	1448.5	1451.1	1453.6	1456.2	1458.6	1461.2
1820	1463.7	1466.3	1468.8	1471.4	1473.9	1476.5	1479.1	1481.6	1484.3	1486.8
1830	1489.4	1491.9	1494.6	1497.1	1499.8	1502.3	1505.0	1507.5	1510.2	1512.9
1840	1515.4	1518.1	1520.6	1523.3	1525.9	1528.6	1531.3	1533.9	1536.6	1539.2
1850	1541.9	1544.7	1547.3	1550.0	1552.6	1555.4	1558.2	1560.7	1563.5	1566.1
1860	1568.9	1571.7	1574.3	1577.1	1579.9	1582.5	1585.4	1588.2	1590.8	1593.6
1870	1596.5	1599.1	1602.0	1604.8	1607.5	1610.3	1613.2	1615.8	1618.7	1621.6
1880	1624.3	1627.2	1630.0	1632.7	1635.6	1638.5	1641.4	1644.1	1647.1	1650.0
1890	1652.7	1655.6	1658.5	1661.5	1664.2	1667.2	1670.1	1673.1	1675.8	1678.8
1900	1681.7	1684.7	1687.5	1690.5	1693.5	1696.4	1699.2	1702.2	1705.2	1708.2
1910	1711.3	1714.0	1717.1	1720.1	1723.1	1726.2	1729.0	1732.0	1735.1	1738.1
	0	1	2	3	4	5	6	7	8	9

APPENDIX II

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
1920	1741.2	1744.0	1747.1	1750.2	1753.3	1756.3	1759.4	1762.3	1765.4	1768.5
1930	1771.6	1774.7	1777.8	1780.9	1783.8	1786.9	1790.1	1793.2	1796.4	1799.5
1940	1802.7	1805.8	1808.7	1811.9	1815.1	1818.2	1821.4	1824.6	1827.8	1831.0
1950	1834.2	1837.4	1840.6	1843.6	1846.8	1850.0	1853.2	1856.5	1859.7	1863.0
1960	1866.2	1869.5	1872.7	1876.0	1879.2	1882.5	1885.8	1889.1	1892.4	1895.6
1970	1898.9	1902.2	1905.5	1908.8	1912.2	1915.5	1918.8	1922.1	1925.5	1928.8
1980	1932.1	1935.5	1938.8	1942.2	1945.6	1948.9	1952.3	1955.7	1959.0	1962.4
1990	1965.8	1969.2	1972.6	1976.0	1979.4	1982.8	1986.2	1989.7	1993.1	1996.5
2000	2000.0	2003.4	2006.9	2010.4	2013.9	2017.4	2020.9	2024.4	2027.9	2031.5
2010	2035.0	2038.5	2042.1	2045.6	2049.2	2052.7	2056.3	2059.9	2063.5	2067.0
2020	2070.6	2074.2	2077.8	2081.4	2085.0	2088.7	2092.3	2095.9	2099.5	2103.2
2030	2106.8	2110.5	2114.2	2117.8	2121.5	2125.2	2128.9	2132.5	2136.2	2139.9
2040	2143.6	2147.4	2151.1	2154.8	2158.6	2162.3	2166.1	2169.8	2173.6	2177.3
2050	2181.1	2184.9	2188.7	2192.5	2196.3	2200.1	2203.9	2207.7	2211.6	2215.4
2060	2219.2	2223.1	2226.9	2230.8	2234.7	2238.5	2242.4	2246.3	2250.2	2254.1
2070	2258.0	2261.9	2265.8	2269.7	2273.6	2277.6	2281.5	2285.5	2289.4	2293.4
2080	2297.3	2301.3	2305.3	2309.3	2313.2	2317.2	2321.2	2325.4	2329.4	2333.4
2090	2337.5	2341.5	2345.5	2349.6	2353.6	2357.7	2361.8	2366.0	2370.1	2374.1
2100	2378.2	2382.3	2386.4	2390.5	2394.7	2399.0	2403.1	2407.2	2411.4	2415.5
2110	2419.7	2423.8	2428.2	2432.4	2436.5	2440.7	2444.9	2449.1	2453.5	2457.7
2120	2462.0	2466.2	2470.4	2474.8	2479.1	2483.3	2487.6	2491.9	2496.4	2500.6
2130	2504.9	2509.2	2513.5	2518.0	2522.3	2526.7	2531.0	2535.5	2539.9	2544.2
2140	2548.6	2553.2	2557.5	2561.9	2566.3	2570.9	2575.3	2579.7	2584.1	2588.7
2150	2593.2	2597.6	2602.3	2606.7	2611.1	2615.6	2620.3	2624.8	2629.2	2634.0
2160	2638.5	2642.9	2647.7	2652.2	2656.7	2661.5	2666.0	2670.6	2675.4	2679.9
2170	2684.5	2689.3	2693.9	2698.4	2703.3	2707.9	2712.5	2717.4	2722.0	2726.6
2180	2731.5	2736.1	2741.0	2745.7	2750.3	2755.3	2759.9	2764.9	2769.6	2774.3
2190	2779.2	2783.9	2789.0	2793.7	2798.4	2803.4	2808.1	2813.2	2817.9	2823.0
2200	2827.8	2832.5	2837.6	2842.4	2847.5	2852.3	2857.4	2862.2	2867.4	2872.2
2210	2877.4	2882.2	2887.4	2892.2	2897.1	2902.3	2907.2	2912.4	2917.3	2922.5
2220	2927.4	2932.6	2937.6	2942.8	2947.8	2953.0	2958.3	2963.3	2968.6	2973.6
2230	2978.9	2983.9	2989.2	2994.2	2999.6	3004.6	3010.0	3015.0	3020.4	3025.8
2240	3032.8	3036.3	3041.3	3046.7	3051.8	3057.3	3062.7	3067.8	3073.3	3078.4
2250	3083.9	3089.4	3094.6	3100.1	3105.2	3110.8	3116.4	3121.5	3127.1	3132.3
2260	3137.9	3143.5	3148.7	3154.3	3159.9	3165.1	3170.8	3176.4	3181.7	3187.3
2270	3193.0	3198.3	3204.0	3209.7	3215.0	3220.7	3226.4	3231.7	3237.5	3243.2
2280	3248.6	3254.4	3260.1	3265.5	3271.3	3277.1	3282.9	3288.3	3294.2	3300.0
2290	3305.4	3311.3	3317.1	3323.0	3328.5	3334.4	3340.3	3346.2	3351.7	3357.6
2300	3363.5	3369.5	3375.0	3381.0	3387.0	3392.9	3398.5	3404.5	3410.5	3416.5
2310	3422.6	3428.1	3434.2	3440.2	3446.3	3452.4	3458.0	3464.1	3470.2	3476.3
2320	3482.5	3488.1	3494.2	3500.4	3506.6	3512.7	3518.9	3524.6	3530.8	3537.0
2330	3543.2	3549.4	3555.7	3561.9	3567.7	3573.9	3580.2	3586.5	3592.8	3599.1
2340	3605.4	3611.7	3617.5	3623.8	3630.2	3636.5	3642.9	3649.3	3655.7	3662.1
2350	3668.5	3674.9	3681.3	3687.2	3693.6	3700.1	3706.5	3713.0	3719.5	3726.0
2360	3732.5	3739.0	3745.5	3752.0	3758.5	3765.1	3771.6	3778.2	3784.8	3791.3
2370	3797.9	3804.5	3811.1	3817.7	3824.4	3831.0	3837.7	3844.3	3851.0	3857.6
2380	3864.3	3871.0	3877.7	3884.4	3891.2	3897.9	3904.6	3911.4	3918.1	3924.9
2390	3931.7	3938.5	3945.3	3952.1	3958.9	3965.7	3972.5	3979.4	3986.2	3993.1
2400	4000.0	4006.9	4013.9	4020.9	4027.9	4034.9	4041.9	4048.9	4055.9	4063.0
2410	4070.0	4077.1	4084.2	4091.3	4098.4	4105.5	4112.7	4119.8	4127.0	4134.1
2420	4141.3	4148.5	4155.7	4162.9	4170.1	4177.4	4184.6	4191.8	4199.1	4206.4
2430	4213.7	4221.0	4228.4	4235.7	4243.1	4250.4	4257.8	4265.1	4272.5	4279.9
2440	4287.3	4294.8	4302.2	4309.7	4317.2	4324.7	4332.2	4339.7	4347.2	4354.7
2450	4362.3	4369.9	4377.4	4385.0	4392.6	4400.2	4407.9	4415.5	4423.2	4430.8
2460	4438.5	4446.2	4453.9	4461.6	4469.4	4477.1	4484.8	4492.6	4500.4	4508.2
	0	1	2	3	4	5	6	7	8	9

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
2470	4516.0	4523.8	4531.6	4539.5	4547.3	4555.2	4563.1	4571.0	4578.9	4586.8
2480	4594.7	4602.6	4610.6	4618.6	4626.5	4634.5	4642.5	4650.8	4658.9	4666.9
2490	4675.0	4683.0	4691.1	4699.2	4707.3	4715.4	4723.6	4732.0	4740.2	4748.3
2500	4756.5	4764.7	4772.9	4781.1	4789.4	4798.0	4806.2	4814.5	4822.8	4831.1
2510	4839.4	4847.7	4856.4	4864.8	4873.1	4881.5	4889.9	4898.3	4907.1	4915.5
2520	4924.0	4932.4	4940.9	4949.7	4958.2	4966.7	4975.3	4983.8	4992.8	5001.3
2530	5009.9	5018.5	5027.1	5036.1	5044.7	5053.4	5062.0	5071.1	5079.8	5088.5
2540	5097.2	5106.4	5115.1	5123.8	5132.6	5141.8	5150.6	5159.4	5168.2	5177.5
2550	5186.4	5195.2	5204.6	5213.4	5222.3	5231.2	5240.6	5249.6	5258.5	5268.0
2560	5277.0	5285.9	5295.4	5304.5	5313.5	5323.0	5332.1	5341.2	5350.8	5359.9
2570	5369.0	5378.6	5387.8	5396.9	5406.6	5415.8	5425.0	5434.8	5444.0	5453.2
2580	5463.0	5472.3	5482.1	5491.4	5500.7	5510.6	5519.9	5529.8	5539.2	5548.6
2590	5558.5	5567.9	5578.0	5587.4	5596.8	5606.9	5616.3	5626.4	5635.9	5646.1
2600	5655.6	5665.1	5675.3	5684.9	5695.1	5704.7	5714.9	5724.5	5734.8	5744.4
2610	5754.8	5764.4	5774.8	5784.5	5794.2	5804.6	5814.4	5824.8	5834.6	5845.0
2620	5854.8	5865.3	5875.2	5885.7	5895.6	5906.1	5916.7	5926.6	5937.2	5947.2
2630	5957.8	5967.8	5978.5	5988.5	5999.2	6009.2	6020.0	6030.0	6040.8	6051.6
2640	6061.7	6072.6	6082.7	6093.5	6103.7	6114.6	6125.5	6135.7	6146.7	6156.9
2650	6167.9	6178.9	6189.2	6200.3	6210.5	6221.6	6232.8	6243.1	6254.2	6264.6
2660	6275.8	6287.0	6297.4	6308.6	6319.9	6330.3	6341.6	6352.9	6363.4	6374.7
2670	6386.1	6396.6	6408.0	6419.4	6430.0	6441.4	6452.9	6463.5	6475.0	6486.5
2680	6497.2	6508.8	6520.3	6531.0	6542.6	6554.3	6565.9	6576.7	6588.4	6600.0
2690	6610.9	6622.6	6634.3	6646.1	6657.0	6668.8	6680.6	6692.4	6703.4	6715.2
2700	6727.1	6739.1	6750.0	6762.0	6774.0	6785.9	6797.0	6809.0	6821.0	6833.1
2710	6845.2	6856.3	6868.4	6880.5	6892.7	6904.8	6916.0	6928.2	6940.5	6952.7
2720	6965.0	6976.2	6988.5	7000.8	7013.2	7025.5	7037.9	7049.2	7061.6	7074.0
2730	7086.5	7098.9	7111.4	7123.9	7135.4	7147.9	7160.5	7173.0	7185.6	7198.2
2740	7210.8	7223.5	7235.1	7247.7	7260.4	7273.1	7285.9	7298.6	7311.4	7324.2
2750	7337.0	7349.8	7362.7	7374.4	7387.3	7400.2	7413.1	7426.0	7439.0	7452.0
2760	7465.0	7478.0	7491.0	7504.0	7517.1	7530.2	7543.3	7556.4	7569.6	7582.7
2770	7595.9	7609.1	7622.3	7635.5	7648.8	7662.1	7675.4	7688.7	7702.0	7715.3
2780	7728.7	7742.1	7755.5	7768.9	7782.4	7795.8	7809.3	7822.8	7836.3	7849.8
2790	7863.4	7877.0	7890.6	7904.2	7917.8	7931.4	7945.1	7958.8	7972.5	7986.2
2800	8000.0	8013.9	8027.8	8041.8	8055.8	8069.8	8083.8	8097.8	8111.9	8126.0
2810	8140.1	8154.3	8168.4	8182.6	8196.8	8211.1	8225.4	8239.7	8254.0	8268.2
2820	8282.6	8297.0	8311.4	8325.8	8340.3	8354.8	8369.3	8383.7	8398.3	8412.8
2830	8427.5	8442.1	8456.8	8471.4	8486.2	8500.9	8515.7	8530.2	8545.0	8559.8
2840	8574.7	8589.6	8604.5	8619.4	8634.4	8649.4	8664.4	8679.4	8694.5	8709.5
2850	8724.6	8739.8	8754.9	8770.1	8785.3	8800.5	8815.8	8831.1	8846.4	8861.7
2860	8877.1	8892.5	8907.9	8923.3	8938.8	8954.2	8969.7	8985.3	9000.8	9016.4
2870	9032.0	9047.6	9063.3	9079.0	9094.7	9110.4	9126.2	9142.0	9157.8	9173.6
2880	9189.5	9205.3	9221.2	9237.2	9253.1	9269.1	9285.1	9301.7	9317.8	9333.8
2890	9350.0	9366.1	9382.3	9398.4	9414.7	9430.9	9447.2	9464.0	9480.4	9496.7
2900	9513.1	9529.5	9545.9	9562.3	9578.8	9596.0	9612.5	9629.0	9645.6	9662.2
2910	9678.8	9695.5	9712.9	9729.6	9746.3	9763.0	9779.8	9796.6	9814.2	9831.1
2920	9848.0	9864.9	9881.8	9899.5	9916.5	9933.5	9950.6	9967.7	9985.6	10002.7
2930	10019.8	10037.0	10054.2	10072.3	10089.5	10106.8	10124.1	10142.3	10159.7	10177.0
2940	10194.5	10212.8	10230.2	10247.7	10265.3	10283.7	10301.3	10318.9	10336.5	10355.1
2950	10372.8	10390.5	10409.2	10426.9	10444.7	10462.5	10481.3	10499.2	10517.1	10536.0
2960	10554.0	10571.9	10590.9	10609.0	10627.0	10646.1	10664.3	10682.4	10701.6	10719.8
2970	10738.0	10757.3	10775.6	10793.9	10813.3	10831.7	10850.1	10869.6	10888.0	10906.5
2980	10926.1	10944.6	10964.3	10982.9	11001.5	11021.2	11039.9	11059.7	11078.5	11097.2
2990	11117.1	11135.9	11156.0	11174.8	11193.7	11213.8	11232.7	11252.9	11271.9	11292.2
3000	11311.2	11330.3	11350.6	11369.8	11390.2	11409.4	11429.9	11449.1	11469.7	11488.9
3010	11509.6	11528.9	11549.6	11569.0	11588.5	11609.3	11628.8	11649.6	11669.2	11690.1

0 1 2 3 4 5 6 7 8 9

II.2. Frequency by Frequency Index

	0	1	2	3	4	5	6	7	8	9
3020	11709.7	11730.7	11750.4	11771.4	11791.2	11812.3	11833.5	11853.3	11874.5	11894.4
3030	11915.7	11935.6	11957.0	11977.0	11998.4	12018.5	12040.0	12060.1	12081.7	12103.3
3040	12123.5	12145.2	12165.4	12187.1	12207.4	12229.3	12251.1	12271.5	12293.5	12313.9
3050	12335.9	12357.9	12378.5	12400.6	12421.1	12443.3	12465.6	12486.2	12508.5	12529.2
3060	12551.6	12574.0	12594.8	12617.2	12639.8	12660.6	12683.2	12705.8	12726.8	12749.5
3070	12772.2	12793.3	12816.1	12838.9	12860.0	12882.9	12905.9	12927.1	12950.1	12973.1
3080	12994.5	13017.6	13040.7	13062.1	13085.3	13108.6	13131.9	13153.4	13176.8	13200.1
3090	13221.8	13245.2	13268.7	13292.3	13314.0	13337.6	13361.3	13384.9	13406.8	13430.5
3100	13454.3	13478.2	13500.1	13524.0	13548.0	13571.9	13594.0	13618.1	13642.1	13666.2
3110	13690.4	13712.6	13736.9	13761.1	13785.4	13809.7	13832.1	13856.5	13881.0	13905.4
3120	13930.0	13952.5	13977.1	14001.7	14026.4	14051.0	14075.8	14098.5	14123.3	14148.1
3130	14173.0	14197.9	14222.9	14247.9	14270.8	14295.9	14321.0	14346.1	14371.3	14396.5
3140	14421.7	14447.0	14470.2	14495.5	14520.9	14546.3	14571.8	14597.3	14622.8	14648.4
3150	14674.0	14699.7	14725.4	14748.9	14774.6	14800.4	14826.3	14852.1	14878.0	14904.0
3160	14930.0	14956.0	14982.0	15008.1	15034.3	15060.4	15086.6	15112.9	15139.2	15165.5
3170	15191.8	15218.2	15244.7	15271.1	15297.6	15324.2	15350.8	15377.4	15404.0	15430.7
3180	15457.5	15484.2	15511.0	15537.9	15564.8	15591.7	15618.6	15645.6	15672.7	15699.7
3190	15726.8	15754.0	15781.2	15808.4	15835.6	15862.9	15890.3	15917.6	15945.0	15972.5
3200	16000.0	16027.8	16055.7	16083.6	16111.6	16139.6	16167.6	16195.7	16223.9	16252.1
3210	16280.3	16308.6	16336.9	16365.3	16393.7	16422.2	16450.8	16479.4	16508.0	16536.5
3220	16565.2	16594.0	16622.8	16651.7	16680.7	16709.6	16738.7	16767.4	16796.6	16825.7
3230	16855.0	16884.2	16913.6	16942.9	16972.4	17001.8	17031.4	17060.5	17090.1	17119.7
3240	17149.5	17179.2	17209.0	17238.9	17268.8	17298.8	17328.8	17358.8	17389.0	17419.1
3250	17449.3	17479.6	17509.9	17540.3	17570.7	17601.1	17631.7	17662.2	17692.8	17723.5
3260	17754.2	17785.0	17815.8	17846.6	17877.6	17908.5	17939.5	17970.6	18001.7	18032.9
3270	18064.1	18095.3	18126.7	18158.0	18189.4	18220.9	18252.4	18284.0	18315.6	18347.2
3280	18379.0	18410.7	18442.5	18474.4	18506.3	18538.3	18570.3	18603.4	18635.6	18667.7
3290	18700.0	18732.2	18764.6	18796.9	18829.4	18861.8	18894.4	18928.1	18960.8	18993.5
3300	19026.2	19059.0	19091.8	19124.7	19157.7	19192.0	19225.0	19258.1	19291.3	19324.5
3310	19357.7	19391.0	19425.8	19459.2	19492.6	19526.1	19559.7	19593.3	19628.5	19662.2
3320	19696.0	19729.8	19763.7	19799.1	19833.1	19867.1	19901.2	19935.4	19971.2	20005.4
3330	20039.7	20074.1	20108.5	20144.6	20179.1	20213.6	20248.3	20284.6	20319.4	20354.1
3340	20389.0	20425.6	20460.5	20495.5	20530.6	20567.5	20602.6	20637.8	20673.1	20710.2
3350	20745.6	20781.0	20818.4	20853.9	20889.5	20925.1	20962.7	20998.4	21034.2	21072.1
3360	21108.0	21143.9	21181.9	21218.0	21254.1	21292.3	21328.6	21364.8	21403.3	21439.6
3370	21476.1	21514.7	21551.3	21587.9	21626.7	21663.4	21700.2	21739.2	21776.1	21813.0
3380	21852.2	21889.2	21928.6	21965.8	22003.0	22042.5	22079.8	22119.5	22157.0	22194.4
3390	22234.3	22271.9	22312.0	22349.7	22387.4	22427.6	22465.5	22505.9	22543.9	22584.4
3400	22622.5	22660.6	22701.3	22739.6	22780.4	22818.8	22859.8	22898.2	22939.4	22977.9
3410	23019.2	23057.9	23099.3	23138.1	23177.0	23218.6	23257.6	23299.3	23338.4	23380.3
3420	23419.5	23461.5	23500.8	23542.9	23582.4	23624.6	23667.0	23706.6	23749.1	23788.8
3430	23831.5	23871.3	23914.1	23954.0	23996.9	24037.0	24080.0	24120.2	24163.4	24206.6
3440	24247.0	24290.4	24330.8	24374.3	24414.9	24458.6	24502.3	24543.1	24587.0	24627.8
3450	24671.8	24715.9	24757.0	24801.2	24842.3	24886.7	24931.2	24972.5	25017.1	25058.5
3460	25103.2	25148.0	25189.6	25234.5	25279.6	25321.3	25366.5	25411.7	25453.7	25499.0
3470	25544.5	25586.6	25632.2	25677.8	25720.1	25765.9	25811.8	25854.2	25900.2	25946.3
3480	25989.0	26035.2	26081.5	26124.3	26170.7	26217.2	26263.8	26306.8	26353.6	26400.3
3490	26443.6	26490.5	26537.5	26584.6	26628.1	26675.3	26722.6	26769.9	26813.6	26861.1
3500	26908.7	26956.4	27000.3	27048.1	27096.0	27143.9	27188.1	27236.2	27284.3	27332.5
3510	27380.8	27425.3	27473.8	27522.3	27570.9	27619.5	27664.3	27713.1	27762.0	27810.9
3520	27860.0	27905.0	27954.2	28003.4	28052.8	28102.1	28151.6	28197.1	28246.7	28296.3
3530	28346.1	28395.9	28445.8	28495.8	28541.7	28591.8	28642.0	28692.2	28742.6	28793.0
3540	28843.5	28894.0	28940.4	28991.1	29041.9	29092.7	29143.6	29194.6	29245.7	29296.9
3550	29348.1	29399.4	29450.8	29497.8	29549.3	29600.9	29652.6	29704.3	29756.1	29808.0
3560	29860.0	29912.0	29964.1	30016.3	30068.6	30120.9	30173.3	30225.8	30278.4	30331.0
	0	1	2	3	4	5	6	7	8	9

INDEX

INDEX