RSL Radar Systems and
Remote Sensing Laboratory

11

PART A
A DATA ACQUISITION/CONTROLLER SYSTEM

PART B
SUPPLEMENT TO
A DATA ACQUISITION/CONTROLLER SYSTEM

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
2291 Irving Hill Road
Lawrence, Kansas 66045-2969

PART A
A DATA ACQUISITION/CONTROLLER SYSTEM

PART B
SUPPLEMENT TO
A DATA ACQUISITION/CONTROLLER SYSTEM

Part A: Fitratullah Khan
Part B: Marcelo Cavalcanti

Radar Systems and Remote Sensing Laboratory
Department of Electrical Engineering and Computer Science, University of Kansas
2291 Irving Hill Road, Lawrence, Kansas 66045-2969
TEL: 913/864-4835 * FAX: 913/864-7789 * OMNET: KANSAS.U.RSL

RSL Technical Report 8621-1

Part A: January 1992; Part B: May 1994

Sponsored by:

Office of Naval Research
Arlington VA 22217-5000

Grant N00014-89-J-3221

PART A

A DATA ACQUISITION/CONTROLLER SYSTEM

MARCELO CAVALCANTI

A DATA ACQUISITION/CONTROLLER SYSTEM

Radar Systems and Remote Sensing Laboratory

The University of Kansas Center for Research Incorporated
2291 Irving Hill Road
Lawrence, Kansas 66045-2969

January 1992

ACKNOWLEDGMENTS

The design and development of the data acquisition/controller system was a group effort supervised by Dale Rummer. Other members of the group were Fitratullah Khan, Charles Roberts, Asif Salam, and Bita Sistani.

ABSTRACT

A data acquisition/controller system was required to be designed for gathering data simultaneously from different radars located spatially apart. Problems encountered in the earlier generation of data acquisition systems were to be alleviated. One of the major problems was the corruption of weak signals by noise on long analog signal cables. Some evidence of video display being another source of noise was also found. Another goal was to reduce the amount of cables going from radar sites to the operator's computer.

TABLE OF CONTENTS

NEED FOR A NEW DATA ACQUISITION/CONTROLLER SYSTEM:

Analog signals received from Radars are usually weak and are further degraded by the long cables that carry them to the data acquisition system. Also, as experienced with the previous generation of data acquisition systems, long analog cables are susceptible to noise. Furthermore, the video display of the PC (Z-386) was found to introduce electrical noise into the signals carried by the analog cables being terminated at the PC.

SOLUTION:

Different pieces of the final solution simply fell into place by addressing the problems encountered in the previous data acquisition systems. Firstly, the long analog cables had to be replaced by relatively very short ones (less than 10 feet). Secondly, the analog cables had to be kept away from the video display of the PC, so that the noise introduced by it cannot corrupt the analog signals. This suggested that digitized data rather than analog should be brought into the PC. Figure 1.1 shows a block diagram of the new data acquisition system shaped by the preceding two constraints. The system consists of a host unit (operator's computer) and two remote units, the latter two being located at the radar site. The incorporation of two remote units rather than one for collecting data gives more flexibility in placing different radars spatially far apart without increasing the length of the analog cables.

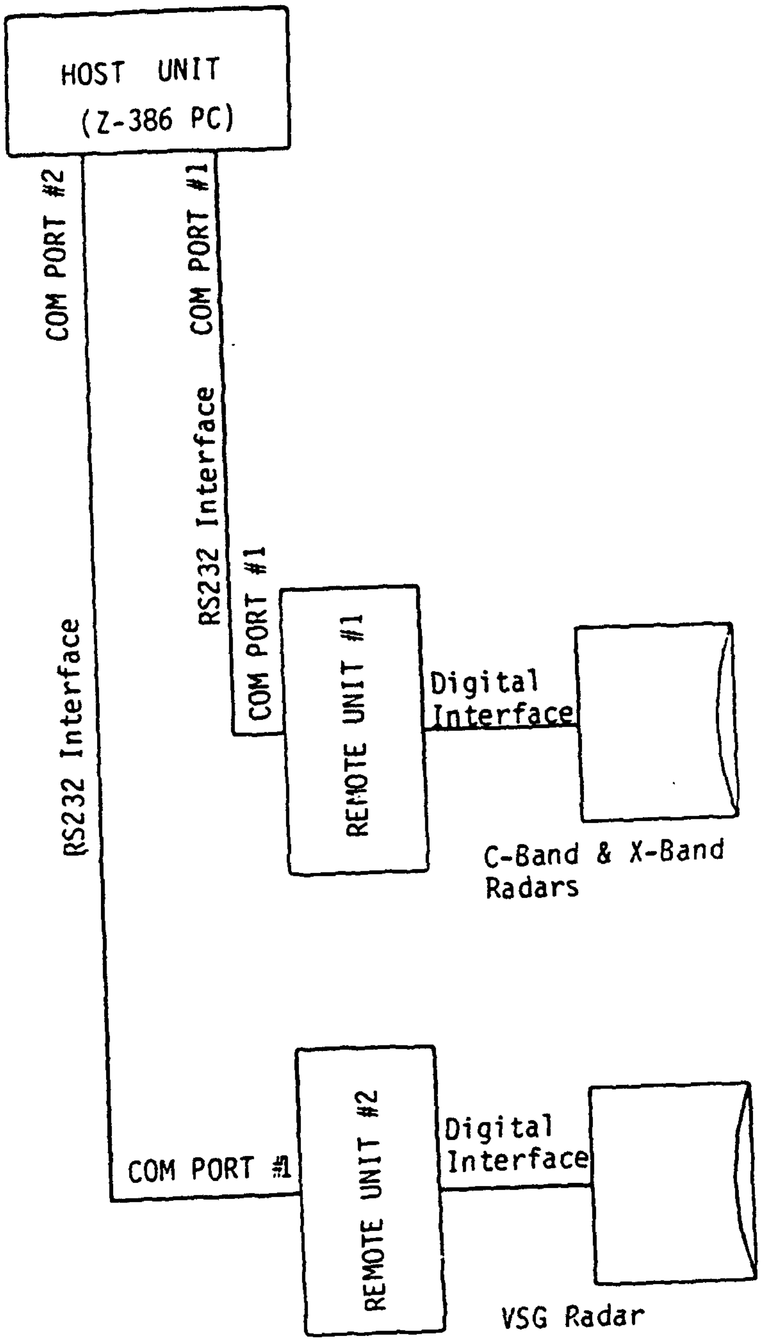


Figure 1.1 Layout of the new data acquisition/controller system.

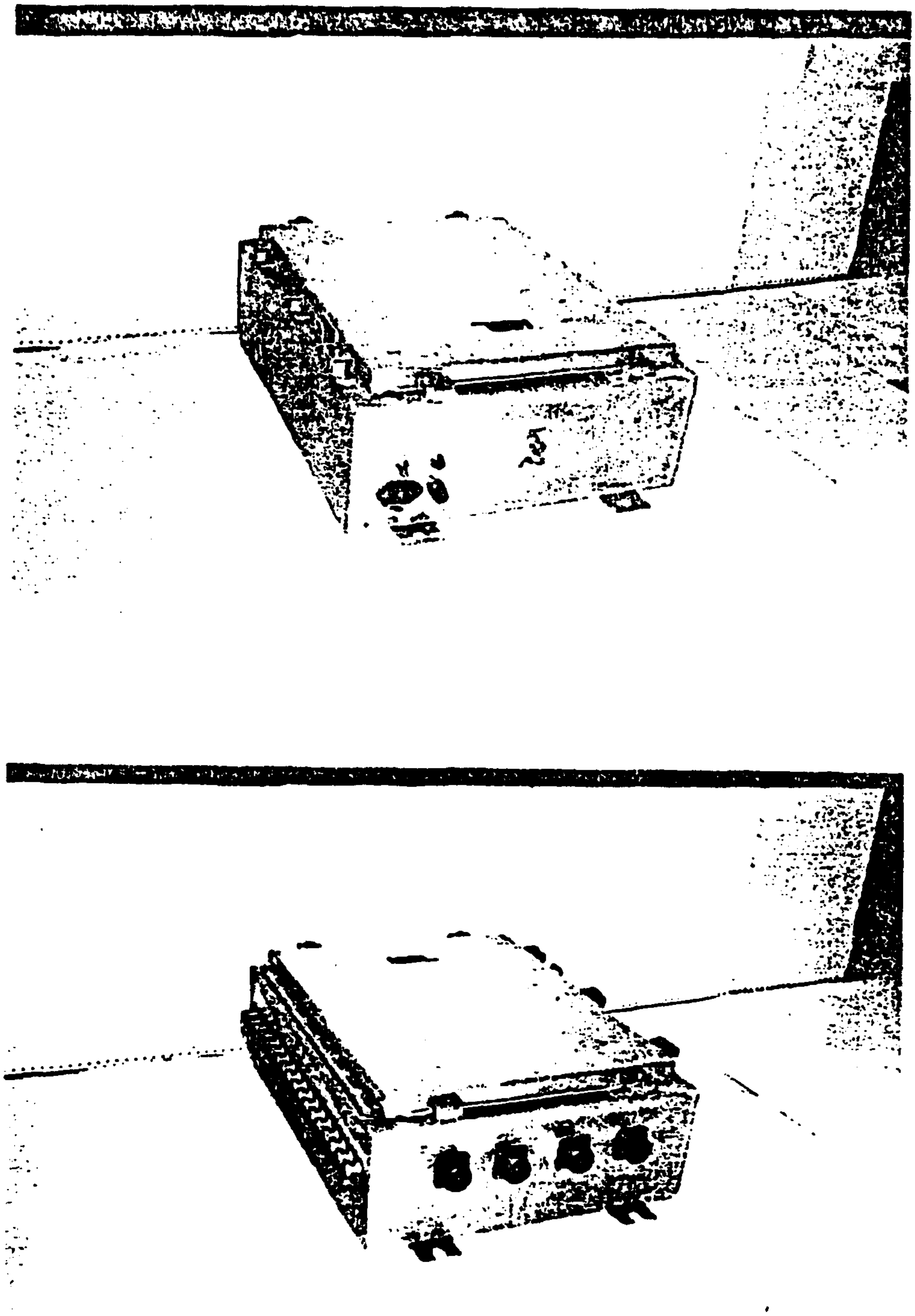


Figure 1.2 Photographs of a remote unit showing different sides.

weather resistent nature. This was one of the important factors in the design of the remotes because these remotes had to take on the harsh weather of North Atlantic.

Figure 1.2 shows two photographs of a remote unit. The top photograph shows the front side and the right side of the unit, the latter having the AC-power inlet and an ethernet coax-connector for future development. The bottom photograph shows the back side and the left side of the unit. The former carries sixteen coax-connecters for the 16 analog input channels. The left side is shown to incorporate four round connectors. The sixteen digital outputs are routed through the two right most connectors. The second connector from the left most connector is used for RS232 connection.

The Zenith386-based host unit has also an A/D I/O card identical to the one that is in a remote unit. This makes it possible to keep track of readings from other relevant equipment accessible in the vicinity of the host unit.

SYSTEM HOOK-UP:

Figure 1.3 shows a layout of the hook-up of the data acquisition system used in its first experiment. The back-panel of the Z386-based host unit shows the connections to the two remotes and the wave gauge equipment. As part of the back-panel, three relevant cards are shown each having two connectors. Starting from left, the first card is zenith's own serial communication card. The top connector on this card is COM port #1 which is used to connect remote #1 to the host. The second card is the A/D I/O card. Only one channel (#0) was used to note down readings from the wave gauge equipment. Pin #1 of the top connector is the signal and pin #2 is the corresponding ground for channel #0. The third

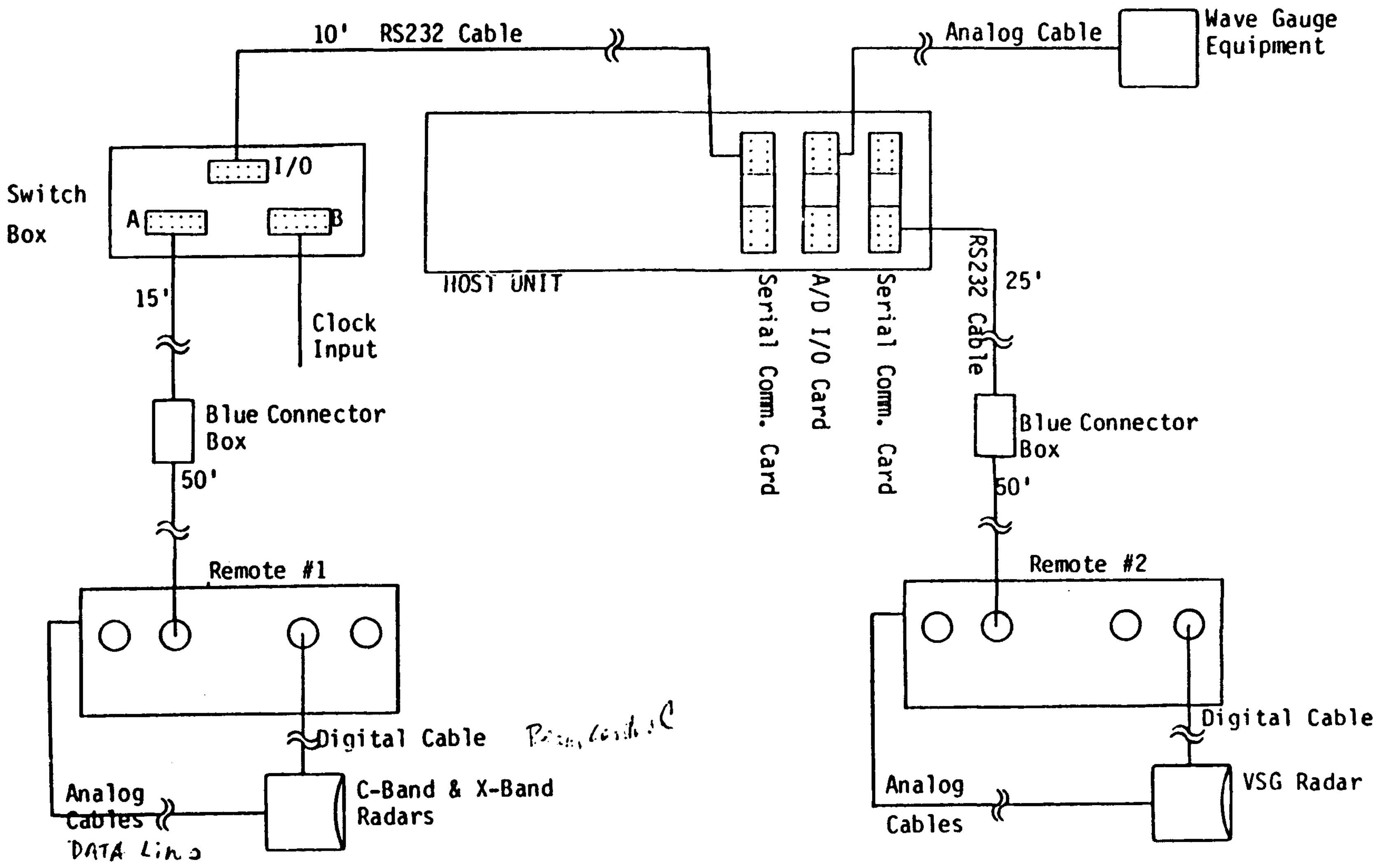


Figure 1.3 Hook-up of the data acquisition/controller system.

card (right most) is again a serial communication card having two ports. The lower connectoı is COM port #2 which is used to connect to remote #2.

The RS232 cable of remote #2 is shown to consist of two pieces joined by a blue box. The first portion is 25 feet long whereas the second portion is 50 feet long making the total length equal to 75 feet. Tł RS232 cable for remote #1 is same as that for remote #2 except that it goes through a switch box. The purpose of the switch box is to let the host access a universal clock for synchronizing its local clock with the other experiments going on in the same environment. Before starting to operate the host, channel B of the switch box is set by the operator and a clock-program is run to synchronize the clock of the host unit with the universal clock. Channel A of the switch box is then selected by the operator for normal operation.

On the remote side the RS232 cable terminates on a round 9-pin weather-resistent connector which is the second connector counting from the left end. From a remote two sets of cables are taken to the radar. One set consists of analog coaxial cables to route-in analog readings from the attached radar. The second set consists of a single 9-wire digital cable to deliver a maximum of eight digital outputs from a remote to its radar for controlling it. A total of sixteen digital outputs can be delivered, but it was found that only a small number was required for each radar. Hence, to make the two remote units universal, the first eight digital outputs are dedicated to remote #1 and the last eight to remote #2. In Figure 1.3 the digital cable for remote #1 is shown connected to the first round connector (counting from right) whereas the second connector, corresponding to the eight most

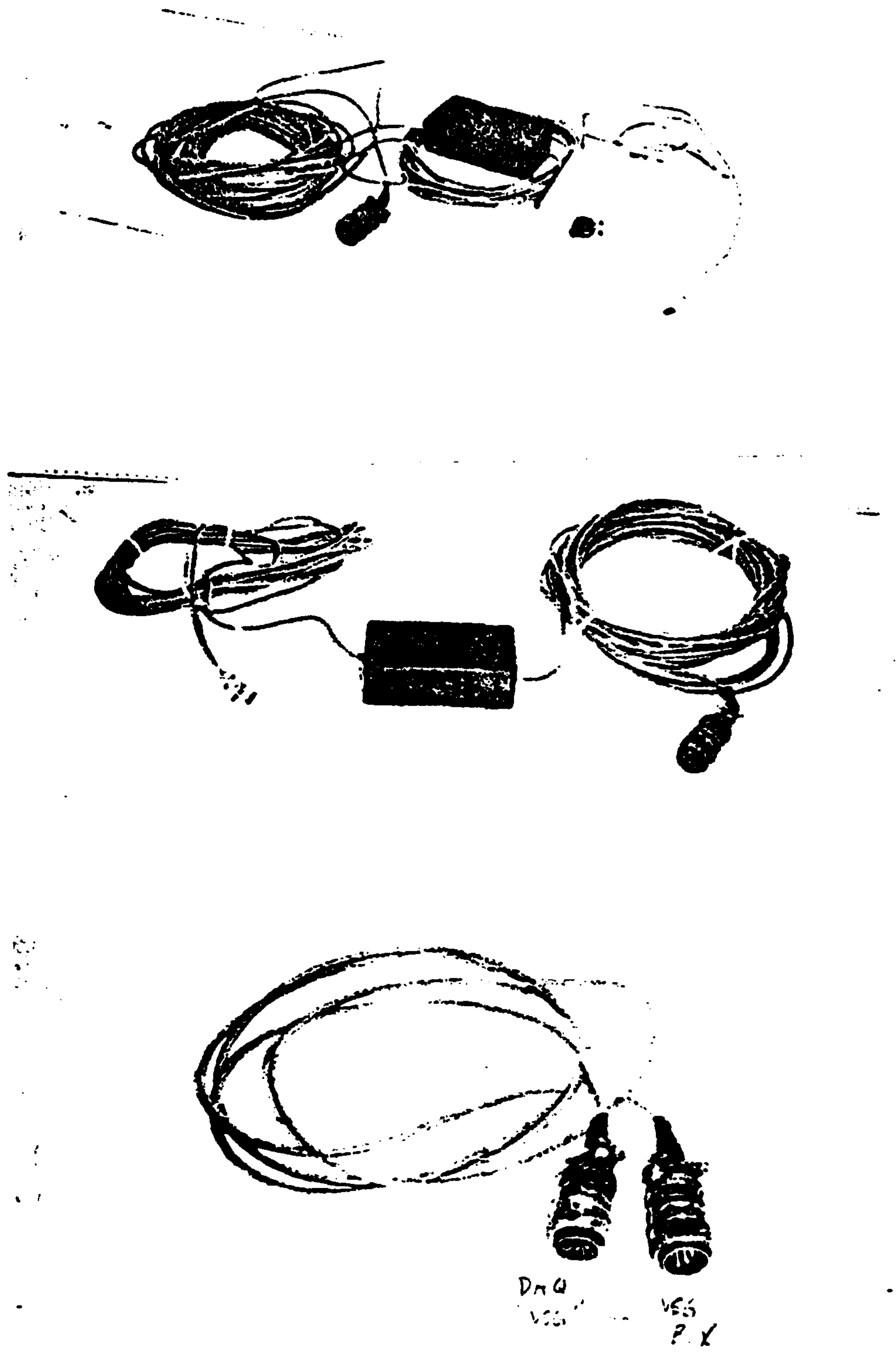


Figure 1.4 Photographs of different cables used in the system's hook-up.

significant bits, is used for remote #2. However, this division is at the software level and can be changed if the need arises to increase the number of digital outputs for a radar.

Figure 1.4 shows photographs of some important cables used in the hook-up of the data acquisition system. The topmost photograph is that of RS232 cable for remote #1. Shown is the switch box used for accessing either the universal clock for synchronization or the remote for normal operation. The blue box is the cable extender for increasing the length of the RS232 cable to a total of 75 feet. The middle photograph is that of the RS232 cable for remote #2, which is different from its peer in only that it lacks a switch box. The last photograph shows a 9-wire digital cable used for delivering digital output from a remote to the attached radar. The digital cables in either case of remote are identical. The 9-wire cable is used to carry a maximum of eight digital outputs. The ninth wire is used as a common ground reference.

It should be noted that the cables meant to be used at a remote site, or the cables leading to a remote site are incorporated with weather-resistent connectors to ensure proper operation even in a harsh weather, such as that of North Atlantic where this data acquisition system was used to collect ocean wave data.

COMMUNICATION AND SAMPLING RATES:

The host and the two remotes communicate via RS232 links at 38.4 KBits per second. Data length is set to 8 bits, parity is disabled, and 1 stop bit is specified. A commercially available communication software "COMMBIOS/COMMBUFF" is used which employs TSR (Terminate and Stay Resident) routines to store incoming data on serial lines into a memory buffer. A separate buffer is used for each serial port. The software written for the host and the remote units can then retrieve data from the buffers at the proper time. COM port #1 and #2 each have a buffer of 4KBytes. A message from any party to another does not exceed 25 bytes within the period in which the receiver checks the buffers at least once. Hence, the 4 Kbyte size of a buffer is more than enough.

Though sixteen analog channels are available at a remote, the current software loaded onto each remote unit makes it possible for the remote units to have 10 analog input channels each having a 12-bit resolution. All channels are sampled at 30 Hertz. But when the data samples are sent to the host unit, all data are not stored each time. Hence, the effective sampling rate for the channels varies. Figure 1.5 tabulates the sampling rates of different analog input channels for the two remotes and the host unit. Samples from channels 0 through 5 of remote #1 are not stored every time, rather they are stored every other time. However, instead of skipping the in-between sample, average of two samples is stored; the two samples being the sample which is received at the storage instance and the sample which was not stored previously. Similarly, samples from channels 6 through 9 of remote #1, and 8 through 9 of remote #2 are stored every 30 seconds or at the rate of 1/30 HZ. Samples from the rest of the channels, 0 through 7 of remote #2, and channel 0 of the host unit are stored every time, that is at 30 HZ.

The sampling rates given here are only the raw sampling rates of the channels. The actual sampling rates of the signals coming into the channels may differ as a function of how the attached equipment is delivering the signals. For example, in the actual experiment VSG radar was hooked to remote #2. The VSG radar had three beams, A, B, and C. The

Remote #1

Channel #s	Averaging Rate	Sampling Frequency
0-5	Average of 2 samples is stored (every 66 ms)	15 HZ
6-9	Average of 900 samples is stored (every 30 seconds)	1/30 HZ

Remote #2

Channel #s	Averaging Rate	Sampling Frequency
0-7	Each sample is stored (every 33 ms)	30 HZ
8-9	Average of 900 samples is stored (every 30 seconds)	1/30 HZ

Host

Channel #s	Averaging Rate	Sampling Frequency
0	Each sample is stored (every 33 ms)	30 HZ

*The effective sampling frequency may vary for any of the channels depending upon the mode of operation of the attached equipment. For example, remote #2 was attached to the VSG radar having three beams: A, B, and C. The beams were switched in a cyclic manner. Hence, the effective sampling rate of channels 0-7 of remote #2 for a particular beam was 10 HZ instead of 30 HZ.

Figure 1.5 Sampling rates of different channels as determined by storage of their samples.

beams were switched sequentially in a round-robin fashion. This was accomplished by the remote delivering digital control words to the radar via the dedicated digital cable. The switching of the beams was done at 30 HZ. Hence, within 3 sampling periods each beam was given a turn. This reduced the sampling rate to 10 HZ per beam. However, only the data samples coming through channels 0-7 were dependent upon the type of beam in effect. Therefore, the actual sampling rate of channels 8 through 9 remained at 1/30 HZ.

For the channels for which samples are not stored every time, the averaging is done in a skewed manner. The later samples are given more weight with respect to the earlier ones. As soon as a sample arrives, it is added with the previous average of the samples and then the result divided by 2. This algorithm yields the following formula for averaging:

$$X_0/2^n + \sum_{i=1}^{n}(X_i/2^{n-i+1})$$

where n represents the averaging of sample numbers 0 through n. In other words, n represents the number of samples skipped before their average is stored. Note that for n=1 (storing every other time), the average is not skewed, since the formula yields the following:

$$(X_0 + X_1)/2$$

DIGITAL CONTROL WORD:

As mentioned earlier, each remote provides 16 bits of digital output which can be used to control the attached equipment. The 16 digital outputs form the digital control_word of a remote. Figure 1.6 tabulates the function associated with each bit of the control_word of a remote. The digital control_word originally comes from the host unit as part of every

Control Word for remote #1

Bit #	
0	SW1
1	SW2
2	SW3
3	SW4
4-8	[Unused]
9	Power-on the C-Radar
10	Power-on the X-Radar
11-12	[Unused]
13	Put the remote into the RESET state
14	Put the remote into the OFF state
15	Restart the remote's software

Polarization	SW4	SW3	SW2	SW1
HH	1	1	1	1
HV	0	1	1	1
VH	1	1	0	1
VV	0	1	0	1
DLC	0	0	0	1
DLX	0	0	0	0

Control Word for remote #2

Bit #	
0-8	[Unused]
9	Power-on the VSG-Radar
10-12	[Unused]
13	Put the remote into the RESET state
14	Put the remote into the OFF state
15	Restart the remote's software

Switching of Beams Performed by Bits 13, 14, and 15

Antenna	Bit_{15}	Bit_{14}	Bit_{13}
A	0	0	1
B	1	0	0
C	0	1	0

Relation of Bits to the Physical pins of the Digital Output Connector

Left Connector		Right Connector	
Bit #	Pin #	Bit #	Pin #
0	A	8	A
1	B	9	B
2	C	10	C
3	D	11	D
4	E	12	E
5	F	13	F
6	G	14	G
7	H	15	H
	I (GROUND)		I (GROUND)

Figure 1.6 Functional and physical bit-assignments for the Control_Word.

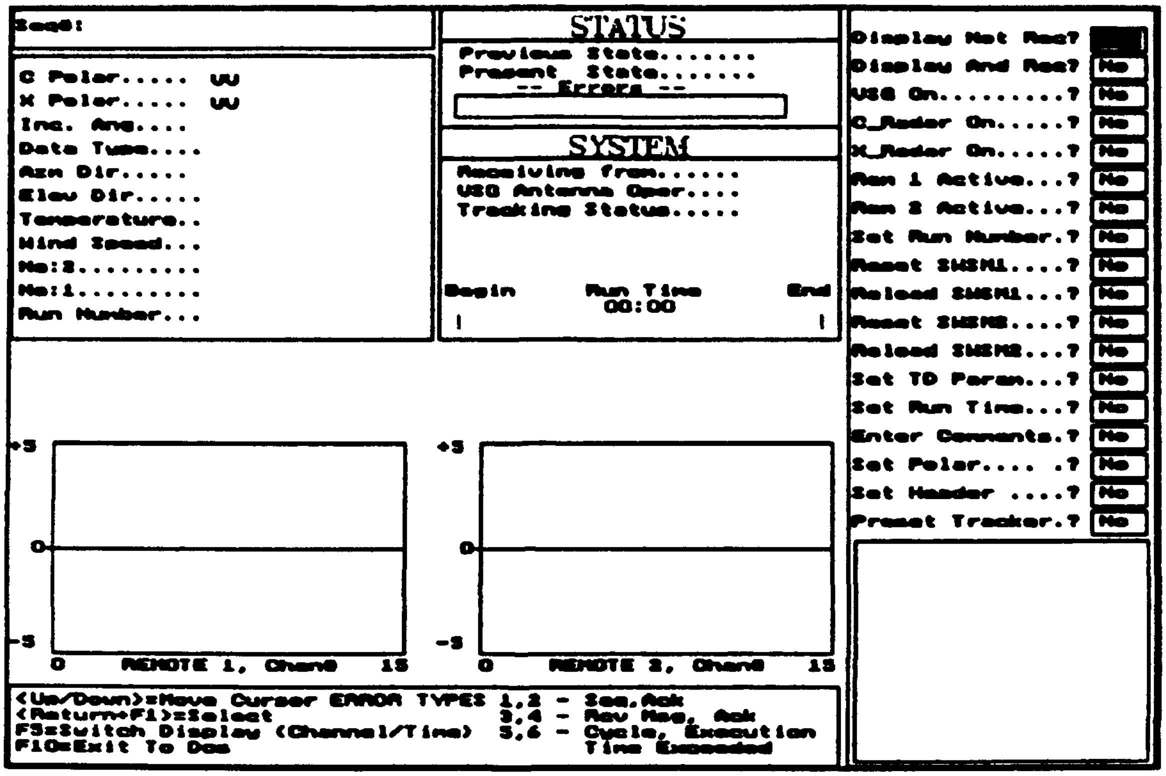
message delivered to a remote. Some of the bits act as commands to perform functions on the destination remote itself. For example, moving a remote to a software RESET or OFF state. Hence, such bits are not meant to be directly delivered to the attached equipment. Instead of wasting these bits, each remote is programmed to make use of them for providing continuous switching of beams for VSG radar. Bit_{15}, Bit_{14}, and Bit_{13} are shown to perform this function. It should be noted that no matter which remote it is, these bits have a universal function of providing the control for switching VSG antennas/beams A, B, and C. This was done to make the two radars compatible and universal.

With the exception of $Bit_{15:13}$, all other bits are directly relayed to the outside world via the two round 9-pin digital output connectors. That is to say, its the host unit's responsibility to format the correct bit-configuration for a particular operation that needs to be performed on the attached equipment. For example, the first four bits of the digital control_word of remote #1 were used for setting the "polarization" of the C-Band and X-band radars. The software at the host sets the required bit-configuration once the operator selects the polarization-type via the user-interface.

As shown in the figure, most of the bits of a control_word were not used. However, the software can be changed to add more functions as the need arises.

OPERATION AND THE USER-INTERFACE:

The data acquisition system is controlled by an operator at the host unit using graphical user-interface. Figure 1.7 shows the screen-setup of the user-interface. The screen is basically divided into three sections. The first column on the right is the input section where



```
Seq#:                                   STATUS
                                  Previous State.......    Display Not Rec?  [  ]
C Polar.....  VV                  Present  State.......    Display And Rec?  [No]
X Polar.....  VV                       -- Errors --        USG On.........?  [No]
Inc. Ang....                                               C_Radar On.....?  [No]
Data Type....                           SYSTEM             X_Radar On.....?  [No]
Azm Dir.....                      Receiving from......     Rem 1 Active...?  [No]
Elev Dir.....                     USG Antenna Oper....     Rem 2 Active...?  [No]
Temperature..                     Tracking Status.....     Set Run Number.?  [No]
Wind Speed...                                              Reset SWSM1....?  [No]
No:2.........                    Begin    Run Time    End  Reload SWSM1...?  [No]
No:1.........                              00:00           Reset SWSM2....?  [No]
Run Number...                    |                      |  Reload SWSM2...?  [No]
                                                           Set TD Param...?  [No]
                                                           Set Run Time...?  [No]
                                                           Enter Comments.?  [No]
                                                           Set Polar.... .?  [No]
                                                           Set Header ....?  [No]
                                                           Preset Tracker.?  [No]

<Up/Down>=Move Curser ERROR TYPES 1,2 - Seq.Ack
<Return+F1>=Select                3,4 - Rev Msg, Ack
F3=Switch Display (Channel/Time)  5,6 - Cycle, Execution
F10=Exit To Dos                         Time Exceeded
```

Figure 1.7 Screen-setup of the User-Interface at the Host Machine.

the operator may select different options involved in running the data acquisition system. The second section is the real-time display area in which sample values of all the active channels are shown in real-time. The rest of the screen displays either instructions/warnings, errors, system status, run-time elapsed etc.

In the input section, the option "Display Not Rec (Record)" can be selected for displaying the samples but not recording them. This option is usually used to test the integrity of signals received from the radars before proceeding to run a lengthy session of recording the data by selecting "Display and Rec" option. The next three options of "VSG On", "C_Radar On", and "X_Radar On" were meant to switch the VSG, C-Band, and X-Band radars, respectively, on or off.

It is not necessary to make both remotes operational if one is interested in collecting data from only one. "Rem 1 Active" and "Rem 2 Active" are available to select either or both the remotes to be operational. An inactive remote if powered-up hangs in its software OFF-state waiting for the host unit to tell it to start. It is possible that a remote is powered-down or does not exist. In such a case, it is essential that the operator marks it as inactive otherwise the host unit will send messages to and expect messages from it, hence, causing errors.

A remote can be put into its RESET software-state for synchronization with the over all operation. "Reset SWSM1 (SoftWare State Machine)" and "Reset SWSM2" are included in the user-interface for this purpose. Also, earlier it was thought that if less serious problems occur in synchronization of a remote and the host unit while a run is being performed, it may be remedied by restarting the software of the problematic remote. However, the communication protocol was made robust enough so that synchronization problems are handled gracefully. Hence, the possibility of correcting problems via restarting the software remotely was alleviated. If a remote gets stuck at some point the only way to give it sanity is to physically reset it. It should be noted that in the course of using the system in the North Atlantic experiment the need to physically reset the remotes never arose. Hence, "Reload SWSM1" and "Reload SWSM2" options were not implemented.

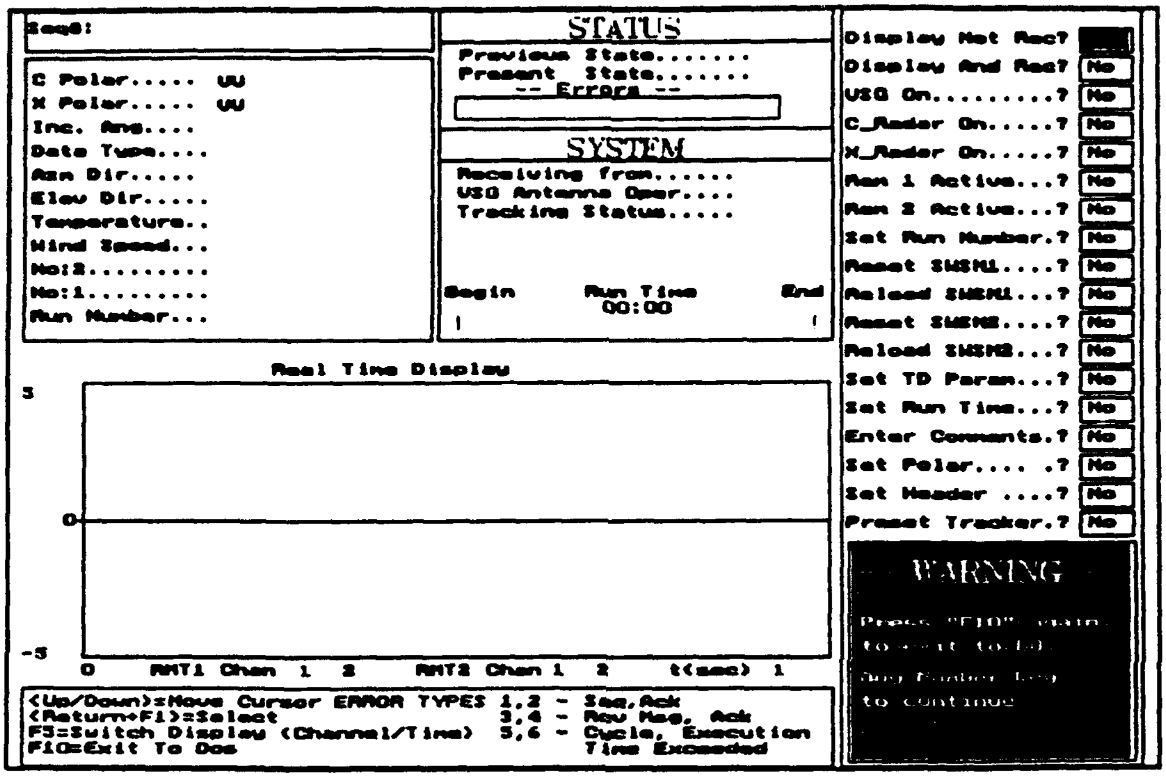
"Preset Tracker" option lets the operator preset tracking if the radar loses it. Similarly "Set Polar (Polarization)" is used to set the polarization of X-band and C-band radars. "Set Run Number" is used to set the run number of the upcoming run. This run number becomes the extension of the filename that will store the data. Hence, the run number cannot exceed 999. "Enter Comments" lets the user enter comments regarding "Inc. Ang (Inclination Angle)", "Data Type", "Azm Dir (Azimuth Direction)", "Elev Dir (Elevation Direction)", "Temperature", "Wind Speed", "No:2 (Remote #2)", and "No:1 (Remote #1)." This information is considered as comments because it has no affect upon the data being collected. However, this information is useful for later processing and interpretation of data. Similarly, "Set Header" lets the user enter 256 bytes of information as the header of the file which will store the data generated during the upcoming run.

Once the above mentioned parameters are set, the system is ready to be run for a specified amount of time. "Set Run Time" is used to set the length of the run. In order to start a session, the cursor is moved to either "Display Not Rec" or "Display And Rec" option and No is changed to Yes. The host then starts the remotes, and data samples begin to flow in. The data are displayed in the real-time display area. Here, all the active channels of the active remotes are displayed. The real-time display can be switched such that two particular channels are displayed as a function of time. This can be done by pressing *F5 function* key before starting a run session. Figure 1.8 shows the screen setup with the real-time display area switched.

During a run-session errors may occur. These may be communication errors, sampling-period time exceeded error, invalid message received error etc. In such a case, it is helpful to know why the error occurred, where it originated from, in which software-state did it occur, and so on. For debugging purposes, certain statuses are displayed. These include, "Previous State", "Present State", "Receiving from", "VSG Antenna Oper (Operational)", "Tracking Status", and type of error if any occurred. If an error occurs once or twice a run, it is no cause for concern. However, more frequent occurrence of errors must be investigated, and these statuses will then be helpful in pin-pointing the source of errors.

A **Bar clock** is provided which shows the fraction of time elapsed/remaining. Also, **Sequence number** is updated as the time progresses. Actually, the sequence number is the 32-bit time stamp of the system clock.

If data are being collected in a run-session, a data file is automatically named and opened. The current software opens it on drive "d:" which is the Bernoulli diskette drive. The name of the data file consists of two parts; eight digit name and a three digit extension. The extension of the filename is the run number specified by the user, whereas the eight-digit-name is formed from the local time stamp at the instance of the file's creation.



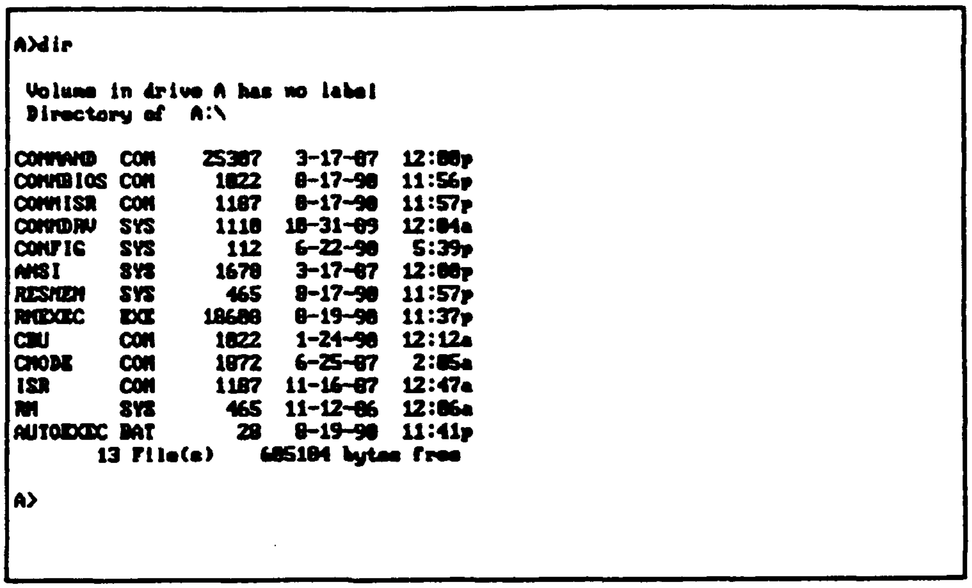
```
Seq#:

C Polar.....  VV
X Polar.....  VV
Inc. Ang....
Data Type....
Azm Dir.....
Elev Dir.....
Temperature..
Wind Speed...
No:2.........
No:1.........
Run Number...

STATUS
Previous State.......
Present  State.......
     -- Errors --

SYSTEM
Receiving from......
USG Antenna Oper....
Tracking Status.....

Begin        Run Time        End
               00:00

Display Not Rec?
Display And Rec?  No
USG On.........?  No
C_Radar On.....?  No
X_Radar On.....?  No
Ram 1 Active...?  No
Ram 2 Active...?  No
Set Run Number.?  No
Reset SWSM1....?  No
Reload SWSM1...?  No
Reset SWSM2....?  No
Reload SWSM2...?  No
Set TD Param...?  No
Set Run Time...?  No
Enter Comments.?  No
Set Polar.... .?  No
Set Header ....?  No
Preset Tracker.?  No

WARNING
Press "F10" again
to exit to DOS
Any Number Key
to continue

<Up/Down>=Move Cursor ERROR TYPES 1,2 - Seq,Ack
<Return+F1>=Select                3,4 - Rqv Msg, Ack
F5=Switch Display (Channel/Time)  5,6 - Cycle, Execution
F10=Exit To Dos                         Time Exceeded
```

Figure 1.8 A variation in the screen-setup of the User-Interface at the Host Machine.

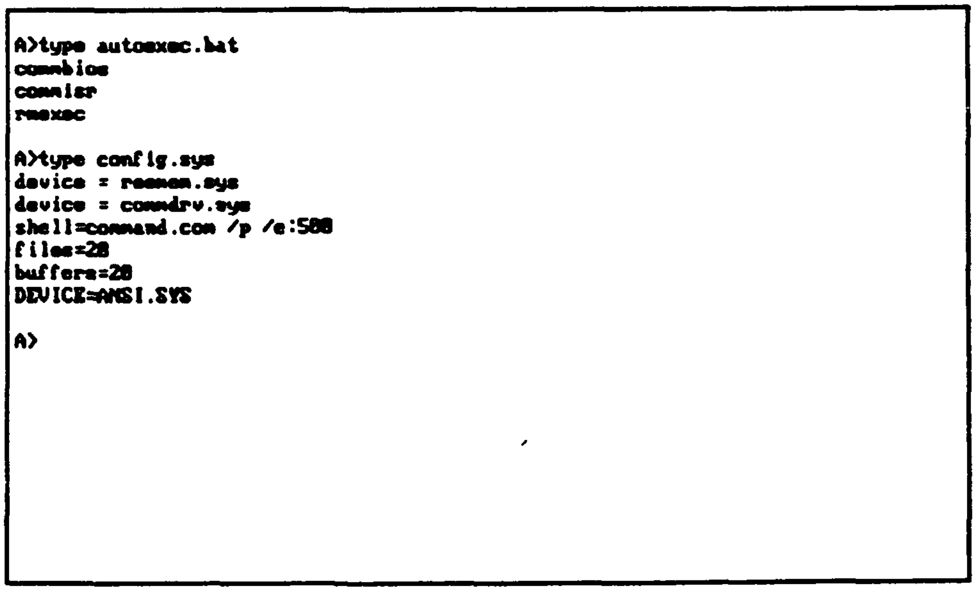
SOFTWARE SETUP:

The software that comes along with the hardware of the data acquisition system is spread over several parts. Firstly, **DOS version 3.3** is required at the host and the two remotes. Appendix D contains two boot-up disks; one for the host and the other for a remote. The second piece of the software comprises of routines to setup communication buffers for receiving data flowing in through the RS232 links. These are TSR routines and must be loaded at the start up time. Figure 1.9 lists the contents of the directory of the boot-up disk of a remote, and the contents of CONFIG.SYS and AUTOEXEC.BAT files. When the system is started, the TSR routines are setup and buffers are created in the RAM. The boot-up disk of the host is similar, except its AUTOEXEC.BAT file terminates after running TSR routines. That is to say, it does not run "HMEXEC" like its peer at a remote which runs "RMEXEC" to start the third piece of the software which is the actual software of the data acquisition system. The operator is responsible for invoking the software by typing MAIN at the DOS prompt. Since, remote is on its own, the software is run automatically at the boot-up time through the AUTOEXEC.BAT file. Figure 1.10 shows the messages appearing on the screen of the host unit at the boot-up time.

Another difference between the boot-up disks of a remote and the host is that the CONFIG.SYS file has "device = rcd.sys" statement for adding the bernoulli drive to the system's resources. Also, "break = on" statement is added, and "files = 20" statement is modified to "files = 40". Some additional files appear in the host's boot-up disk directory. These are RCD.SYS (for bernoulli), MAIN.EXE (host's software in the executable form),

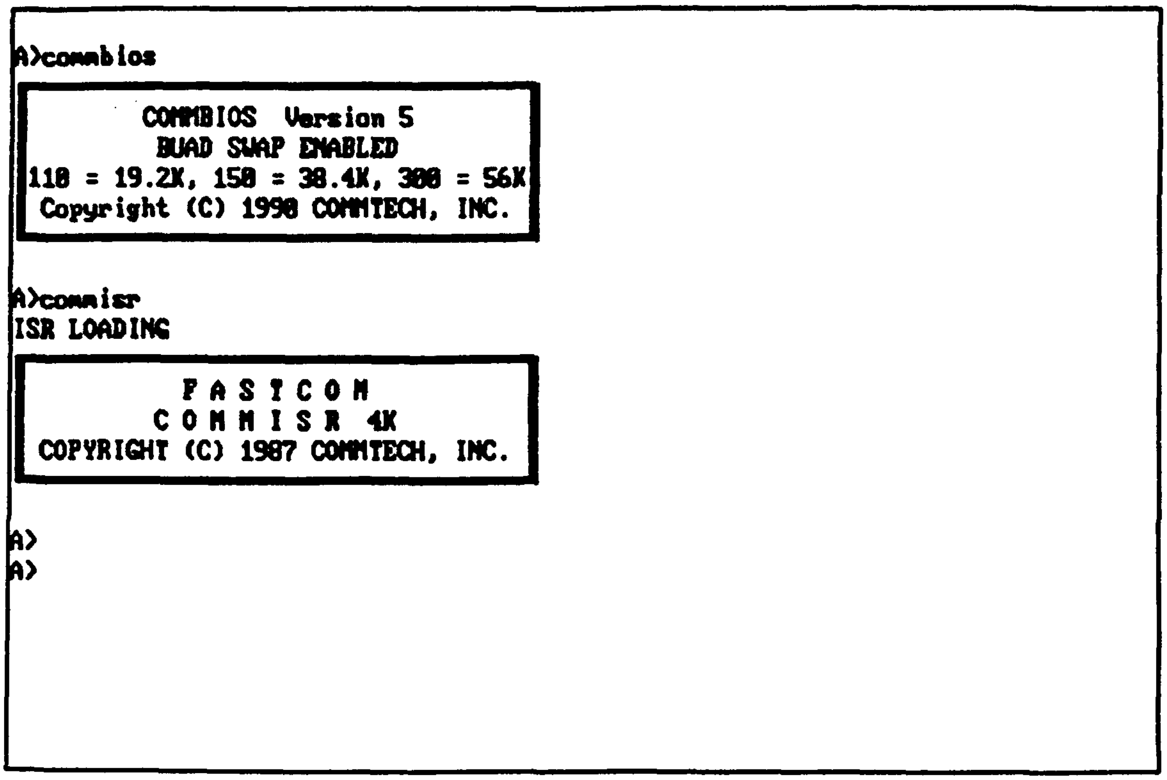


```
A>dir

 Volume in drive A has no label
 Directory of  A:\

COMMAND  COM    25307   3-17-87  12:00p
COMMBIOS COM     1022   8-17-90  11:56p
COMMISR  COM     1187   8-17-90  11:57p
COMMDRV  SYS     1110  10-31-89  12:04a
CONFIG   SYS      112   6-22-90   5:39p
ANSI     SYS     1670   3-17-87  12:00p
RESMEM   SYS      465   8-17-90  11:57p
RMEXEC   EXE    18600   8-19-90  11:37p
CBU      COM     1022   1-24-90  12:12a
CMODE    COM     1072   6-25-87   2:05a
ISR      COM     1187  11-16-87  12:47a
RM       SYS      465  11-12-86  12:06a
AUTOEXEC BAT       20   8-19-90  11:41p
       13 File(s)    685104 bytes free

A>
```



```
A>type autoexec.bat
commbios
commisr
rmexec

A>type config.sys
device = resmem.sys
device = commdrv.sys
shell=command.com /p /e:500
files=20
buffers=20
DEVICE=ANSI.SYS

A>
```

Figure 1.9 Remote's boot-up directory and the contents of AUTOEXEC.BAT & CONFIG.SYS files.



```
A>commbios

        COMMBIOS  Version 5
        BUAD SWAP ENABLED
110 = 19.2K, 150 = 38.4K, 300 = 56K
 Copyright (C) 1990 COMMTECH, INC.

A>commisr
ISR LOADING

            F A S T C O M
          C O M M I S R  4K
 COPYRIGHT (C) 1987 COMMTECH, INC.

A>
A>
```

Figure 1.10 Messages appearing on the screen of the host at the boot-up time.

EGAVGA.BGI, and *.CHR files of TURBO PASCAL 5.5 for providing the graphics environment.

Since a remote unit is not equipped with a floppy drive, the solid-state hard disk at a remote is the boot-up disk. A computer is used first to format the solid-state disk and to load the relevant files on it. The solid-state disk is then moved to its remote. At power-on time, the system boots-up using this hard disk.

MEMORY STORAGE SIZE REQUIREMENT:

The data that are being gathered have to be stored on a disk for later processing. In this section some helpful calculations are presented for knowing the amount of storage required for a particular length of a run. Each active remote sends 10 data samples each being 12 bits wide. All the data samples received are not stored every time. Figure 1.5 tabulated the frequency of storage of samples from different channels. Hence, the size of the data packet fluctuates from time to time. However, the header of the data packet, which consists of essential information about the packet, is fixed at 8 bytes.

Two samples occupy 3 bytes of storage because one sample is 12 bits wide. Hence, 10 data samples flowing in from each remote consume 15 bytes amounting to 30 bytes. The wave gauge sample from the host takes 2 bytes. The net size of the data packet comes out to be 40 bytes long. But this largest size would only occur once every 30 seconds. Keeping in view the numbers given in Figure 1.5 the total number of bytes required every minute is calculated as follows:

Number of bytes stored
every 1/30th of a second =8 bytes (header) +
2 bytes (wave gauge equipment channel) +
12 bytes (8 channels of remote #2)

Number of bytes stored
every 1/15th of a second =2 * (8 + 2 + 12) bytes +
9 bytes (6 channels of remote #1)

Number of bytes stored every second = 15 * (2 * (8 + 2 + 12) + 9)

Number of bytes stored
every 30 seconds = 30 * (15 * (2 * (8 + 2 + 12) +9)) +
6 bytes (4 channels of remote #1) +
3 bytes (2 channels of remote #2)

Number of bytes stored every minute = 2 * (30 * (15 * (2 * (8+2+12) + 9)) + 9)
= 47718

The above figure of 47718 bytes/minute translates into 2863080 bytes/hour. The bernoulli diskette used for storage has a capacity of 20 megabytes or 20971520 bytes. Hence, one bernoulli diskette can pack a little more than 7 hours worth of data. These figures are for the case when data from both remotes are being collected. That is to say that both the remotes are active. If only remote #1 is active:

Number of bytes per minute = 2 * (30 * (15 * (2 * (8 + 2) + 9)) + 6)
= 26112

Thus, 1566720 bytes of storage are required every hour. This implies that 20 megabytes of bernoulli diskette can store a little more than 13 hours of data if only remote #1 is active. Now, considering that only remote #2 is active:

$$\text{Number of bytes per minute} = 2 * (30 * (15 * (2 * (8 + 2 + 12))) + 3) = 39606$$

Therefore, 2376360 bytes of storage are needed for every hour of a run. A 20Meg bernoulli diskette can pack almost 9 hours (8.8 hours) worth of data if only remote #2 is active. Following is a summary of the above calculations:

Bytes/hour (both remotes active) = 2863080 (≈7.3 hours on a 20Meg bernoulli)

Bytes/hour (only remote #1 active) = 1566720 (≈13.3 hours on a 20Meg bernoulli)

Bytes/hour (only remote #2 active) = 2376360 (≈8.8 hours on a 20Meg bernoulli)

DATA STORAGE AND RETRIEVAL:

A data file, which is generated for a run, consists of a 256 bytes of header followed by a number of data packets. As explained in the previous section, the data packets fluctuate in size. Figure 1.11 shows the format of a data file and a data packet of largest possible size. The 256-byte header contains information and comments entered by the user before initiating a run. A data packet consists of a maximum of 2 messages. Message #2 entirely consists of data gathered form remote #2, and hence, may not exist of remote #2 is inactive. Message #1 consists of time stamp, data status word, sample from the wave gauge equipment, and data samples received from remote #1. Since not all samples are stored every time, the two messages may be of different size at different instances.

The first byte of a data packet contains the size-in-bytes of message #1. Similarly, the second byte specifies size of message #2. The next four bytes (2 through 5) comprise of the

Header (256 bytes)
Data Packet #0
Data Packet #1
Data Packet #2
⋮
Data Packet #n

(a)

Byte #		
0	Size of message #1	
1	Size of message #2	
2 3 4 5	Time_Stamp	Stored each time (30 HZ)
6 7	Data_Status_Word	
8 9	Sample from "Wave Gauge" equipment	
10 11 12	Samples from channels 0 & 1 of remote #1	
13 14 15	Samples from channels 2 & 3 of remote #1	Stored every other time (15 HZ)
16 17 18	Samples from channels 4 & 5 of remote #1	
19 20 21	Samples from channels 6 & 7 of remote #1	
22 23 24	Samples from channels 8 & 9 of remote #1	Stored every 30 seconds
25 26 27	Samples from channels 0 & 1 of remote #2	
28 29 30	Samples from channels 2 & 3 of remote #2	
31 32 33	Samples from channels 4 & 5 of remote #2	Stored every time (30 HZ)
34 35 36	Samples from channels 6 & 7 of remote #2	
37 38 39	Samples from channels 8 & 9 of remote #2	Stored every 30 seconds

(b)

Figure 1.11 (a) Format of the data storage file for a RUN.
(b) Format of a data packet which is the basic unit of data storage.

time stamp. Bytes 6 through 7 carry the 16-bit data status word which consists of information about the integrity of data contained in the packet, and it also specifies the VSG_Antenna (or beam) to which the data of remote #2 corresponds. The most significant byte of the 16-bit data status word is in byte #6 and the least significant byte in byte #7. The format of the data status word is given in Figure 1.12.

Bytes 8 through 9 of a data packet correspond to the sample collected from the wave gauge equipment. The rest of the bytes of message #1 consist of data samples collected from remote #1. Message #2 starts where message #1 ends. All the bytes of message #2 consist of data samples received from remote #2.

As can be noted from the format of the data packet given in Figure 1.11, two 12-bit data samples are stored in 3 bytes. It is important to know how the actual storage is done for proper retrieval later. Figure 1.13 shows the position of each bit of a pair of data samples stored in 3 consecutive bytes.

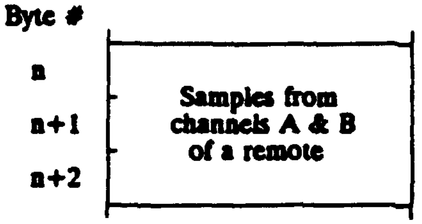
In case of the wave gauge sample, since a pair of data samples does not exist, the storage is a little different. The 8 least significant bits of a 12-bit sample is stored in byte #9 of a data packet and the 4 most significant bits are stored in the lower nibble of byte #8.

The 32-bit time stamp is the same as that made available to the DOS machine. The most significant byte is in byte #2 and the least significant byte is in byte #5.

Data retrieval is done based upon the information given above. However, it becomes quite involved due to checking of the data status word for possible errors, keeping track of which data belongs to which VSG_antenna in case of remote #2, extraction of data samples corresponding to specific channels, and taking averages etc. Appendix D lists a routine

Bit #			
0	Host Data is OK	(Channel #0)	
1	Host Data is OK		[Unused]
2	Host Data is OK		[Unused]
3	Remote #1 Data is OK		[Unused]
4	Remote #1 Data is OK	(Channels 0-5)	
5	Remote #1 Data is OK	(Channels 6-9)	
6	Remote #2 Data is OK	(Channels 0-7)	
7	Remote #2 Data is OK		[Unused]
8	Remote #2 Data is OK	(Channels 8-9)	
9			
10			
11			
12			
13	Data Corresponds to VSG_Antenna_A		
14	Data Corresponds to VSG_Antenna_B		
15	Data Corresponds to VSG_Antenna_C		

Figure 1.12 Bit assignment of the "Data_Status_Word."



12-bit sample from channel A ====> $A_{11}A_{10}A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$

12-bit sample from channel B ====> $B_{11}B_{10}B_9B_8B_7B_6B_5B_4B_3B_2B_1B_0$

Byte #	n	n+1	n+2
	$A_7A_6A_5A_4A_3A_2A_1A_0$	$A_{11}A_{10}A_9A_8$ $B_{11}B_{10}B_9B_8$	$B_7B_6B_5B_4B_3B_2B_1B_0$

Figure 1.13 Diagram showing how 2 12-bit samples are stored in 3 consecutive bytes in the memory.

written in C language which converts the data samples from a data file into a format acceptable by MATLAB software package for post-processing of data. The user specifies the data filename and couple of other parameters for the conversion to take place.

The relationship of a 12-bit sample to its analog value is based on the fact that the Analog to Digital Converter (ADC), used in the A/D I/O card employed in the system's units, has an analog range of +5 volts to -5 volts. The 12-bit binary equivalent is in offset-binary form; 000 Hex corresponds to -5 volts and FFF Hex corresponds to +5 volts. Hence, the range of 10 volts (-5 volts to +5 volts) is distributed over 4096 (12 bits) quanta. Each step or quantum corresponds to 10/4096, or 5/2048 for half range. Therefore to get the analog equivalent, 2048 is first subtracted from the digital value (offset) and the result is then multiplied by 5/2048:

$$\text{Analog value} = (\text{Binary value} - 2048) * 5/2048$$

HARDWARE NOTES:

Earlier the different hardware components of the data acquisition system were given an overview. This section gives a little more detail about the individual hardware components involved in the design of a remote unit. As mentioned previously, the design is based upon Intel 8088 Wild-Kit board. Figure 1.14 presents photographs of a remote unit showing the wild-kit board and other basic interface circuitry. There are three boards visible. The larger green board, situated alongside the power supply, is the wild-kit board. Three white slots (on the left) and three black slots are available on the board. The left most white slot

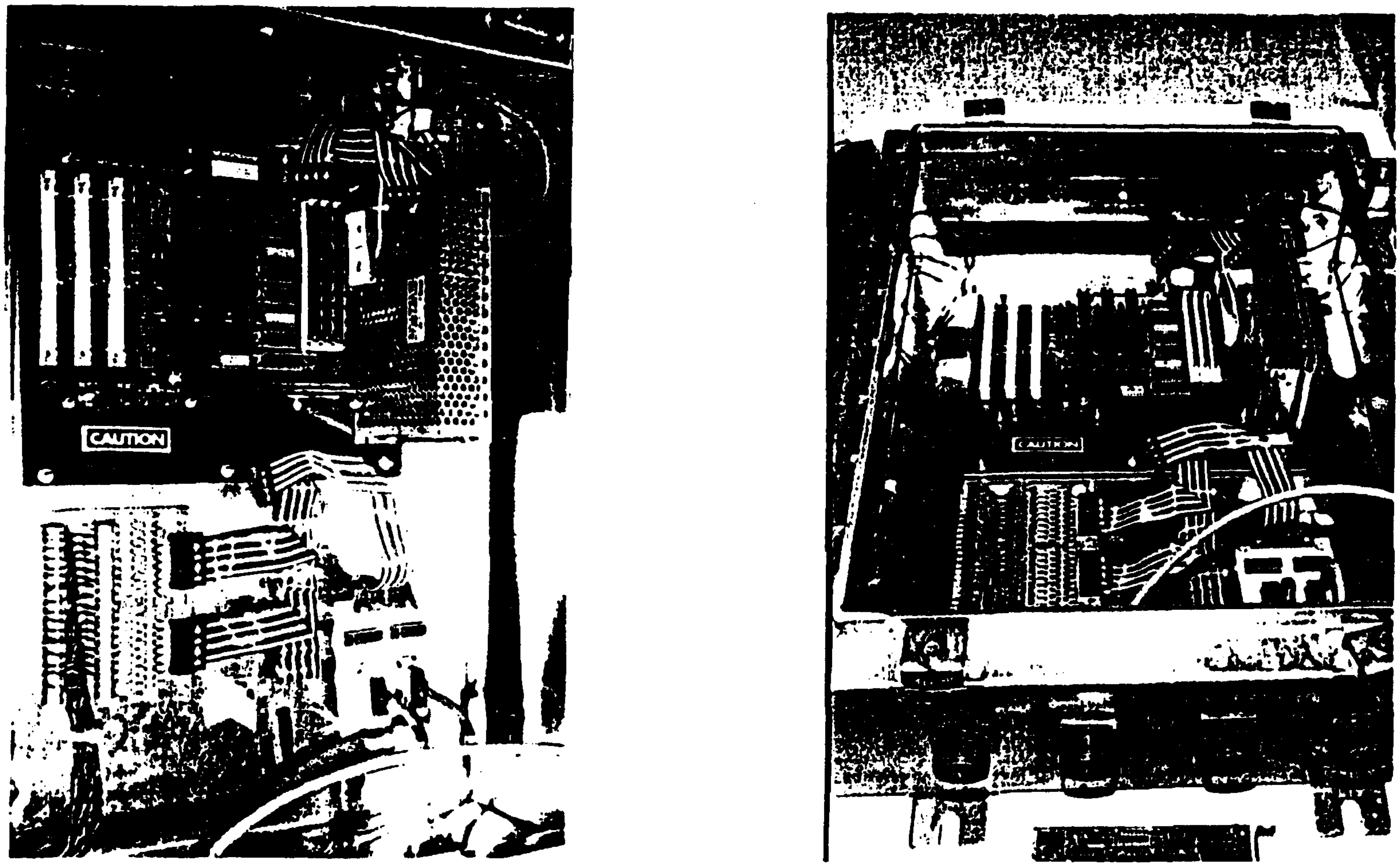


Figure 1.14 Photographs of a remote unit showing internal circuitry.

receives the Intel 8088 wild-card. The three black slots are meant to accommodate up to three PC compatible cards. An RS232 serial communication card, a solid state hard disk emulator, and an A/D I/O card are the three PC compatible cards that are plugged into the wild-kit board. On the right side of the wild-kit board are three more white sockets for receiving three SIMMs each being 256 KBytes. The smaller green board shown in the photographs is the analog interface between the A/D I/O card and the sixteen analog channels. This board consists of a resistor-network acting as voltage dividers for the incoming signals. However, in the current design this network is bypassed, and hence, the signals undergo no change. Red and black wires are used as twisted pairs for bringing the signals from the coax connectors to this interface board. The red wire in a pair carries the signal whereas the black is the corresponding ground. The smaller cream colored wire-wrapped board in the bottom right corner of the photographs consists of digital interface circuitry for driving the sixteen digital output signals on to the out going digital cables. These signals make up the 16-bit control word described earlier.

Figure 1.15 shows three photographs; the two photographs at the top capture the two sides of the Intel 8088 wild-card whereas the third photograph is that of the solid state hard disk emulator showing a battery on the top right corner of the card. This battery keeps the contents of the hard disk alive in the RAM ICs used for storage.

Figure 1.16 shows photographs of the RS232 serial communication card (top photograph) and the A/D I/O card. The serial communication card shows two connectors. The one at the top, to which is hooked a cable, is COM port #1 and the bottom connector belongs to COM port #2. The remotes always use COM port #1 for communicating with the host.

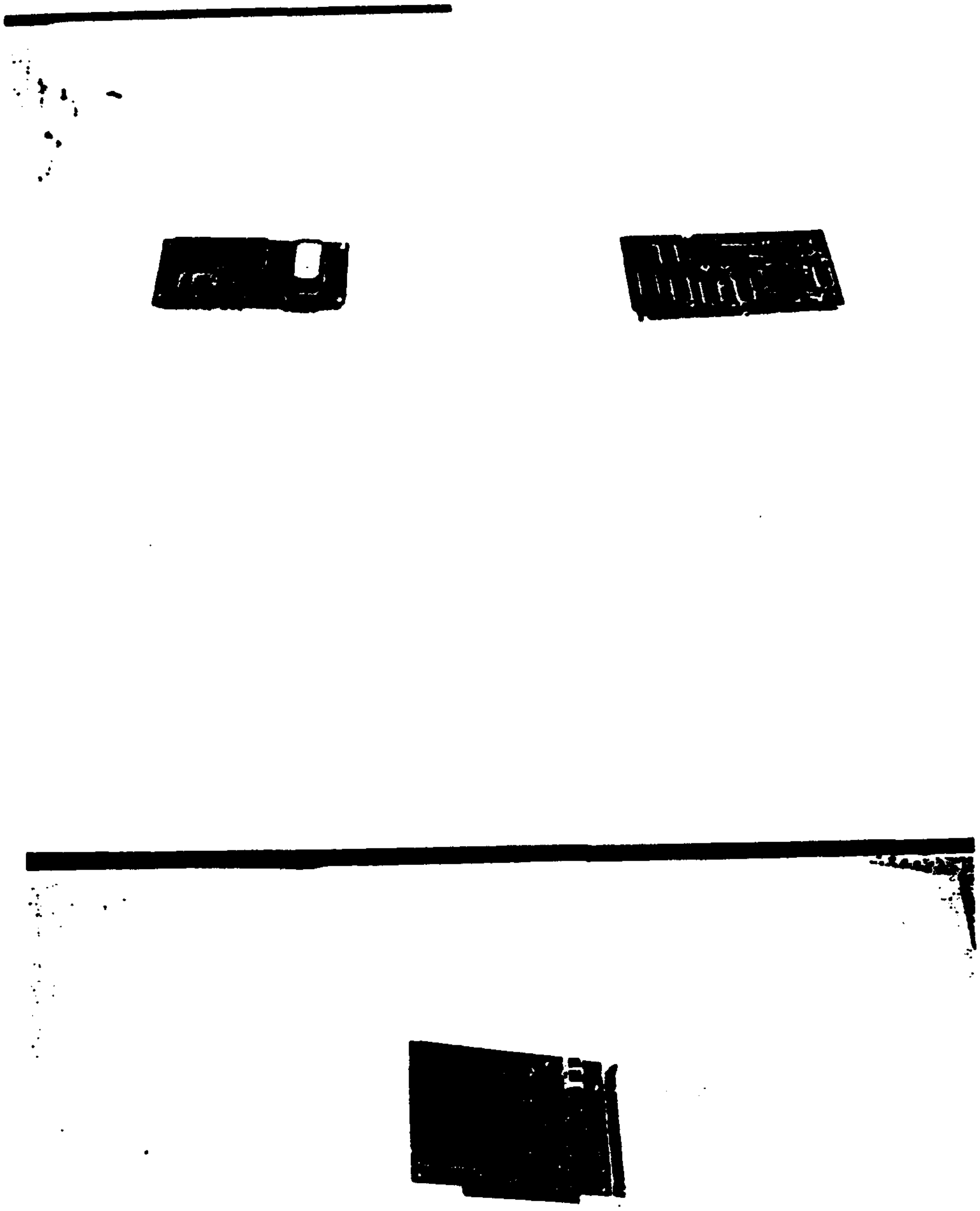


Figure 1.15 Photographs of a wild-card 88 (top) and a hard disk emulator (bottom).

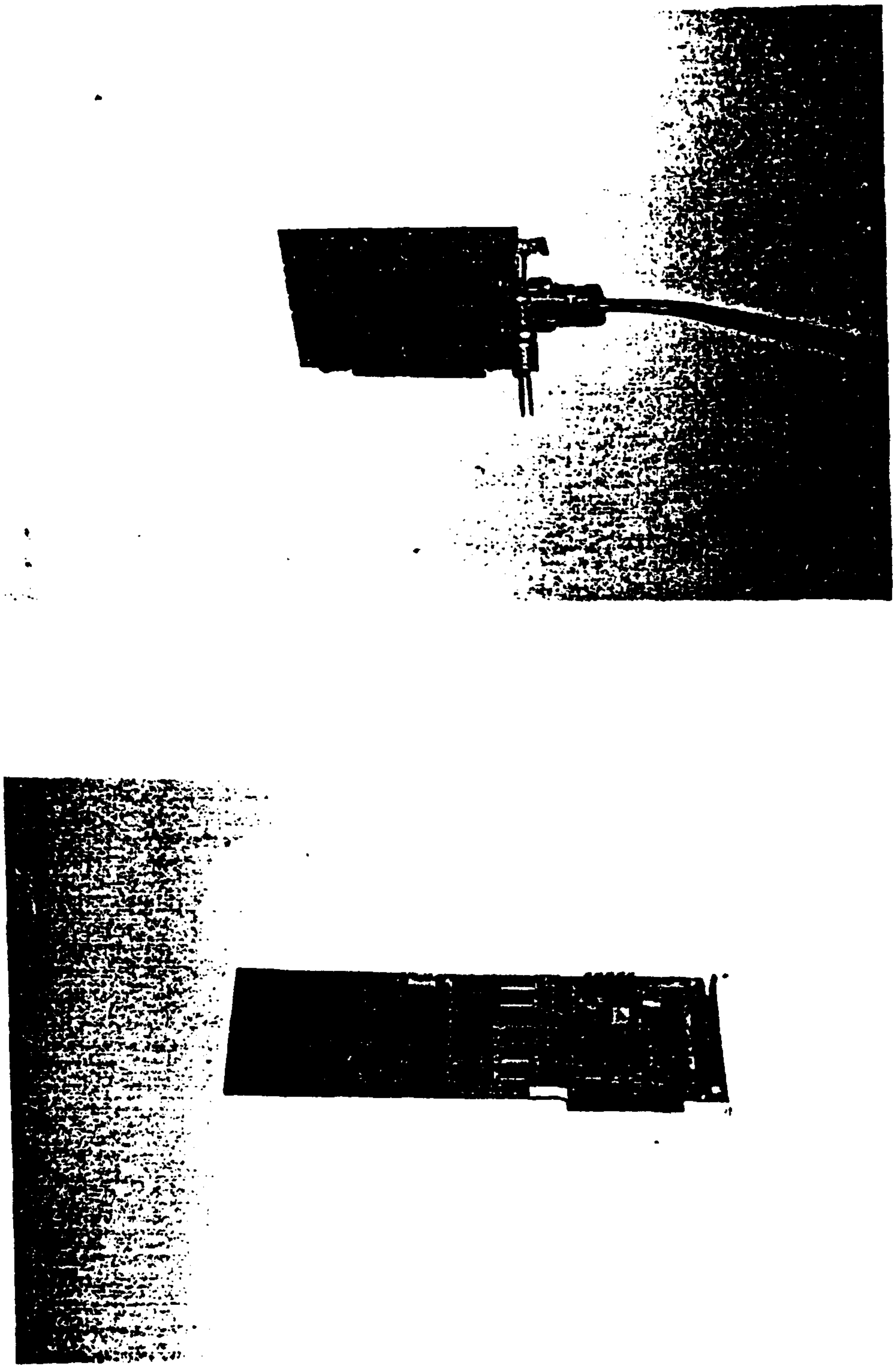


Figure 1.16 Photographs of a serial communication card (top) and an A/D I/O card (bottom).

The A/D card has two sets of connectors on the board. The set of connectors on the right side correspond to analog channels. Another set of connectors located between the middle and the left of the board carries the digital Input and Output signals. The top connector in this set corresponds to the 16-bit digital output. The current design does not make use of the 16-bit digital input made available by the board.

The photographs shown in Figure 1.17 are the same as Figure 1.14 with the three PC compatible boards shown in place. This makes up the total system with the exception of the external cables used for hooking up the over all system.

The photographs of a remote show four round connectors on the left side of the box, two female receptacles on the right and two male receptacles on the left. The left most connector is not used presently. The right male connector is used for connecting RS232 cable. The 16 digital outputs (16-bit control word) are interfaced with the outside world via the two female connectors. The left female connector carries the least significant byte of the control word and the right most connector delivers the most significant byte. The relationship of the control_word bits to the physical pins is shown in Figure 1.18. Also, shown is the designation of connectors according to their function, and the relationship of the RS232 connector pins to that of a standard RS232 9-pin connector.

It should be noted that all the 16 digital outputs are not totally consumed by a single radar. In fact only a small portion of the control_word is required to control each radar. Hence, to make the two remote units universal, the first eight digital outputs are reserved for remote #1's radar (C-Band/X-Band) and the last eight for remote #2's radar (VSG).

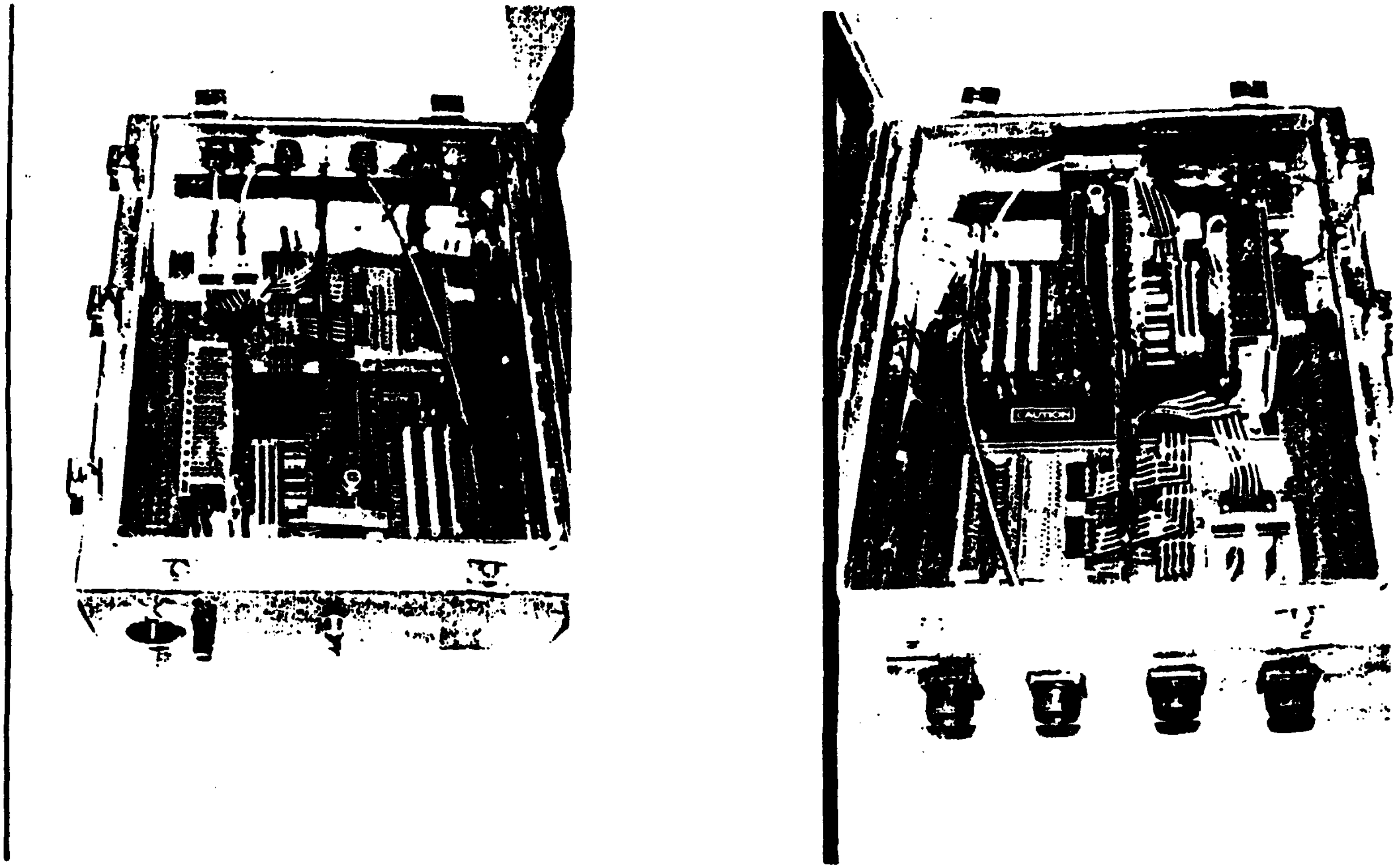
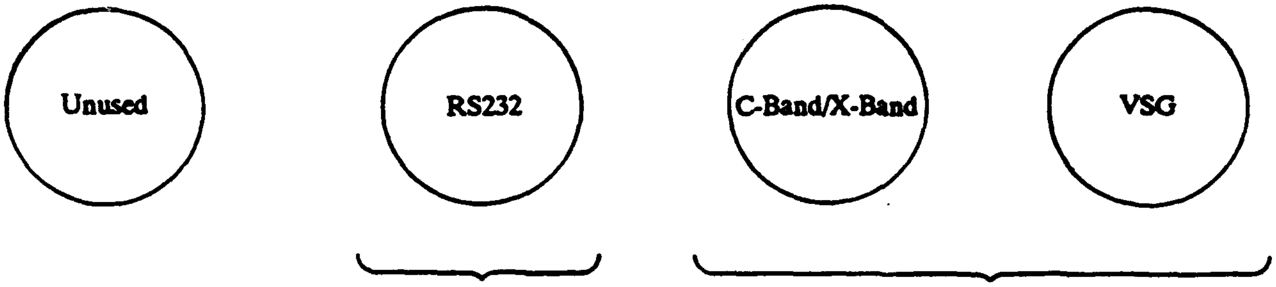


Figure 1.17 Photographs of a remote unit with the three PC compatible cards plugged into the wild-kit board.



Round Connector Pin#	RS232 9-pin Connector Pin#
A	1
B	2
C	3
D	4
E	5
F	6
G	7
H	8
I	9

Left Connector		Right Connector	
Bit #	Pin #	Bit #	Pin #
0	A	8	A
1	B	9	B
2	C	10	C
3	D	11	D
4	E	12	E
5	F	13	F
6	G	14	G
7	H	15	H
	I (GND)		I (GND)

Figure 1.18 Designation of the four round-connectors and their physical pins.

However, this division of the control_word is at the software level and can be changed if the need arises to increase the number of digital outputs for a radar.

The three PC compatible boards plugged in the wild-kit board have to be properly configured. The hardware configuration is done via jumpers and dip-switches provided on the boards. Figure 1.19 shows the jumper-assignments for the RS232 serial communication card. Some of the jumpers are to be left open. For example, w4 is shown to be completely open. Similarly, Figures 1.20 and 1.21 show the jumper-assignments and dip-switch settings for solid-state hard disk emulator and the A/D I/O card, respectively.

SOFTWARE NOTES:

The host unit and the two remotes communicate back and forth via different types of messages. A remote can send three different types of messages, **DATA_message**, **ERROR_message**, and **ACK_RESET_message**. Figure 1.22 shows the format of a DATA_message. The first byte contains the code of the message type, and the second byte lists the size of the message in number of bytes which is 25. Two data samples are paired in three consecutive bytes according to the format given in Figure 1.13. Therefore, the next 15 bytes are consumed by the data samples received from the 10 active analog channels. Next comes the 16-bit system_word, the most significant byte being in byte #17. The system_word consists of important information about the data_message and the state of the remote unit originating the data_message. Bits #19 through #22 contain the 32-bit sequence number which is actually the 32-bit time stamp made available by DOS. The most significant byte is located in byte #19. The last two bytes constitute the 16-bit checksum of

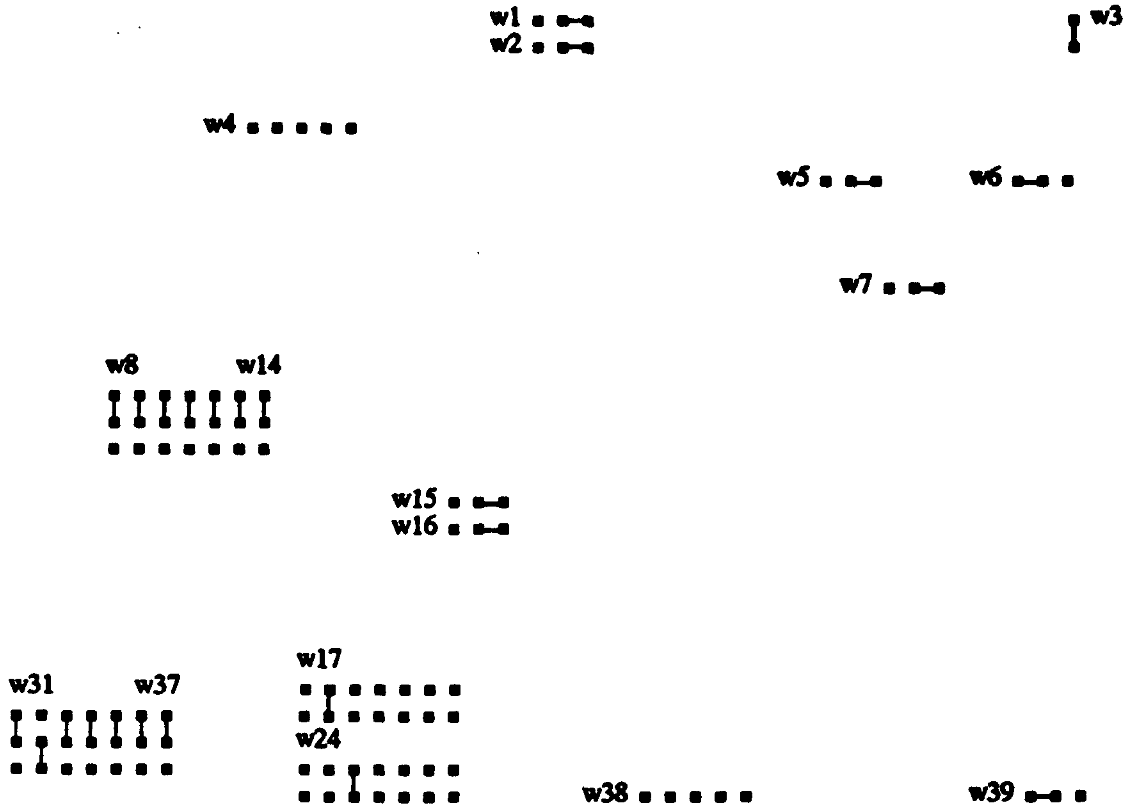


Figure 1.19 Jumper-assignment for the communication card.

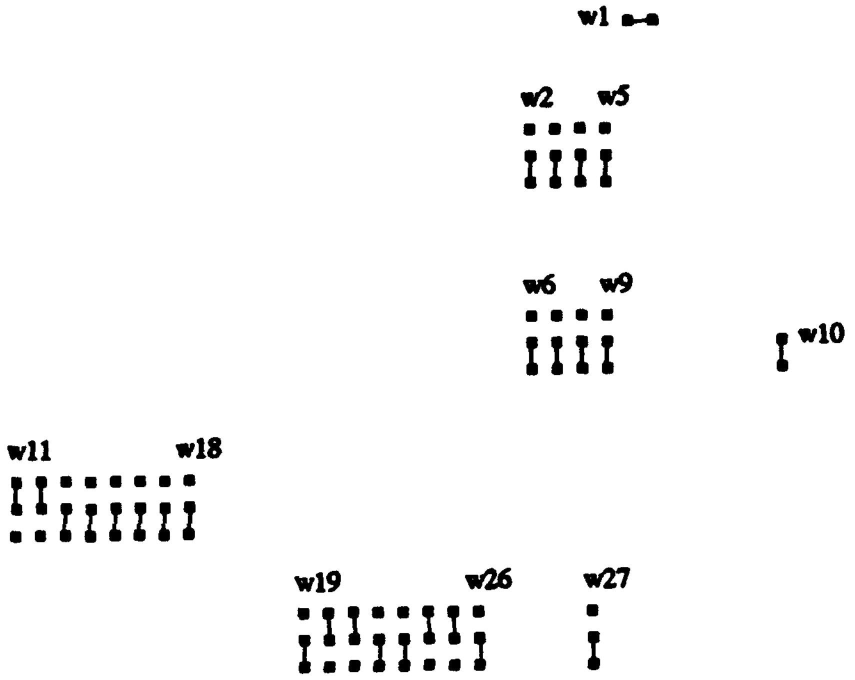


Figure 1.20 Jumper-assignment for the solid state hard disk card.

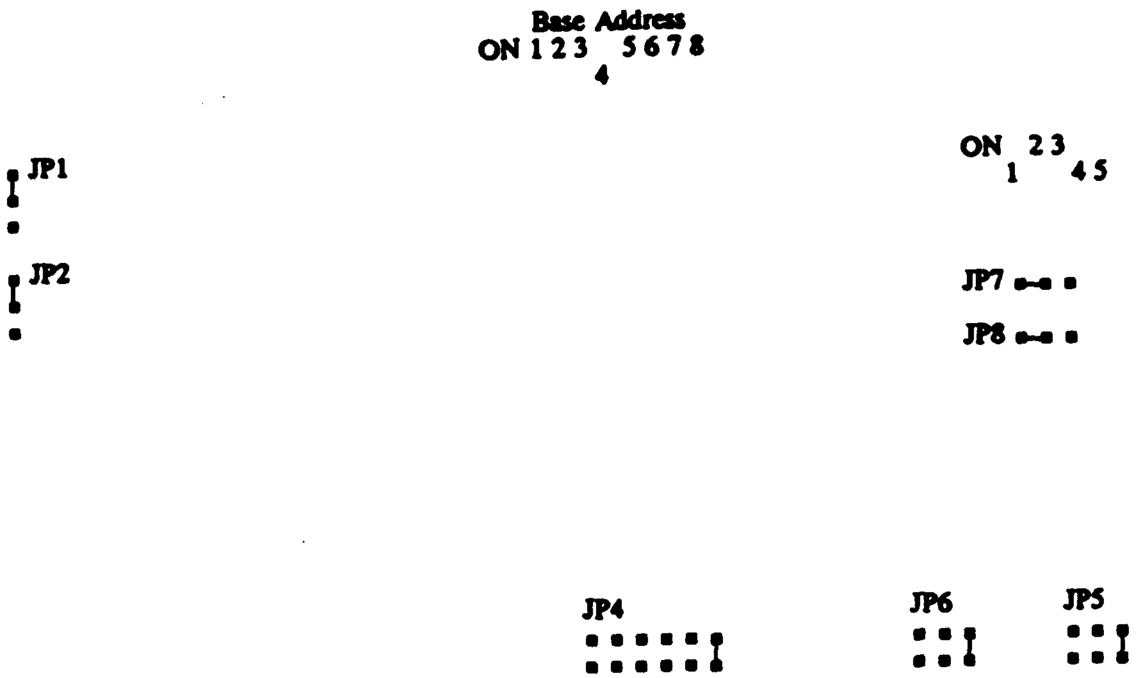


Figure 1.21 Jumper-assignment and Dip-switch settings for the A/D card.

Byte #		
0	DATA_Message	Message type
1	25	Size of message in bytes
2	Samples from channels 0 & 1	
3		
4		
5	Samples from channels 2 & 3	
6		
7		
8	Samples from channels 4 & 5	
9		
10		
11	Samples from channels 6 & 7	
12		
13		
14	Samples from channels 8 & 9	
15		
16		
17	System_Word	Flags reporting the state of Remote Machine
18		
19	Sequence_Number	32-bit Time Stamp
20		
21		
22		
23	Checksum	Sum of "0" to "22" bytes
24		

Figure 1.22 Message format of "DATA_Message" sent by a remote to the host machine.

the message, where the least significant byte is the very last one. The checksum is computed by simply adding bytes #0 through #22.

Figure 1.23 shows the format of the system_word used for relaying important information about the data_message and state of the corresponding remote. Note that the format is good for the system_words of both the host and a remote. The first four bits tell the previous and the present states. $\text{Bit}_{4:6}$ code the type of the message received last time. For example, if the system_word belongs to a remote, the last message received may be of type ACK_DATA informing the host that it received the acknowledgment of data sent to it previously. Or if a remote is in the OFF state and host sends it an ON_message, the remote would set $\text{Bit}_{4:6}$ of the system_word to 011 when sending the first data_message in order to inform the host that it received the ON_message. Similarly, if the system_word belongs locally to the host it sets $\text{Bit}_{4:6}$ accordingly.

Errors may occur while communicating back and forth. For example, if a remote is waiting for an acknowledgement of the data_message that it sent to the host earlier, and the time exceeds the sampling period, it quits waiting and sends the next data_message informing the host of the error situation. Similarly, a remote may receive a message with illegal type or with illegal checksum. These conditions can also be relayed to the host in the system_word of the next message that the remote is going to send to the host. In the system_word of every message that a remote sends to the host, the remote tells the host what it expects as the next message. Also the last three bits inform the host about which VSG_antenna is switched-on currently. Since the system_word is tagged to a data_message, the host knows to which antenna do the data samples correspond.

Bit #		
0	HMERROR/RMERROR	(Present State)
1	HMOFF/RMOFF	(Present State)
2	HMERROR/RMERROR	(Previous State)
3	HMOFF/RMOFF	(Previous State)

4
5
6

Bit$_6$	Bit$_5$	Bit$_4$	Received Message Type
0	0	0	—
0	0	1	ACK_DATA
0	1	0	NACK_DATA
0	1	1	ON
1	0	0	RESET
1	0	1	DATA
1	1	0	ACK_RESET
1	1	1	ERROR

7	Time exceeded error
8	Checksum error
9	illegal size of message
10	Host expecting DATA message or remote expecting ACK_DATA message
11	Remote expecting ON message
12	Host expecting ACK_RESET message or remote expecting RESET message
13	VSG_Antenna_A is on
14	VSG_Antenna_B is on
15	VSG_Antenna_C is on

Figure 1.23 Format of the Host/Remote System_Word.

In addition to the data_message type, a remote can send two other messages to the host. One is the error_message and the other is the ack_reset_message. Figure 1.24 shows the format of each type of message. The error_message is sent to the host if multiple communication or protocol errors occur consecutively. This usually happens when a remote is sending data_messages to the host but is not receiving acknowledgements or an alternate message (RESET or ON) from the host. Ack_reset_message is sent to the host in response to a RESET message.

There are four messages that the host may send to a remote, **ACK_DATA_message, NACK_DATA_message, ON_message,** and **RESET_message.** The formats of these messages is given in Figure 1.25. When the operator at the host is not running a session, the remotes are in their OFF states. Upon initiating a run-session, the host sends an ON_message to each active remote. A remote is released from its OFF state when it receives an ON_message, and it starts the process of sampling and sending the data samples to the host in the data_message format. If the host receives a data_message without any errors, it sends an ACK_RESET_message to the respective remote otherwise a NACK_DATA_message is sent. If more than a certain number of errors occurs the host goes through a synchronization procedure which consists of sending a RESET_message to each of the active remotes. This forces the remotes to go into their OFF states and wait for an ON_message from the host. When the host receives ACK_RESET_message from each of the active remotes, it knows that both remotes are synchronized and are idling in their OFF states. The host then sends an ON_message to each one thereby restarting the process.

Byte #		
0	ERROR_Message	Message type
1	10	Size of message in bytes
2	System_Word	Flags reporting the state of Remote Machine
3		
4	Sequence_Number	32-bit Time Stamp
5		
6		
7		
8	Checksum	Sum of "0" to "7" bytes
9		

Byte #		
0	ACK_RESET_Message	Message type
1	10	Size of message in bytes
2	System_Word	Flags reporting the state of Remote Machine
3		
4	Sequence_Number	32-bit Time Stamp
5		
6		
7		
8	Checksum	Sum of "0" to "7" bytes
9		

Figure 1.24 Message format of "ERROR_Message" and "ACK_RESET_Message" sent by a Remote Machine to the Host Machine.

Byte #	
0	ON_Message
1	10
2	Control_Word
3	
4	Sequence_Number
5	
6	
7	
8	Checksum
9	

Byte #		
0	RESET_Message	Message type
1	10	Size of message in bytes
2	Control_Word	Command word to control Remote Machine and Radar
3		
4	Sequence_Number	32-bit Time Stamp
5		
6		
7		
8	Checksum	Sum of "0" to "7" bytes
9		

Byte #	
0	ACK_DATA_Message
1	10
2	Control_Word
3	
4	Sequence_Number
5	
6	
7	
8	Checksum
9	

Byte #		
0	NACK_DATA_Message	Message type
1	10	Size of message in bytes
2	Control_Word	Command word to control Remote Machine and Radar
3		
4	Sequence_Number	32-bit Time Stamp
5		
6		
7		
8	Checksum	Sum of "0" to "7" bytes
9		

Figure 1.25 Message format of "ON_Message", "RESET_Message", "ACK_DATA_Message", and "NACK_DATA_Message" sent by the Host Machine to a Remote Machine.

In the messages sent by the host, a 16-bit control_word is also sent. In any format of a message shown in Figure 1.25, the location of the control_word is same as that of the system_word in a message sent by a remote. The most significant byte of the control_word is in byte #2. The format of the control_word was given earlier in Figure 1.5.

The functioning of the host-software and the remote-software will now be detailed with the help of flow-charts. In case of the host, there are four software units that have the crux of the protocol for its link to the two remote units. The first is the executive unit which runs the other three units and is called **HM_EXECUTIVE** (HM_EXEC). Its flow-chart is shown in Figure 1.26 which also shows the flow-chart of the USER_INTERFACE unit from where the HM_EXEC unit is invoked. The user first sets some of the relevant parameters for a run and then initiates the data collection process. This invokes HM_EXEC unit where first the initialization of the file system and some parameters takes place. Max_Exec_Time is set to the sampling period. This is the time limit which when exceeded causes an error condition. This may happen when the host does not receive a message from a remote within the expected time. It is important to note that in every sampling period the data must be collected, displayed and stored. To adhere to the sampling rate of 30 HZ, these tasks must be accomplished comfortably within the sampling period of 33 milliseconds.

After initialization process, **HM_OFF** unit is invoked where an ON_message is sent to each of the active remotes. If the total ERROR_COUNT has not exceeded the assigned limit, **HM_RCV_BLK** is accessed where the host receives data_messages sent by the remotes. These data messages are decoded to extract data samples which are then displayed and stored. The host sends acknowledgement messages back to the remotes. If errors are

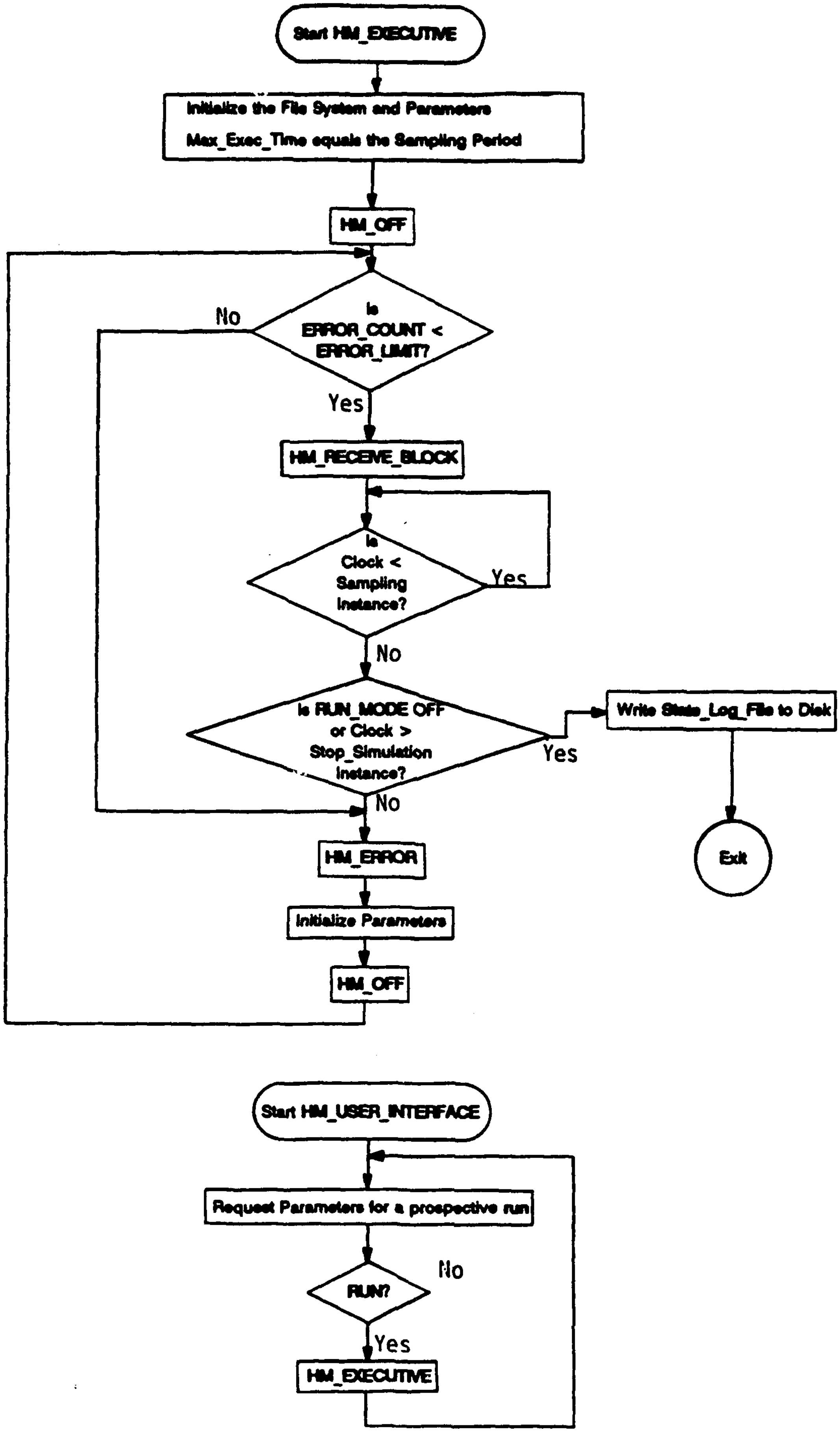

Start HM_EXECUTIVE
Initialize the File System and Parameters
Max_Exec_Time equals the Sampling Period
HM_OFF
Is ERROR_COUNT < ERROR_LIMIT?
No
Yes
HM_RECEIVE_BLOCK
Is Clock < Sampling Instance?
Yes
No
Is RUN_MODE OFF or Clock > Stop_Simulation Instance?
Yes
Write State_Log_File to Disk
No
Exit
HM_ERROR
Initialize Parameters
HM_OFF
Start HM_USER_INTERFACE
Request Parameters for a prospective run
RUN?
No
Yes
HM_EXECUTIVE

Figure 1.26 Flow chart of "EXECUTIVE" procedure of the Host Machine, and Flow chart of "USER_INTERFACE" of the Host Machine.

encountered in this procedure, ERROR_COUNT is updated. When the process gets back to the HM_EXEC unit after exiting from HM_RCV_BLK unit, the host waits until the current sampling period has expired. If RUN_MODE is still on and the run-time has not expired, the process is repeated. RUN_MODE is a flag which is TRUE initially, and may be set to FALSE if the session needs to be terminated. In case the run-time has expired or RUN_MODE is FALSE, STATE_LOG_FILE is written to the disk and process exits HM_EXEC unit. The STATE_LOG_FILE consists of the record of up to 256 most recent states/units visited by the host unit. This helps in debugging fault conditions.

If the total ERROR_COUNT exceeds its assigned limit, the host executes HM_ERROR unit. Here, a RESET_message is sent to each remote for synchronization purposes. This unit is exited when the host receives the respective ACK_RESET_messages. Then a partial initialization of the host unit takes place followed by execution of HM_OFF unit where an ON_message is sent to each remote unit to restart the sampling process. The host then returns to its normal task of data collection. The flow-charts of HM_ERROR and HM_OFF units are given in Figures 1.27 and 1.28, respectively. Note that upon entering and exiting a unit, the system_word and the STATE_LOG record are updated.

Figure 1.29 shows the flow-chart of HM_RCV_BLK unit. After updating system_word and STATE_LOG record, A/D card is requested to sample its channel #0 to which is hooked the wave gauge equipment. When the sample arrives, its integrity is checked and the data_status_word is modified accordingly. The host then receives data_messages from each of the active remotes. In each case, the samples are checked and the data_status_word is updated. If an error occurs the appropriate ERROR_COUNT (corresponding to remote

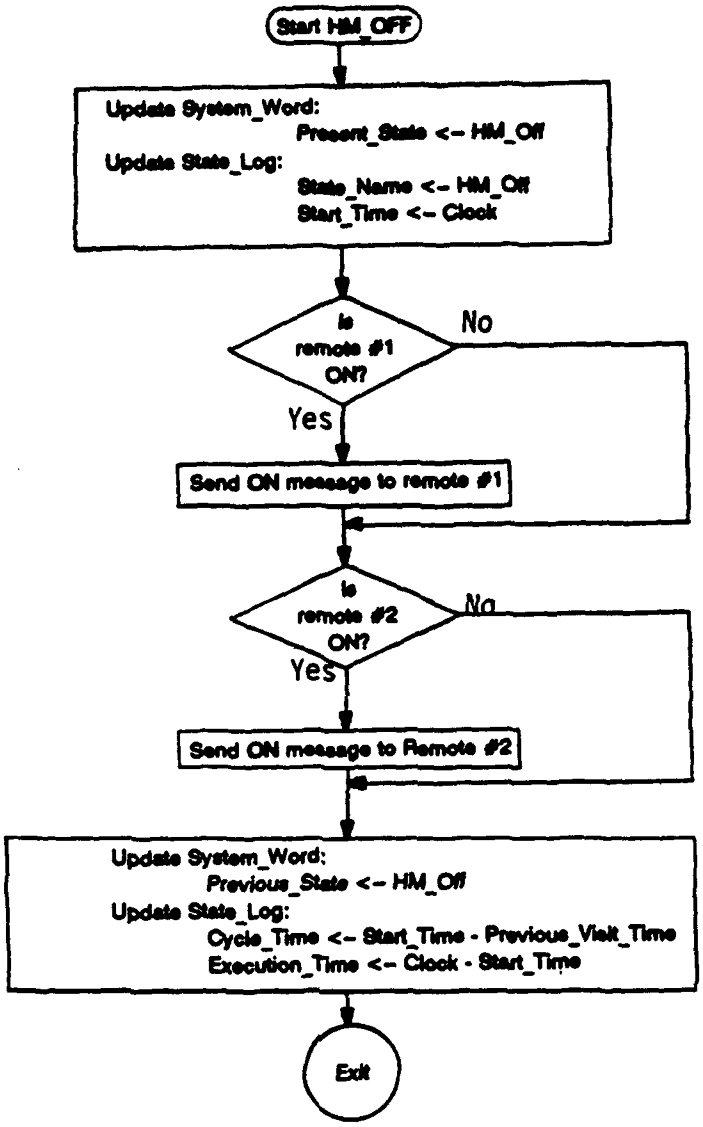


Figure 1.27 Flow Chart of "OFF" procedure of the Host Machine.

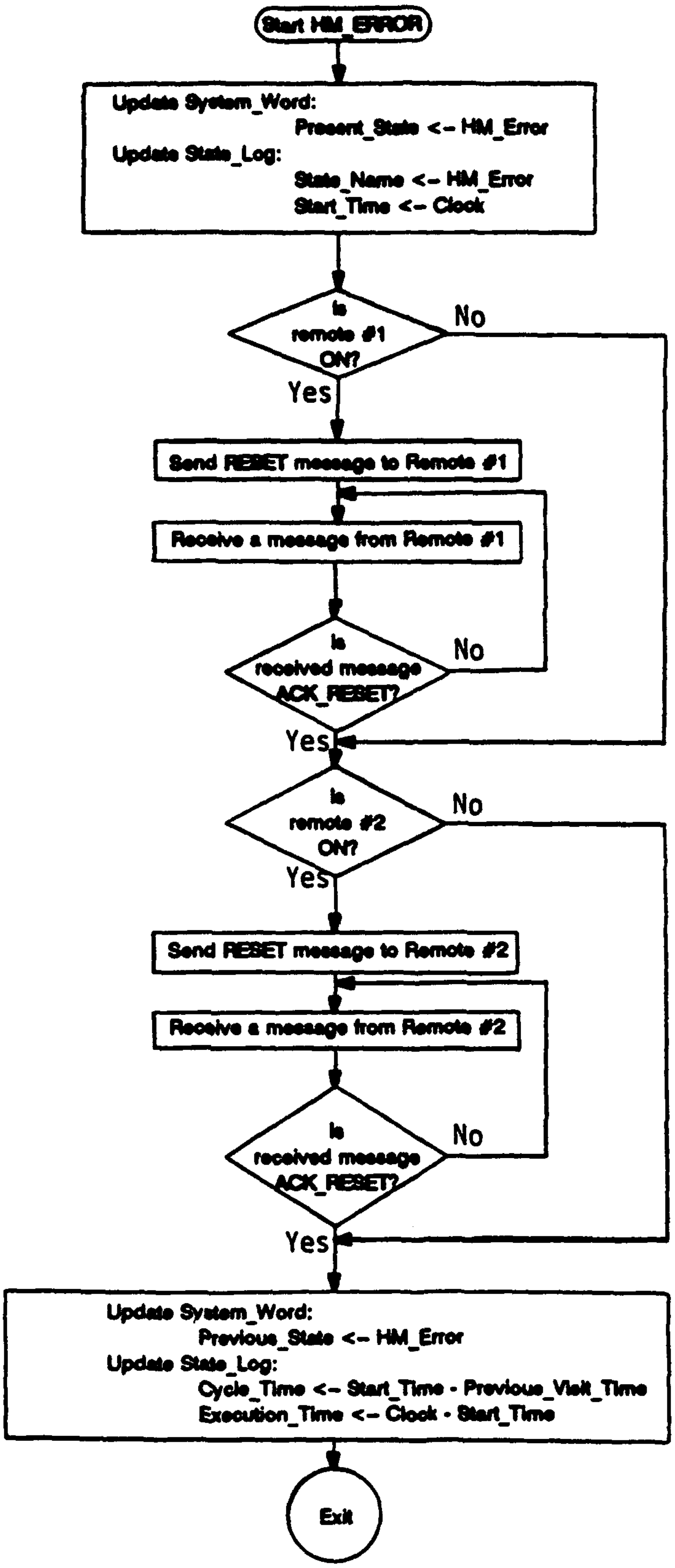

Start HM_ERROR
Update System_Word:
Present_State <-- HM_Error
Update State_Log:
State_Name <-- HM_Error
Start_Time <-- Clock
Is remote #1 ON?
No
Yes
Send RESET message to Remote #1
Receive a message from Remote #1
Is received message ACK_RESET?
No
Yes
Is remote #2 ON?
No
Yes
Send RESET message to Remote #2
Receive a message from Remote #2
Is received message ACK_RESET?
No
Yes
Update System_Word:
Previous_State <-- HM_Error
Update State_Log:
Cycle_Time <-- Start_Time - Previous_Visit_Time
Execution_Time <-- Clock - Start_Time
Exit

Figure 1.28 Flow Chart of "ERROR" procedure of the Host Machine.

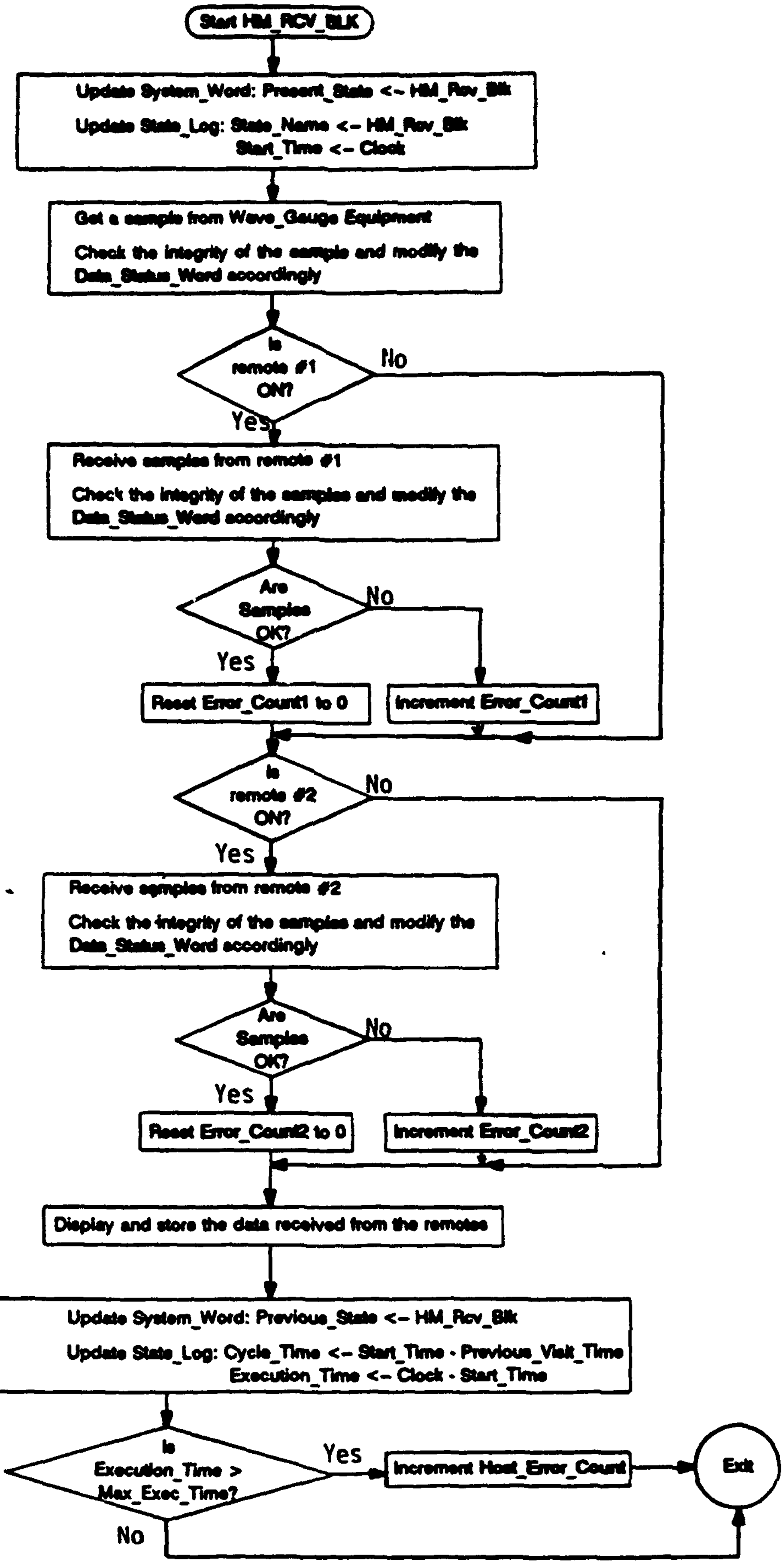

Start HM_RCV_BLK
Update System_Word: Present_State <-- HM_Rcv_Blk
Update State_Log: State_Name <-- HM_Rcv_Blk
Start_Time <-- Clock
Get a sample from Wave_Gauge Equipment
Check the integrity of the sample and modify the Data_Status_Word accordingly
Is remote #1 ON?
No
Yes
Receive samples from remote #1
Check the integrity of the samples and modify the Data_Status_Word accordingly
Are Samples OK?
No
Yes
Reset Error_Count1 to 0
Increment Error_Count1
Is remote #2 ON?
No
Yes
Receive samples from remote #2
Check the integrity of the samples and modify the Data_Status_Word accordingly
Are Samples OK?
No
Yes
Reset Error_Count2 to 0
Increment Error_Count2
Display and store the data received from the remotes
Update System_Word: Previous_State <-- HM_Rcv_Blk
Update State_Log: Cycle_Time <-- Start_Time - Previous_Visit_Time
Execution_Time <-- Clock - Start_Time
Is Execution_Time > Max_Exec_Time?
Yes
Increment Host_Error_Count
Exit
No

Figure 1.29 Flow Chart of "RECEIVE_BLOCK" procedure of the Host Machine.

#1 or remote #2) is incremented. After this the received data samples are displayed. At the end, the system_word and the STATE_LOG record are updated, and the execution time of the state is checked to see if the sampling period time has been exceeded. If so, ERROR_COUNT is incremented.

The software at a remote unit has complementary characteristics. Four software units are implemented, RM_ERROR, RM_OFF, RM_XMT_BLK, and RM_EXEC. The executive unit is RM_EXEC and is used to drive the other three units. The flow-chart of RM_EXEC is given in Figure 1.30. As in the case of the host unit, the executive unit starts by initialization of the file system and some important parameters. Maximum execution time that the remote can spend in waiting for an acknowledgement of a data_message from the host is set to the sampling period.

After the initialization step, RM_OFF is invoked where a message from the host is awaited. ERROR_COUNT is reset upon entering the loop for collecting and sending data samples. The loop continues as long as ERROR_COUNT is less than its assigned limit and conditions have not developed which make it necessary to go to the ERROR_state or OFF_State. RM_XMT_BLK is the unit where data samples are collected, packed into a data_message, and sent to the host. When this loop is exited, either RM_ERROR needs to be accessed or RM_OFF has to be invoked. A remote calls upon RM_ERROR unit only if it gets a RESET_message from the host. In RM_ERROR unit, a remote sends ACK_RESET_message to the host and exits. It is important to initialize certain parameters after visiting RM_ERROR unit. The remote then invokes RM_OFF unit for synchronization with the host.

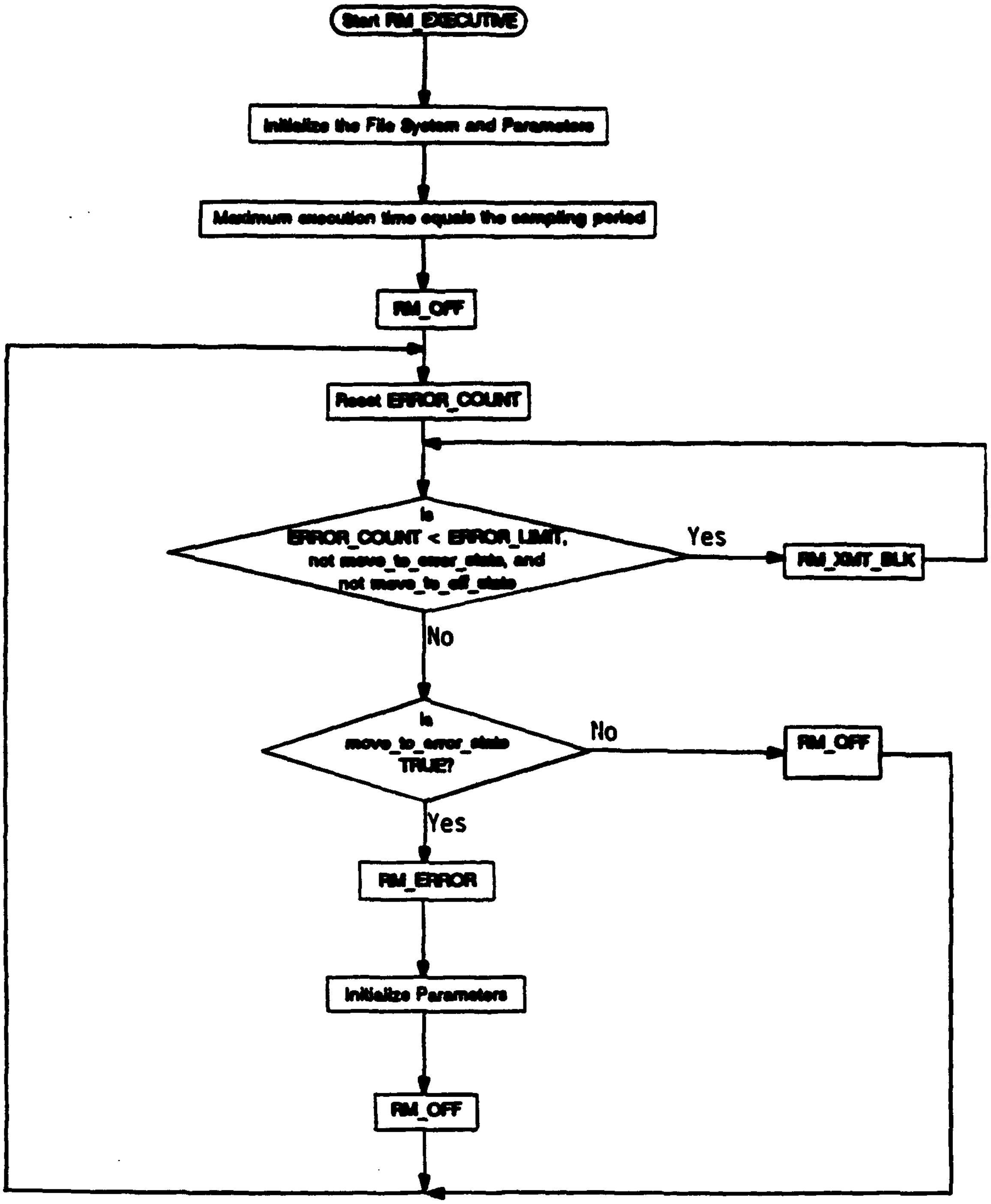

Start RM_EXECUTIVE
Initialize the File System and Parameters
Maximum execution time equals the sampling period
RM_OFF
Reset ERROR_COUNT
Is ERROR_COUNT < ERROR_LIMIT, not move_to_error_state, and not move_to_off_state
Yes
RM_XMT_BLK
No
Is move_to_error_state TRUE?
No
RM_OFF
Yes
RM_ERROR
Initialize Parameters
RM_OFF

Figure 1.30 Flow chart of "EXECUTIVE" procedure of a Remote Machine.

During execution move_to_error_state and move_to_off_state flags may be set to FALSE or TRUE. These flags tell the executive unit to either visit RM_ERROR unit or RM_OFF unit, respectively. Figure 1.31 shows the flow-chart of RM_OFF unit which as all other units updates the system_word and the STATE_LOG record upon entry and exit. Since this unit is being visited, move_to_off_state flag is set to FALSE in order to avoid unnecessary revisits. The attached radar is set according to the contents of the most recent control_word, and digital output is generated in order to set VSG_antenna_A. As explained earlier, no matter which remote it is, the signals for switching the VSG_antennas are always produced to make the two remotes interchangeable.

At this point the remote waits for a message from the host. The received message may be an ON_message, a RESET_message, or a late arriving ACK_DATA_message. If it is a RESET_message, move_to_error_state flag is set to TRUE and the unit is exited. Otherwise, move_to_error_state flag is set to FALSE, the local clock is synchronized with that of the host's (received as a time stamp as part of the message), and data_samples are collected corresponding to VSG_antenna_A. The radar is set according to the newly received control_word, VSG_antenna_B is set and the unit is exited. It is recalled that the data collection process is not affected if the attached radar is not using the VSG_antenna switching signals.

Figures 1.32 and 1.33 show the flow-charts of RM_ERROR and RM_XMT_BLK units, respectively. RM_ERROR unit is only called upon when the host sends a RESET_message. This software unit sends ACK_RESET_message telling the host that the remote unit is idle and waiting for an ON_message for a synchronized start of the data collection process. The

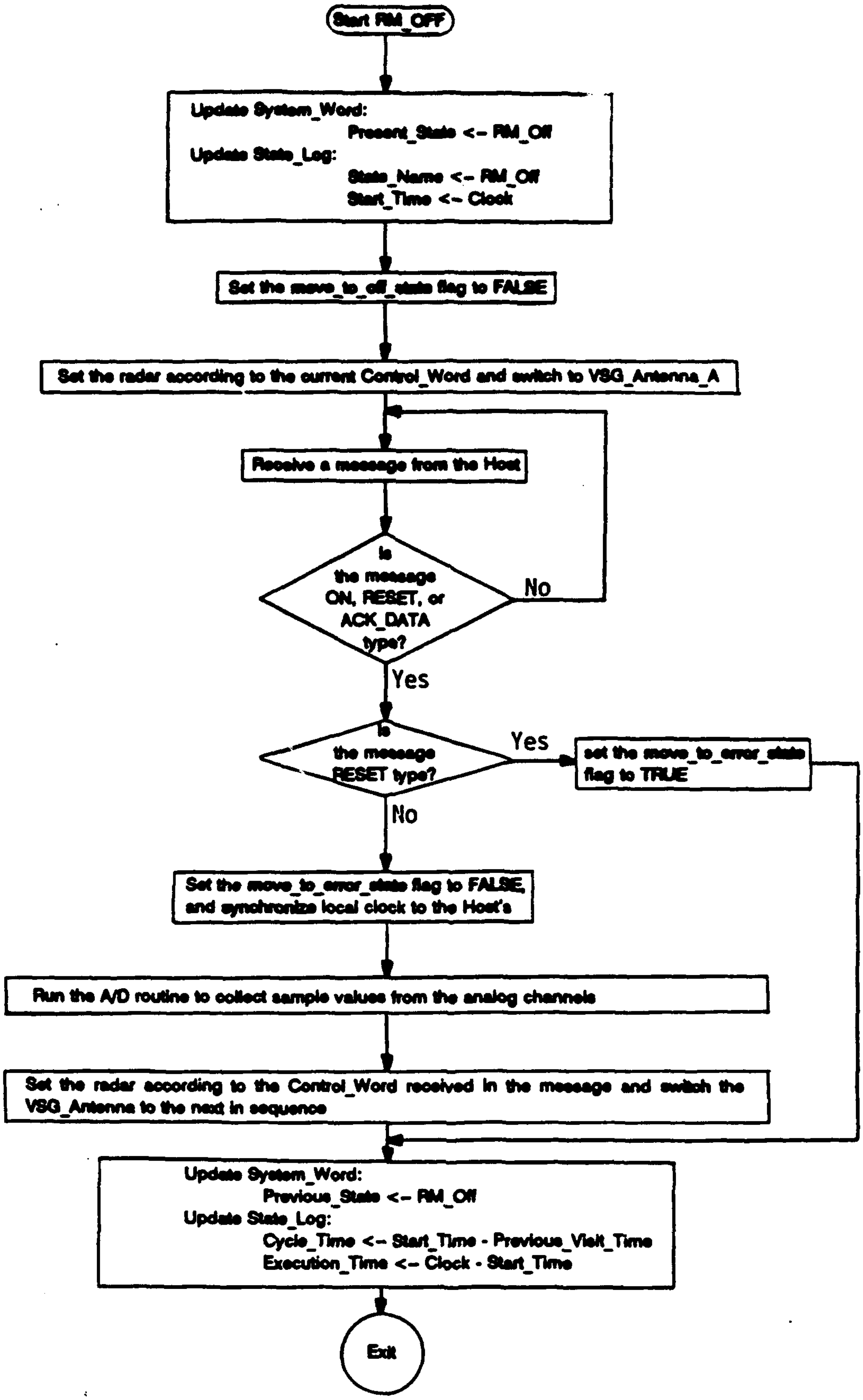

Start RM_OFF
Update System_Word:
Present_State <-- RM_Off
Update State_Log:
State_Name <-- RM_Off
Start_Time <-- Clock
Set the move_to_off_state flag to FALSE
Set the radar according to the current Control_Word and switch to VSG_Antenna_A
Receive a message from the Host
Is the message ON, RESET, or ACK_DATA type?
No
Yes
Is the message RESET type?
Yes
set the move_to_error_state flag to TRUE
No
Set the move_to_error_state flag to FALSE, and synchronize local clock to the Host's
Run the A/D routine to collect sample values from the analog channels
Set the radar according to the Control_Word received in the message and switch the VSG_Antenna to the next in sequence
Update System_Word:
Previous_State <-- RM_Off
Update State_Log:
Cycle_Time <-- Start_Time - Previous_Visit_Time
Execution_Time <-- Clock - Start_Time
Exit

Figure 1.31 Flow Chart of "OFF" procedure of a Remote Machine.

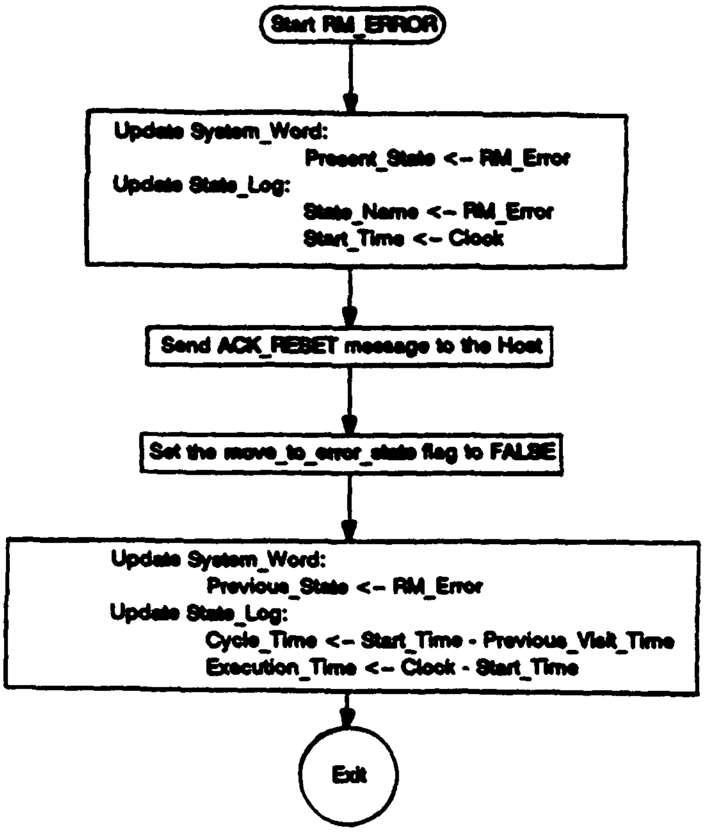

Start RM_ERROR
Update System_Word:
Present_State <-- RM_Error
Update State_Log:
State_Name <-- RM_Error
Start_Time <-- Clock
Send ACK_RESET message to the Host
Set the move_to_error_state flag to FALSE
Update System_Word:
Previous_State <-- RM_Error
Update State_Log:
Cycle_Time <-- Start_Time - Previous_Visit_Time
Execution_Time <-- Clock - Start_Time
Exit

Figure 1.32 Flow Chart of "ERROR" procedure of a Remote Machine.

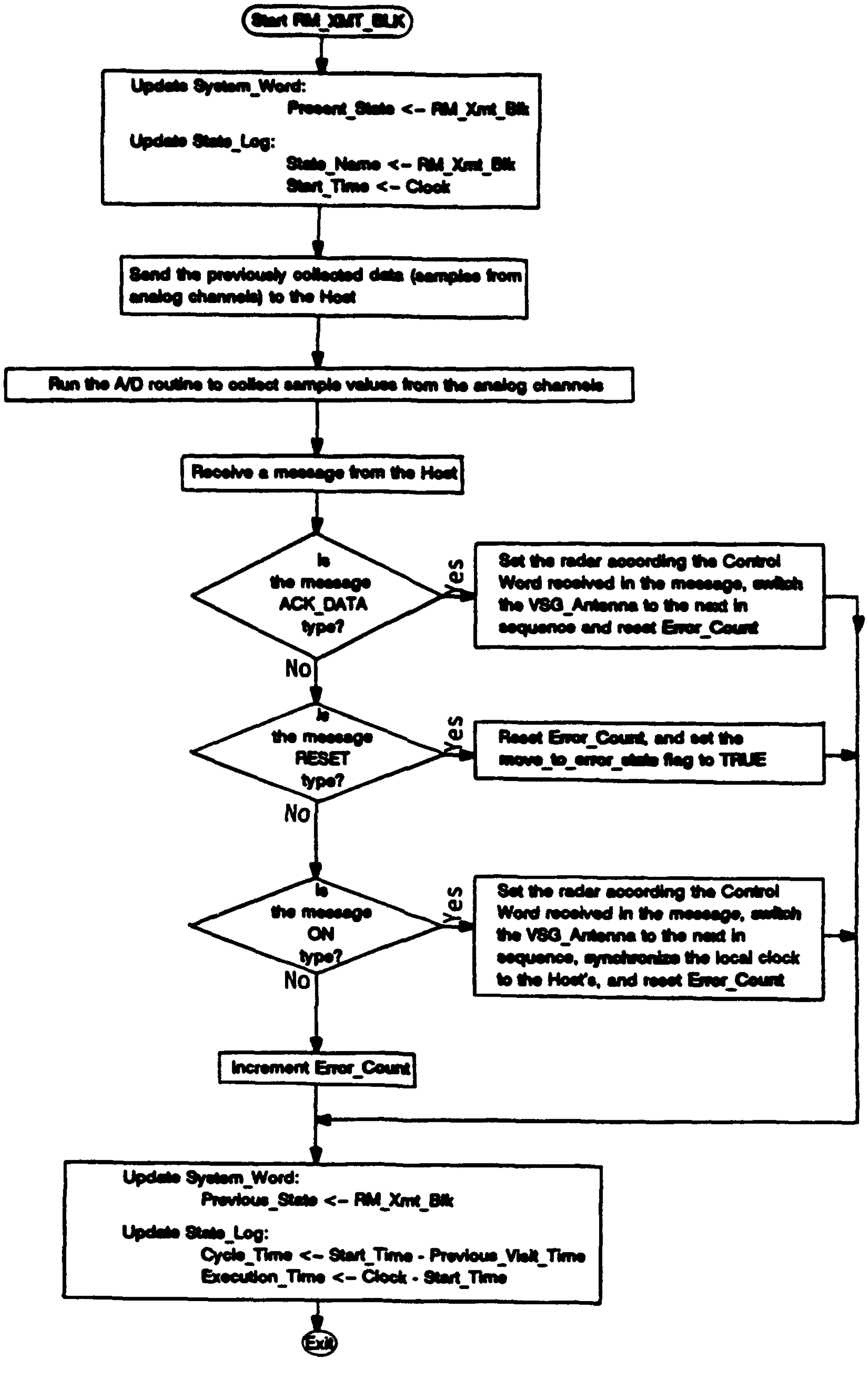

Start RM_XMIT_BLK
Update System_Word:
Present_State <-- RM_Xmt_Blk
Update State_Log:
State_Name <-- RM_Xmt_Blk
Start_Time <-- Clock
Send the previously collected data (samples from analog channels) to the Host
Run the A/D routine to collect sample values from the analog channels
Receive a message from the Host
Is the message ACK_DATA type?
Yes
Set the radar according the Control Word received in the message, switch the VSG_Antenna to the next in sequence and reset Error_Count
No
Is the message RESET type?
Yes
Reset Error_Count, and set the move_to_error_state flag to TRUE
No
Is the message ON type?
Yes
Set the radar according the Control Word received in the message, switch the VSG_Antenna to the next in sequence, synchronize the local clock to the Host's, and reset Error_Count
No
Increment Error_Count
Update System_Word:
Previous_State <-- RM_Xmt_Blk
Update State_Log:
Cycle_Time <-- Start_Time - Previous_Visit_Time
Execution_Time <-- Clock - Start_Time
Exit

Figure 1.33 Flow Chart of "TRANSMIT_BLOCK" procedure of a Remote Machine.

data samples are transmitted as a data_message (block) by RM_XMT_BLK unit. Upon entry it updates the system_word and the STATE_LOG record, and then it transmits the previously collected samples. These data samples would correspond to VSG_antenna_A if it is the first visit to RM_XMT_BLK after exiting RM_OFF. While the host would be preparing the ACK_DATA_message, the remote collects the next batch of samples from its analog channels corresponding to the current settings of the control_word.

The host then waits for a message from the remote. Different responses are generated depending upon the type of message received. If it is an ACK_DATA_message, the radar is set according to the newly received control_word, control signals corresponding to the next in sequence VSG_antenna are produced, ERROR_COUNT is reset, and the unit is exited. If the received message is a RESET_message, ERROR_COUNT is reset, move_to_error_state flag is set to TRUE so that the remote goes into RM_ERROR unit after its exit from RM_XMT_BLK. If the received message is an ON_message, everything that is done in response to it in the RM_OFF unit is repeated here. That is to say, the radar is set according to the new control_word, VSG_antenna is cycled, local clock is synchronized to the host's clock, and ERROR_COUNT is reset. In case of receiving an illegal message, ERROR_COUNT is incremented.

It is important to note that the A/D routine to collect data samples is not run immediately after setting the radar (according to the new control_word) and generating control signals for a VSG_antenna. The wait between the two processes is the time taken to send the previously collected samples, i.e. length of one data_message. A data_message is 25 bytes long. Considering 9 bits for every byte of data (8 bits of data and 1 bit for the stop-bit) at 38400 bits per second, the wait comes out to be nearly 6 milliseconds (9*25/38400) which is enough for the radar circuitry to get stable at the new settings.

DEBUGGING NOTES:

If the data acquisition system is hooked-up according to the instructions given earlier and hardware problems such as broken cables/connectors, loose boards etc. are not present and still the system is not functioning properly then the host's STATE_LOG file must be examined. This file keeps track of the most recent 255 software state visits by the host. If it is seen that the host is trying to send RESET_messages and ON_messages to the remotes with no response, then the remotes have not been properly booted-up. This may be due to power problems or that the software loaded on the solid state hard disk emulator of a remote has been erased because of a bad battery on its board. Even if the battery is good, the battery's electrical connection to the board may have been intermittently disrupted due to a strong mechanical shock.

In order to check the integrity of the solid state hard disk emulator card, it must be taken out of a remote and plugged into an expansion slot of a PC. If the PC has two floppy drives (A and B) and no hard disk, this hard disk emulator card will appear as C drive. The contents of the hard disk can then be checked to see if all the files are present. Otherwise, the disk must be formatted and loaded with the required boot-up software (Appendix D).

It should be noted that a remote may take up to 5 minutes to boot-up once it is given power. For test purposes, a known DC supply may be connected to a specific channel and real-time display at the host can be examined for a correct value of the signal. Each remote must be made active one at a time while testing the system. The real-time display changes smoothly if there is no problem.

REFERENCES

[1] Intel Wild-Kit 88 and Wild-Card-88-10N.

[2] Intel P8256A8, 256 KByte SIMM for Wild-Kit 88, 120 ns.

[3] Model 6M20-R256, 256 KByte Solid State Hard Disk Emulator Card; Industrial Computer Source, Chicago, Illinois.

[4] Cyber Research PCAL-812 12-Bit A/D I/O Card Complete with Software Drivers, Cables, Screw Terminal Board; Cyber Research, Connecticut.

[5] RS232 Serial Communication Card; Industrial Computer Source, Chicago, Illinois.

[6] COMMBIOS/COMMBUFF, BIOS INT 14H Interface and Programming Tools, COMMTECH, Inc., KS

APPENDICES

Appendix A: Software units of the host machine

Appendix B: Software units of a remote machine

Appendix C: Data retrieval program

Appendix D: Boot-up disks for the host and remote machines

The above appendices are contained in a disk that is filed with the RSL file copy of this technical report (RSL TR 8621-1). This disk contains the files to run the SAXON Data Acquisition System. See this technical report for operating instructions about MAIN.EXE and VSGBIN.EXE. MAIN.EXE is located in directory Disk_D and VSGBIN.EXE is located in directory Disk_C.

To obtain a copy of this disk, communicate with:

Ms. Donnis Graham
Radar Systems and Remote Sensing Laboratory
2291 Irving Hill Road
Lawrence KS 66045-2969
TEL: 913/864-4835 * FAX: 913/864-7789
INTERNET: graham@ardneh.rsl.ukans.edu

PART B

SUPPLEMENT TO
A DATA ACQUISITION/CONTROLLER SYSTEM

SUPPLEMENT TO THE MANUAL "DATA ACQUISITION/CONTROLLER SYSTEM"

Marcelo Cavalcanti
Radar Systems and Remote Sensing Laboratory
Center for Research INC.
The University of Kansas

May 11, 1994

ACKNOWLEDGEMENTS:

The purpose of this report is to supplement the manual entitled "A Data Acquisition/Controller System". The mentioned manual and the described system was implemented by Fitratullah Khan, Charles Roberts, Asif Salam and Bita Sistani under the supervision of Dale Rummer.

The points explained herein in more detail are the work of Marcelo Cavalcanti. Several sources were consulted in order to have the report in hands now, they were: Fitratullah Khan, Bita Sistani, Dale Rummer, Wes Ellison, Matt Rose, Keng-Yin Chok, Samuel Haimov and Richard K. Moore.

INTRODUCTION:

The material covered in this supplement is what is believed was not explained in detail in the report. As we were trying to make the data acquisition system operationable again, we encountered several difficulties. This report's intention is that in the future as others decide to use and upgrade the system, these same obstacles will not be encountered.

There are two sections to this report. One which deals with the system setup, namely, the Hardware. The second part deals with the actual usage of the "Main" program and the acquisition, conversion and understanding of the data.

It is important to mention that at the time of this report, the data acquisition system had not yet been retested, since its original use, with the actual radar. Several independent tests with signal generators have been run.

All the documentation and a disk with the necessary files utilized to operate the Data Acquisition System can be found in a folder entitled "SAXON DAS MANUAL - May 1994" under the care of Samuel Haimov.

SOFTWARE SETUP:

The 3½ disk that can be found on the back of the DAS folder has all the necessary programs to run the data acquisition system. It is suggested to use a 386 or 486 computer. A Bernoulli drive is required. (The data files are stored there).

To install the necessary programs, create a directory in the C: drive and under this directory create 5 directories named diska, diskb, diskc, diskd_1, diskd_2 where one should copy the respective files from the disk. (The disk is also organized in this fashion). Now, the original disk can be put away and the user will have all the files available for running and debugging the data acquisition system.

To run the data acquisition, type "Main" under the directory created for the disk.

The data files from the DAS will be stored in the Bernoulli drive (drive D:), where the user should run the data conversion program VSGBIN as will be explained later.

HARDWARE SETUP:

The most important piece of information that is missing from the manual is that a NULL MODEM connector was used between the remote and the host unit (computer). These null modems were "hidden" in the "Blue Connector Box" (See figure in manual). Since our test did not make use of the original cables, this was a drawback in our schedule and after utilizing a breakout box we were able to pin point the problem. Later when the blue boxes were found we also found the null modem connectors. Two RS232 cables with the proper wires switched were constructed in order to not use the null modem connectors.

One step that we took in order to check if the solid-state disks were not damaged, was to boot a computer with the solid-state disk in one of its ports. We were able to verify that the solid-state disk still contained the necessary files to run the data acquisition.

The Intel 8088 Wild Card has an empty socket, where a math coprocessor would be inserted, but one is not necessary for the data acquisition being used now. The integrity of these cards were also tested.

From the figures of the system in the manual we were able to check some wire connections and adjust them accordingly.

The computer being used as the data acquisition "host unit" should have two 9 pin serial connectors. The top one being used for remote #1 and the bottom one for remote #2.

SOFTWARE USAGE NOTES:

In order to perform a quick test of the data acquisition system one should proceed as follows:

1) Turn remote units on. (They take at least 5 min to warm up)

2) Connect remotes to host unit appropriately (top port - remote #1, bottom port - remote #2)

OBS: Remote #1 should be used for the C-X Band Radar and Remote #2 for the VSG Radar. Although both remotes were designed equally, remote #1 does not perform the beam switching necessary for the VSG Radar properly. Further, as we utilize the data translation software it is expected that remote #1 be used with the C-X Band Radar and remote #2 be used with the (beam switching) VSG Radar.

3) Connect desired signals to one or more of the analog inputs of either (or both) remotes. It is best to connect to analog inputs 1 and 2 since these are the default screen inputs and no prior change will be necessary.

4) At the directory where the software has been installed type "main". This should bring up the main interactive screen.

5) Next, one should move (using arrow keys) to the box where Rem 1 Active is written. By pressing the enter key, followed by the F1 key the No will be toggled to a Yes. This should be done to "turn on" the remote being used (Rem #1, Rem #2 or both).

6) Next, one should move to the Set Run Time box and toggle it in order to choose the number of minutes to run the data acquisition. A small self explanatory screen will appear at the bottom right of the screen.

7) After these steps are performed one can choose either Display Not

Rec or Display And Rec boxes by toggling either one, in order to start the data acquisition.

8) It might be more obvious as to what signal is being received, to observe the data from the real time display instead of the channel # display. In order to switch between the two press F5.

The only two other boxes utilized by our test were the Set Run Number and Set TD Parameters options. The Set Run Number lets you input a 3 digit number that will be the extension of the file that will be created with your data if recorded. The Set TD Parameters option will let you choose which channel you wish to track on the real time display. Note that only two channels may be tracked at once. Again a small self explanatory screen will appear at the bottom right.

CONVERTING DATA FILE TO MATLAB FILE:

There is a program that will translate the data file outputed by the data acquisition system into one that is acceptable by matlab. This program is called VSGBIN.

The file created by the data acquisition system will be in drive D. The VSGBIN file should also be in drive D. Once in drive D, one should type DIR in order to verify what name was given to your file. This "name" should be the date and time stamp from the computer followed by the extension specified by the user. By typing "vsgbin" the conversion program starts.

Several prompts will appear and here are the explanations of what they expect from the user:

1) Remote Number to be processed?

1 or 2 depending of which one you used and want processed now.

2) Numbers of channels to be processed?

Indicate the number of channels in the remote specified above that you wish to process now.

3) Channel [1]:
Channel [2]:

Indicate the number of the channel you want processed.

4) File to be processed?

Indicate file name. (Example of file name: 03041447.001)

5) Name of data-matrix for matlab:

Indicate a name for the .mat file to be created.

6) Number of minutes to be processed?

Indicate the number of minutes you wish to be processed (This should be less than or equal to the run time)

7) Enter n [period = n * 33ms]:

The data acquisition system has a recorded value every 33ms, indicate if you want every one (n = 1), every other one (n = 2), every third one (n = 3), and so on...

For remote #2, since there are three beams the period is 99ms.

8) Remote number used for skipping?
Number of initial samples to be skipped?

This feature allows you to skip some of the initial data which may be corrupted if the radar had not yet warmed up. In order to skip this feature, just type the remote number being used and then zero (0) for the number of samples to be skipped.

9) Precision [1 for single; 2 for double]:

Indicate the matlab type in which one wants the matlab datamatrix.

If any of the prompts is answered incorrectly a DANGER flag will appear on the screen.

Some extra notes:

The matlab file-matrix created with remote #2 will have 3 columns for each channel processed. This is because there are three beams involved with the VSG. If a signal generator is used as a test these three columns will have identical data.

IMPORTANT: When plotting the columns of the matrix in matlab, one should be aware that the even numbered channels are treated as power channels and the odd numbered channels are treated as normal channels. What this means is that when plotting data taken and processed from an even numbered channel the plot will represent the square of the data taken.

```
 Volume in drive E is VSG DAS
 Volume Serial Number is 0E30-13DD
 Directory of B:\DISKD_2

.            <DIR>     02-04-94   12:20p
..           <DIR>     02-04-94   12:20p
COMMAND  COM      25307 03-17-87  12:00p
COMMBIOS COM       1022 08-17-90  11:56p
COMMISR  COM       1187 08-17-90  11:57p
COMMDRV  SYS       1110 10-31-89  12:04a
CONFIG   SYS        112 06-22-90   5:39p
ANSI     SYS       1678 03-17-87  12:00p
RESMEM   SYS        465 08-17-90  11:57p
RMEXEC   EXE      18608 08-19-90  11:37p
CBU      COM       1022 01-24-90  12:12a
CMODE    COM       1872 06-25-87   2:05a
ISR      COM       1187 11-16-87  12:47a
RM       SYS        465 11-12-86  12:06a
AUTOEXEC BAT         28 08-19-90  11:41p
       15 file(s)      54063 bytes
                      651776 bytes free
 Volume in drive B is VSG DAS
 Volume Serial Number is 0E30-13DD
 Directory of B:\DISKD_1

.            <DIR>        02-04-94  12:20p
..           <DIR>        02-04-94  12:20p
COMMAND  COM     52925 03-09-93   6:00a
COMMBIOS COM      1022 08-17-90  11:56p
COMMISR  COM      1187 08-17-90  11:57p
COMMDRV  SYS      1110 10-31-89  12:04a
CONFIG   SYS       141 02-01-94   3:14p
ANSI     SYS      1678 03-17-87  12:00p
RESMEM   SYS       465 08-17-90  11:57p
RCD      SYS     12708 04-08-88  12:00a
CBU      COM      1022 01-24-90  12:12a
CMODE    COM      1872 06-25-87   2:05a
ISR      COM      1187 11-16-87  12:47a
RM       SYS       465 11-12-86  12:06a
AUTOEXEC BAT        20 08-19-90  11:31a
GOTH     CHR      8560 05-02-89   5:50a
LITT     CHR      2138 05-02-89   5:50a
SANS     CHR      5438 05-02-89   5:50a
TRIP     CHR      7241 05-02-89   5:50a
EGAVGA   BGI      5363 05-02-89   5:50a
MAIN     EXE     79344 09-28-90   1:33a
README   TXT      1073 10-22-90   8:50a
STATELOG           565 02-01-94   3:30p
       23 file(s)     185524 bytes
                      651776 bytes free
 Volume in drive B is VSG DAS
 Volume Serial Number is 0E30-13DD
 Directory of B:\DISKB

.            <DIR>     02-04-94  12:20p
..           <DIR>     02-04-94  12:20p
GDECLARE PAS     17654 09-20-90   8:53p
GPROCFNC PAS     61583 09-20-90   9:45p
RMERROR  PAS      1732 08-22-90  12:31p
RMOFF    PAS      2737 09-17-90   4:22a
RMXMTBLK PAS      3367 09-17-90   4:22a
RMEXEC   PAS      4078 09-17-90   4:09a
812TPAS  OBJ      5563 07-04-89   4:55p
        9 file(s)      96714 bytes
                      651776 bytes free
 Volume in drive B is VSG DAS
 Volume Serial Number is 0E30-13DD
 Directory of B:\DISKC

.            <DIR>     02-04-94  12:20p
..           <DIR>     02-04-94  12:20p
VSGBIN   EXE     42679 08-05-91  10:43a
        3 file(s)      42679 bytes
                      651776 bytes free
 Volume in drive B is VSG DAS
 Volume Serial Number is 0E30-13DD
 Directory of B:\DISKA

.            <DIR>     02-04-94  12:20p
..           <DIR>     02-04-94  12:20p
HMERROR  PAS      3207 08-19-90  12:50p
HMOFF    PAS      2403 08-19-90  12:51p
HMRCVBLK PAS      5131 08-19-90   6:20p
HMEXEC   PAS      4198 08-19-90   6:01p
INITGRPH PAS     19230 08-14-90   4:44p
MKSCREEN PAS      7969 08-14-90   2:23p
GPROCFNC PAS     60378 08-20-90  12:36a
GDECLARE PAS     17799 08-19-90   8:49p
DISPLAY  PAS     32349 11-24-93   5:42p
MAINSTAT PAS     35393 08-15-90  12:23p
MAIN     PAS      1260 08-14-90   4:55p
TEMP     PAS       111 08-15-90  12:23p
TYPES    PAS     12463 08-15-90  12:24p
POLSTAT  PAS     11549 08-13-90  11:55a
HEADSTAT PAS     11529 08-14-90   5:09p
DISCACC  PAS      1253 08-14-90   5:57p
TIMESTAT PAS     20014 08-08-90  12:37p
TEXTSTAT PAS      5742 08-08-90  12:36p
RUNSTAT  PAS     15464 08-08-90   1:04p
812TPAS  OBJ      5563 07-04-89   4:55p
TURBO1   TP       1226 08-14-90   2:21p
GOTH     CHR      8560 05-02-89   5:50a
LITT     CHR      2138 05-02-89   5:50a
SANS     CHR      5438 05-02-89   5:50a
TRIP     CHR      7241 05-02-89   5:50a
EGAVGA   BGI      5363 05-02-89   5:50a
RESULT   TXT       480 08-14-90   5:57p
GDEC1 RE TPU     11248 11-24-93   5:43p
GPROCFNC TPU     34688 11-24-93   5:43p
TYPES    TPU      6864 11-24-93   5:43p
INITGRPH TPU      8224 11-24-93   5:43p
MKSCREEN TPU      6032 11-24-93   5:43p
DISPLAY  BAK     32341 11-24-93   5:16p
       35 file(s)     402848 bytes
                      651776 bytes free
```