APPENDIX REM - RC3000 REMOTE CONTROL OPTION

Revision: 6 May 2008, Software Version 1.59

This appendix describes the configuration required and the commands used to implement the remote control interface for the RC3000 antenna controller.

1.1 Appendix Organization

This appendix is provided as a supplement to the baseline RC3000 manual. The corresponding paragraphs in the baseline RC3000 manual are referred to when data specific to the remote control option are described.

Section 1 provides a general overview of the SAbus protocol upon which the RC3000's remote interface is based.

Section 2 describes hardware and software configuration procedures.

Section 3 provides the details of the RC3000 specific commands.

Section 4 provides troubleshooting tips related to the remote control option.

1.3 Theory of Operation – SAbus Overview

The RC3000 command set conforms to the SAbus protocol originally defined by Scientific Atlanta.

The RC3000 may be configured to interface via either the RS-232 or RS-422 electrical standard. Even if the controller is configured as RS-232, it still must be addressed as if it is on a "bus". The following overview describes the more general case of the RC3000 existing on a RS-422 bus.

Electrical Specifications

RS-422 is a unipolar, balanced, 5-volt serial interface designed to connect equipment which must exchange data over considerable distances with high-noise immunity and high speed. Standard IC drivers and receivers are available for RS-422 that convert to and from TTL logic levels. The RS-422 drivers/receivers in the controllers allow up to 32 devices to be connected in parallel with up to 1,500 feet between the master and group of controllers.

Physical Specifications

The physical implementation of the interface takes the form of a 9-pin "D" connector located on the rear panel of the controller. This connector and its wiring is compatible with EIA RS-449, which is the mechanical specification for RS-422/423-compatible equipment. The 9-pin connector chosen is described as the secondary interface in RS-449 and has only the four data lines and shield. No hardware handshaking is used in the protocol, so all the control lines specified for the standard 37-pin connector are not needed. The controller operates as a slave only and has a female connector, whereas master devices have male connectors. Multiple controllers, connected in a daisy chain fashion, can operate in electrical parallel with only a single 5-conductor cable required to connect all devices controlled by a master. Figure 1 illustrates the connection of a master and multiple controllers.

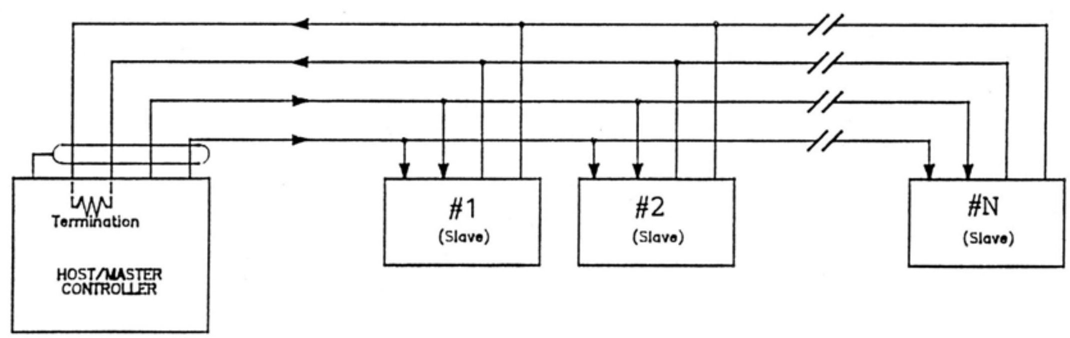


Figure 1. RS-422 Master and Slave Devices on a Network

RS-422 Protocol

The interface is a multi-drop, balanced line, asynchronous, full-duplex communications link designed to interconnect equipment for remote control and switching applications. Products that are compatible can be linked together over a parallel-connected 4-wire circuit without regard to their particular function.

Each network configuration can have one master and up to 32 slave devices. Each controller is internally configured to respond to a unique address. A master could be a protection switch, earth station controller, or any micro- or mini-computer that is electrically and operationally compatible with RS-422. Since the electrical specifications are very similar to EIA standards RS-422 and RS-449, virtually any computer that meets these standards is capable of controlling remote devices.

Data Format

The data format supports the industry's standard asynchronous ASCII format with one start bit, eight data bits (7-bit ASCII with the 8th bit sent as even parity), and one stop bit. The ASCII control character subset 00-1F (hex) are reserved for message control. The printable ASCII characters 20-7F (hex) are used for address, command and data characters. The standard bus data rate via direct connect (up to 1,500 ft.) is 9,600 BAUD; the data rate for devices connected to a master via modem is 1,200 BAUD, typically.

Message Protocol

Message format and protocol over the bus is a derivative of IBM's binary synchronous communications protocol (BISYNC). The master station sends a command over the bus to all remote stations. The station whose address is contained in the second byte of the command message carries out the requested commands, and then replies with a response message containing its own address and status information relating to its present condition. A remote station only sends a response following a command containing its unique address from the master. This prevents bus contention caused by more than one remote device communicating over the bus at the same time.

A remote device ignores all commands that contain parity or checksum errors, protocol errors, a wrong address, or message overrun errors. A remote device replies with a not-acknowledged (NAK) character, 15 hex, if it receives an invalid command or data.

Message Format

Command messages (see Figure 4) begin with Start-of-text byte, STX, followed by a remote address, a command byte and multiple data bytes. The End-of-text byte, ETX, is sent following the last data byte, and the message is terminated by a checksum character. Response messages are identical to command messages in format (but not content) with the exception of the ACK (Acknowledge) or NAK (Not Acknowledge) character at the start of the message instead of STX. Figure 4 illustrates the format of the command and response messages. A command or reply message may have a variable length.

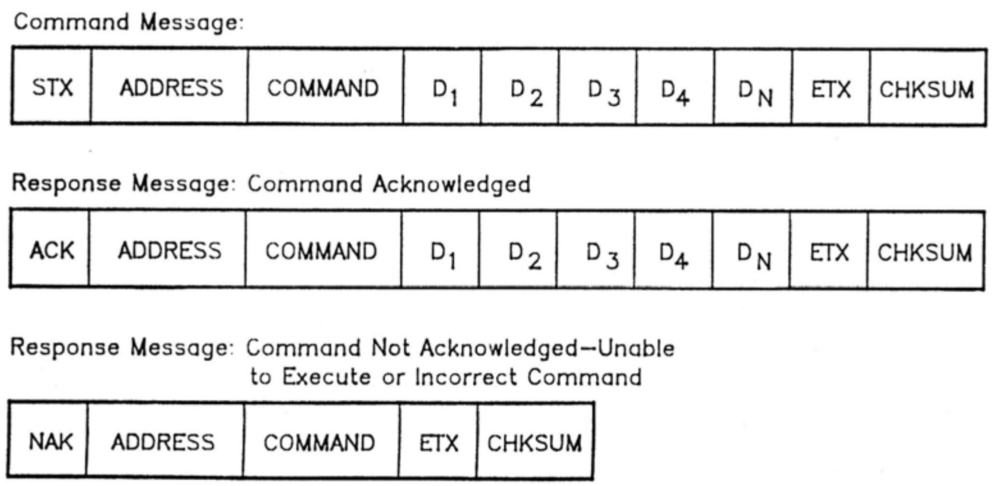


Figure 4. Message Format

Message Delimiters

A command message begins with STX (02 hex), the ASCII Start-of-text control character. A message-acknowledged reply begins with ACK (06 hex), the ASCII Acknowledge control character, and a message-not acknowledged reply begins with NAK (15 hex), the ASCII Not Acknowledge control character. All messages end with the ETX (03 hex), the ASCII End-of-text control character, followed by the checksum byte.

Address Character

The device address must be a valid ASCII printable character between 31 and 6F in hex; thus, 63 addresses are possible. These are set in the controller in decimal format, or 49-111.

Command Character

The command character (CMD) immediately follows the device address and specifies one of several possible commands for a particular device. See RCI document RS422EXT.INC for a complete description of these commands.

Check Character

The last character of any message is the check character (CHK). This character is simply the bit-by-bit exclusive OR of all characters in the message starting with the STX character through the ETX character. This forms a Longitudinal Redundancy parity check over the entire message.

Message Timing

The NAK or ACK reply does not signify that a function has actually taken place, but only that the message was received and understood. The user should query the controller later to see if the command was actually carried out, or is still in progress. Figure 5 shows the controller state diagram.

Command Restrictions

All slaves will respond to a command "0", 30 (hex), with 6 data bytes of ASCII characters in the following form:

ACK ADDR 30 R C D3 K D5 D6 ETX CHSUM

where D5-D6 are two ASCII characters representing a software version number and D3 is 1 for an RC1000A or 2 for an RC2000A.

Slave State Diagram : Introduction

General Description. The slave State diagram (see Figure 5) presents the required protocol implementation at the slave device that guarantees the proper transfer and processing of communication messages sent by a Master controller.

State Diagram Notation. Each state that a slave can assume is represented graphically as a circle. A single-digit number is used within the circle to identify the state.

All permissible transitions between states are represented graphically by arrows between them. Each transition is qualified by a condition that must be true in order for the transition to occur. The device will remain in its current state if conditions which qualify transitions leading to other states are false, or conditions that qualify pseudo-transitions are true. A pseudo-transition is a transition that occurs within the same state and is represented graphically by arrows leaving from and arriving at the same state. Table 1 describes mnemonics used to identify transitions in the state diagram.

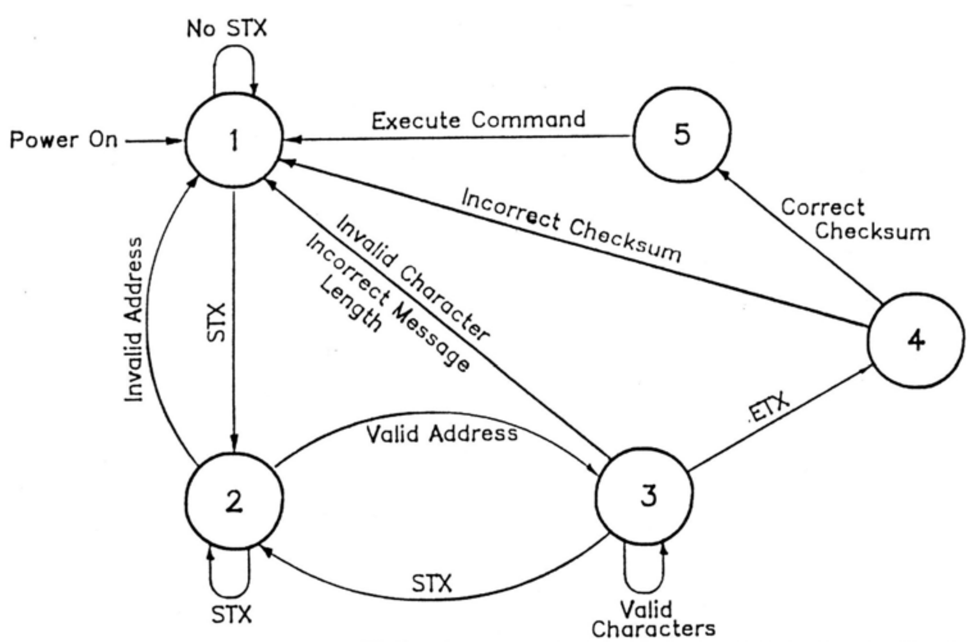


Figure 5. State Diagram – RCI Protocol

Table 1. State Diagram Mnemonics

Mnemonics	Description
STX	Start-of-Text ASCII control character, used as a header in command messages to identify the beginning of a new message.
ETX	End-of-Text ASCII control character, used as a termination character in messages to identify the end of data.
Checksum	LRC byte (Longitudinal Redundancy Check) is a last byte in the message data block. The value of LRC byte is the exclusive OR of all message bytes including the STX and the ETX bytes and is used to detect errors during transmission of data.

State Descriptions

State 1 (Slave Idle State). In State 1, a slave is ready to receive a new message, and therefore, must complete any previous message reception. A slave always powers on in State 1.

A slave will exit State 1 and enter State 2 (Slave Addressed State) only if STX byte is received.

State 2 (Slave Addressed State). In State 2, a slave is waiting to receive the address byte, the second byte of a command message.

A slave will exit State 2 and enter:

a. State 3 (Slave Data State) if received address byte equals a slave's address.

b. State 1 (Slave Idle State) if received address byte does not equal a slave's address.

c. State 2 (remain in current state) if STX byte is received, which may be the beginning of a new message data block.

State 3 (Slave Data State). In State 3, a slave is engaged in receiving the command and associated data bytes sent by a master-controller.

A slave will exit State 3 and enter:

a. State 4 (Slave Data Error State) if ETX byte is received signifying the end of data in the message.

b. State 1 (Slave Idle State) if invalid command, or data character, or incorrect number of data bytes is received.

State 4 (Slave Data Error State). In State 4, a slave is waiting to receive a Checksum byte which tests the transmitted message for errors.

A slave will exit State 4 and enter:

a. State 5 (Command Execute State) if a Checksum byte is true -received LRC value of Checksum byte equals the LRC value computed by a slave during message reception.

b. State 1 (Slave Idle State) if a Checksum byte is false -- received LRC value of Checksum byte does not equal the LRC value computed by a slave during message reception.

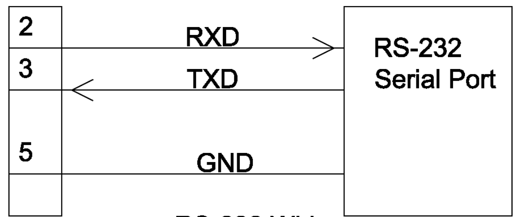
State 5 (Command Execute State). In State 5, a slave, having completed reception of a message, executes a function specified by a command byte. A slave will send an appropriate response message to a master-controller after receiving the last character of the message.

A slave will always exit State 5 and enter Device Idle State, State 1.

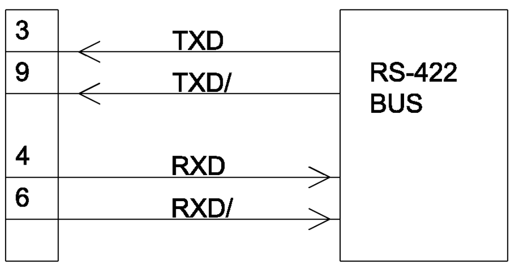
2.0 INSTALLATION

Electrical Interface

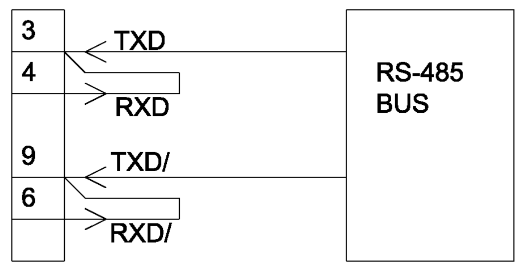
The following is a repeat of section 2.2.11 of the baseline RC3000 manual describing the J5 (DB-9 Female on backpanel) remote control connector. The RC3000 may be configured to communicate either by the RS-232 or the RS-422 / RS-485 standards.



RS-232 Wiring



RS-422 Wiring



RS-485 Wiring

The RC3000 is shipped from the factory configured for RS-422/RS-485 operation.

Original RC3000s

To configure the RC3000 for RS-422 or RS-485, connect the remote control ribbon cable inside the RC3000 to connector J10 on the feature board. Jumper J12 on the feature board must be placed in the RS-422 position.

To configure the RC3000 for RS-232, connect the remote control ribbon cable inside the RC3000 to connector J11 on the feature board. Jumper J12 on the feature board must be placed in the RS-232 position.

Second Generation RC3000s

To configure the RC3000 for RS-422 or RS-485, set the J12 jumper on the analog board to the -422 position and set jumpers X1 through X5 to the -422 position.

To configure the RC3000 for RS-232, set the J12 jumper on the analog board to the -232 position and set jumpers X1 through X5 to the -232 position.

Newer RC3000s also allow for the remote control to interface with an optional internal card hosting a web page. To configure the RC3000 for this, set the J12 jumper to the "RABBIT" position.

Communications Parameters

The controller's baud rate and address must be set before communication with a host is possible. These quantities can be specified via the REMOTE CONTROL configuration screen.

```
  ENABLED:1                CONFIG-REMOTE
  ADDRESS: 50
BAUD_RATE:6          JOG:20
REMOTE CONTROL <0>DISABLED <1>ENABLED
```

ENABLED: **REMOTE CONTROL <0>DISABLED <1>ENABLED**

The remote_enabled flag allows the user to disable the ability to remotely control the RC3000. This may prove useful if the user wants to only operate from the front panel. If remote control is enabled, commands received will force the controller into REMOTE mode.

ADDRESS: **BUS ADDRESS <49-111>**

The comm_port_address item allows the user to specify an unique bus address for the RC3000.

BAUD RATE: **BAUD<1-3 2-6 3-12 4-24 5-48 6-96>(x100)**

The comm_port_baud_rate item allows the user to choose one of six possible baud rates from 300 to 9600.

JOG: **REMOTE JOG HOLD <1-40>**

The remote_jog_hold_timer value is used to smooth azimuth, elevation and polarization jog movements when the RC3000 is operating in MANUAL mode and being commanded via a "remote front panel". The entered number multiplied by 1/40 of a second represents how long a remote front panel jog command will last. For example, a value of 20 corresponds to a hold period of 0.5 seconds.

This configuration item exists to allow the RC3000 to adjust to the key repeat rate from the computer sending the remote front panel commands. This value will have to be adjusted to match the characteristics of different computers implementing a "remote front panel" scheme.

If the value is too low, manual movements will be jerky as the operator holds down a jog key from the remote front panel. In this case the RC3000 sees a jog key from the remote front panel but the hold timer expires before another jog key command is received.

If the value is too high, a remote front panel manual move will continue for too long of a period after the operator releases the jog key.

NOTE: The REMOTE CONTROL configuration screen was presented here but would normally be described in section 3.3.1.3 (Super-User Access Items) of the baseline RC3000 manual.

3.0 DETAILED OPERATION

Command Set

The following table lists the RC3000's remote command set.

CODE (hex)	COMMAND	PARAGRAPH
30	Device Type Query	3.1
31	Device Status Poll	3.2
32	Auto Move	3.3
33	Azimuth / Elevation / Polarization Jog	3.4
34	Polarization Preset	3.5
35	Query Name	3.6
36	Miscellaneous	3.7
37	Reflect Display	3.8
38	Load Signal Strength*	3.9
39	Write Satellite Data	3.10
3A	Read Satellite Data	3.11
3B	Write Two Line Element Data	3.12
3C	Read Two Line Element Data	3.13
3D	Write IESS-412 Data*	3.14
3E	Read IESS-412 Data*	3.15
3F	Read Pulse Count	3.16
40	Extended Device Status Poll	3.17
41	Remote Locate	3.18
42	Remote Store	3.19
43	Write Signpost Data	3.20
44	Read Signpost Data	3.21
45	Read Navigation Data	3.22
46	Write Navigation Data	3.23
47	Jog with Minimal Reply	3.24
48	Remote Keypress	3.25
	*- not yet implemented on the RC3000	

Each command is detailed in paragraphs 3.1 on. The next few paragraphs describe characteristics common to all RC3000 commands.

Message Delimiters

Here are the delimiters used with SA bus messages, along with their values in hex and decimal.

ASCII name	hex value	decimal value
STX	2	2
ETX	3	3
ACK	6	6
NAK	15	21

ASCII Table

As reference, the following table shows the set of ASCII codes available for use by the RC3000 remote protocol.

HEX	0_	1_	2_	3_	4_	5_	6_	7_
_0			Blank	0	@	P		p
_1			!	1	A	Q	a	q
_2	STX		"	2	B	R	b	r
_3	ETX		#	3	C	S	c	s
_4			$	4	D	T	d	t
_5		NAK	%	5	E	U	e	u
_6	ACK		&	6	F	V	f	v
_7			'	7	G	W	g	w
_8			(	8	H	X	h	x
_9			)	9	I	Y	i	y
_A			*	:	J	Z	j	z
_B			+	;	K	[	k	{
_C			,	<	L	\	l	|
_D			-	=	M	]	m	}
_E			.	>	N	^	n	
_F			/	?	O	_	o	

RC3000 Online/Offline Reply

To enable remote control of the RC3000, the internal remote_mode_enable_flag must be set. This flag is set at the Remote Mode Enable prompt in CONFIG-REMOTE mode. When this flag is set, remote commands will be processed by the RC3000.

NOTE: Unlike the RC2000 and RC2500, device type query and device status poll commands do not force the RC3000 into REMOTE mode if it is not currently in REMOTE mode. A few commands (such as Auto Move to a stored satellite) cause the RC3000 to enter the REMOTE mode. Certain commands can be processed by TRACK mode.

If the remote_mode_enable_flag is FALSE and a valid command (correct address, checksum, etc.) arrives via the serial port, the offline reply is sent to the host. Here is the format of the offline reply:

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	'CC'	the command code of the message which triggered this reply.
byte 3	'F'	ASCII 'F', for offline.
byte 4	ETX	
byte 5	'checksum	the checksum. The checksum character is simply the bit-by-bit exclusive OR of all characters in the message starting with the STX character through the ETX character.

RC3000 Unrecognized Commands - NAK Reply

If an unrecognized command arrives (one whose command code is either unknown or whose length is not compatible with the given command code, but which has the correct address and checksum), a NAK reply is sent to the host. The format of the NAK reply is:

byte 0	NAK	
byte 1	A	where A is the RC3000 address
byte 2	'CC'	the command code of the unrecognized message.
byte 3	ETX	
byte 4	checksum	

3.1 Device Type Query Command

The SA Bus specification requires that command character 30h must trigger the return of the six character device type string. The message format for this query will be ...

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	30h	30 hex - the device type query command code
byte 3	ETX	
byte 4	checksum	

The reply to this query will consist of 11 bytes ...

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	30h	the device type query command code
bytes 3,4:	“3K”	controller type – 3K for RC3000
bytes 5-8:	'”A.BC”	version number – example: 1.22
byte 9	ETX	
byte 10 :	checksum	

3.2 Device Status Poll Command

The SA Bus specification requires that command character 31h cause a device to return status information. The reply to this command includes azimuth, elevation and polarization position, current satellite name, as well as limit, alarm and drive status information. The status poll command message consists of 5 bytes and the format is:

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	31h	the status poll query command code
byte 3	ETX	
byte 4	checksum	

The response to this command will consist of 52 bytes, which will be a combination of ASCII and binary data fields. The binary data will be placed in the lower nibble of a byte whose higher nibble will be initialized to a value which will make the result an ASCII character. The idea with this response is to be able to reproduce the information presented on the LCD to the user when manual mode is active. The format of the response is:

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	31h	the status poll query command code
bytes 3-12	sat_name	This field will contain the satellite name in upper case letters. If the name does not occupy the entire field the name will be left justified and the string will be padded with blanks. If a satellite name is not currently displayed, this field will contain blanks.
byte 13	not used	reserved for future use, currently initialized to 0100$0000b.
byte 14-19	azimuth position	This field will contain the formatted azimuth position -180.0 to 180.0. If the analog to digital converter detects an error this field will contain '******'. NOTE: If the primary azimuth sensor is the fluxgate compass, this field will display a value from 0.0 to 359.9 in bytes 14-18. Byte 19 will display either a "M" or "T" to indicate whether the value in bytes 14-18 represent a M(agnetic) or T(rue) heading value.
byte 20-25	elevation position	The field will contain the formatted elevation position, -180.0 to 180.0. If the analog to digital converter detects an error this field will contain '******'.
byte 26-31	polarization position	This field will contain the formatted polarization position -180.0 to 180.0. If the antenna is not equipped with a rotating feed or if the analog to digital converter detects an error this field will contain '******'.

Device Status Poll Command <continued>

byte 32 azimuth limits, binary data

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 A B C
```

A ‘0’ in a bit position implies that the antenna is not at the limit, a ‘1’ in the bit position implies that the antenna is at the limit. Here is the mapping of bit positions to the limits ...

.

A - Azimuth Clockwise
B - Azimuth Counterclockwise
C - Azimuth Stow

byte 33 elevation limits, binary data

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 A B C
```

A ‘0’ in a bit position implies that the antenna is not at the limit, a ‘1’ in the bit position implies that the antenna is at the limit. Here is the mapping of bit positions to the limits ...

.

A – Elevation Up
B – Elevation Down
C - Elevation Stow

byte 34 polarization limits, binary data

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 A B C
```

A ‘0’ in a bit position implies that the antenna is not at the limit, a ‘1’ in the bit position implies that the antenna is at the limit. Here is the mapping of bit positions to the limits ...

.

A - Polarization Clockwise
B - Polarization Counterclockwise
C - Polarization Stow

Device Status Poll Command <continued>

byte 35 polarization equipment and display status code - binary data

```
7 6 5 4   3 2 1 0
0 1 X X $ Y Z Z Z
```

where ‘XX’ is ...

00 if a rotating feed is not present in the system
01 if a single port rotating feed is present in the system
10 if a dual port rotating feed is present in the system. A dual port rotating feed can simultaneously receive both horizontally and vertically polarized signals.

where ‘Y’ is ...

0 if polarization movements are not allowed.
1 if polarization movements are allowed.

Discussion - The ‘Y’ field described above only contains meaningful data when TRACK mode is active. Polarization movement is not allowed during a TRACK mode peaking operation. If a polarization operation occurs while peaking the antenna the peak obtained may not be reliable. If a ‘go to’ H or V polarization command is received via the serial port the controller will execute the command after the peaking operation is completed. The reply to the ‘go to’ command will be an ACK.

A polarization jog command which is received during a peaking operation will not be registered and executed later. The reply to the command will be a NAK.

where ‘ZZZ’ is ...
000 if the ‘H' polarization code is displayed
001 if the 'h' polarization code is displayed
010 if the 'V' polarization code is displayed
011 if the 'v' polarization code is displayed
100 if no polarization code is displayed

Device Status Poll Command <continued>

byte 36 azimuth movement/alarm status - binary data

```
7 6 5 4   3 2 1 0
0 1 0 S $ X X X X
```

where ‘S’ is ...
0 if the axis is configured for slow speed movement
1 if the axis is configured for fast speed movement

where ‘XXXX’ is ...
0000 no alarms or movement
0010 ccw jog movement pending
0011 cw jog movement pending
0100 ccw automatic movement in progress
0101 cw automatic movement in progress
0111 remotely commanded auto move is in progress

1000 off axis alarm active. This alarm code is reported if an elevation runaway alarm is active.

1001 sensor direction alarm active
1010 runaway alarm active
1011 jammed alarm active
1100 drive alarm active. This is triggered by an overcurrent condition.

Note: Higher value status codes have priority over lower value ones, i.e. if as part of an auto move command the antenna is moving clockwise the status will be reported as 'auto move in progress' rather than 'clockwise movement in progress'.

Device Status Poll Command <continued>

byte 37 elevation movement/alarm status - binary data

```
 7 6 5 4   3 2 1 0
 0 1 0 S $ X X X X
```

where ‘S’ is ...
0 if the axis is configured for slow speed movement
1 if the axis is configured for fast speed movement

where ‘XXXX’ is ...
0000 no alarms or movement
0010 down movement pending
0011 up movement pending
0100 down movement in progress
0101 up movement in progress
0111 an auto move is in progress

1000 off axis alarm active. This alarm code is reported if an azimuth runaway alarm is active.

1001 sensor direction alarm active
1010 runaway alarm active
1011 jammed alarm active
1100 drive alarm active. This is triggered by an overcurrent condition.

Note: Higher value status codes have priority over lower value ones, i.e. if as part of an auto move command the antenna is moving down status will be reported as 'auto move in progress' rather than 'down movement in progress'.

Device Status Poll Command <continued>

byte 38 polarization movement/alarm status - binary data

```
7 6 5 4   3 2 1 0
0 1 0 S $ X X X X
```

where ‘S’ is ...
0 if the axis is configured for slow speed movement
1 if the axis is configured for fast speed movement

where ‘XXXX’ is ...
0000 no alarms or movement
0010 ccw movement pending
0011 cw movement pending
0100 ccw movement in progress
0101 cw movement in progress
0111 an auto move is in progress

1000 off axis alarm active. This alarm code is currently not supported for the polarization axis.

1001 sensor direction alarm active
1010 runaway alarm active
1011 jammed alarm active
1100 drive alarm active. This alarm is not currently supported.

Note: Higher value status codes have priority over lower value ones, i.e. if as part of an auto move command the antenna is moving clockwise the status will be reported as 'auto move in progress' rather than 'clockwise movement in progress'.

byte 39 alarm code - binary data

```
0 1 A A $ A A A A
    5 4   3 2 1 0
```

A5 .. A0 specify the alarm code (0-63). Alarm messages flash on the bottom row of the display. Here are the alarm codes which have been defined ...

0 - No alarm active
1 - Low battery
2 – Azimuth Jammed
3 – Azimuth Runaway
4 – Elevation Jammmed
5 – Elevation Runaway
14 – Communication Port Error
16 – Track Configuration Error
18 - Time/Date Error
22 – Polarization Jammed
24 – Limits Inactive Warning

Device Status Poll Command <continued>

byte 40: Track Mode submode or error status and track frequency band - binary data

```
7 6 5 4   3 2 1 0
0 B B B $ S S S S
```

where 'BBB' is ...

000 - not used to avoid generating a message delimiter character
001 - X band
010 - Ka band
011 - S band
100 - C band
101 - Ku band
110 - band not defined
111 - L band

and where 'SSSS' is ...
0000 - track mode not active
0001 - track setup submode active
0010 - track auto mode entry
0011 - step track submode active
0100 - track auto search submode active
0101 - program track submode active
0110 - track manual search submode active
1000 - track jammed error
1001 - track limit error
1010 - track drive error
1011 - track peak limit error
1100 - track geo position error
1101 - track system error track
1110 - track checksum error

bytes 41-43: AGC Level The AGC channel voltage is represented internally by a numeric value between 0 and 999. This numeric value is converted to an ASCII string - '   0' and '999'. The most significant digit will be placed in byte 41 and the least significant digit will be placed in byte 43. The string will be right justified and padded with blanks (on the left).

Byte 44: AGC Channel The AGC channel currently selected.

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 0 C C
```

where 'CC' is ...
00 - RF
01 – SS1
10 – SS2
11 – Not Applicable

bytes 45 - 49: Reserved - At this time these bytes are initialized to 0100$0000b.
byte 50: ETX
byte 51: checksum

3.3 Auto Move Command

This command causes the controller to automatically position the antenna in either azimuth and elevation and/or polarization. The command contains 16 bytes. Here is the format:

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	32h	the auto move command code
byte 3	polarization	This field can specify 'H', 'V', ' ' (blank), 'P', 'A', 'E' or 'C'.
byte 4-13	sat_name/position	This field specifies the satellite name or a target azimuth and elevation or polarization position.
byte 14 :	ETX	
byte 15 :	checksum	

The normal reply to this command will be the same as the reply to the status poll query except that the command code field will be '32h'. Note that if the satellite name is not found or target positions for a move to a target position are not specified properly a NAK reply will be sent to the host. For forms 1 and 3 of this command (described below), if byte 3 of the command specifies polarization movement but the Polarization Type is set to CIRCULAR (no polariization control device present) the NAK reply will be sent to the host.

The Auto Move command has several forms.

Form 1. If the sat_name/position field contains the name of a satellite saved via the controller's STORE mode the controller will position the antenna at the azimuth and elevation positions associated with that satellite. The satellite name should be in capital letters, left justified and padded on the right with blanks in the sat_name/position field. Note that the satellite name specified in the command must exactly match a satellite name stored in the controller's non-volatile memory. Form 1 automates the RC3000's RECALL mode.

With this form of the command, the polarization field may contain either 'H', 'V', or ' ' (a blank, 20 hex or 32 decimal). If an 'H' or a 'V' is specified, in addition to positioning the antenna in azimuth and elevation, the polarization control device will be commanded to go to the position associated with either the horizontal (if 'H' is specified) or vertical (if 'V' is specified) polarization specified for the satellite. If the field contains a blank the polarization is not changed. For example, this command with 'H' in the polarization field and 'SBS 6   ' in the sat_name/position field will specify an auto move to SBS 6 and the polarization will be adjusted to horizontal for the SBS 6 satellite.

Form 2A. If the sat_name/position field contains a valid pair of azimuth and elevation sensor positions (scaled by 10), the antenna will move to the position specified. The first 5 characters of the sat_name/position field specify the azimuth position (azimuth sub-field) and the last five characters specify the elevation position (elevation sub-field). Within each of the sub-fields the position must be right justified and left padded with zeroes. For example, a sat_name/position field value of '-152500456' specifies an azimuth position of -152.5 degrees and an elevation position of 45.6 degrees. For this form of the auto move command, only the blank character is accepted in the polarization field.

Form 2B. If the antenna system is equipped with "count"-type sensors (pulse or resolvers), Automove form 2B accepts a pair of pulse or resolver counts to move to. For example, a sat_name/position field value of '1105012152' specifies an azimuth pulse position of 11050 and an elevation pulse position of 12152. The polarization field should contain a 'C'.

Form 2C. For systems that are capable of generating azimuth and elevation position feedback to the one hundredth of a degree resolution, form 2C provides the capability to command either an azimuth or an elevation movement to a target specified within one hundredth of a degree. To command an azimuth move, insert 'A' into byte 3. To command an elevation move, insert 'E' into byte 3. Bytes 4 to 9 contains the target azimuth or elevation position. As with form 2A, the position must be right justified and left padded with zeroes. Bytes 10 to 13 should be filled with blanks. For example, if byte 3 is 'A' and bytes 4 – 9 contain '-12345', an azimuth automove to the target of -123.45 will be initiated.

Form 3. If the polarization field contains the 'P' character, the command is interpreted as a go_to_polarization command. For this form of the command, the first 5 characters of the sat_name/position field specify the target polarization position in the controller's internal polarization position representation (polarization sub-field). The polarization position in the polarization sub-field must be right justified and left padded with zeroes. The second 5 characters of the sat_name/position field must contain '00000'. For example, if the sat_name/position field contains '0050000000' the polarization control device is commanded to adjust the polarization to a position of 50.0.

3.4 Azimuth/Elevation/Polarization Jog Command

This command jogs the antenna in azimuth, elevation, or polarization. The command contains 11 bytes. Here is the format of the command:

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	33h	the command code
byte 3	direction	This field can specify 'E', 'W', 'D', 'U', O, L, or 'X' where ... E refers to azimuth Counter clockwise, W refers to azimuth clockWise, D refers to elevation Down, U refers to elevation Up, O refers to polarization cOunter clockwise, L refers to polarization cLockwise, and X means stop all movement.
byte 4	speed	This field specifies the jog speed, either 'F' (Fast) or 'S' (Slow). Note that this field must contain a valid value even if the direction field specifies 'X' (Stop).
bytes 5-8 :	duration	This field specifies the duration of the jog command in milliseconds. The valid range of values for this field is '0000' to '9999'. As a practical matter, the resolution of the timer used to time the move is approximately 50 milliseconds, so any move will be for a time interval equal to a multiple of approx. 50 milliseconds. Note that this command must contain a valid value even if the direction field specifies 'X' (Stop).
byte 9	ETX	
byte 10 :	checksum	the checksum

If this command can be executed, the reply to this command will be the same as the reply to the status poll query command except the command code will be '33h'. A NAK reply will be sent to the host if the direction specifies C, W, U, D, O, or L and the limit input associated with the axis and direction specified by the command is asserted (only for versions of the controller which support individual limits). Note that the controller can only support a remote jog about a single axis. For example, if a remote jog is in progress about the azimuth axis and a remote elevation jog command is received (that can be executed - i.e. no limits or alarms are active), the azimuth jog will terminate regardless of the duration specified for the remote azimuth jog. A NAK reply will also be sent to the host if polarization movement is specified and the Polarization_Type CONFIG mode item is set to 0 (Circular Polarization). If the direction byte contains ‘X’ all antenna movement will stop. If TRACK mode is active and the direction byte specifies ‘X’, ‘C’, ‘W’, ‘D’, or ‘U’ REMOTE mode will receive control and all tracking will cease. If TRACK mode is active and a peaking or search operation is in progress the NAK reply will be returned to the host.

3.5 Polarization Command

The following command specifies a move to a preset polarization position. The command contains 6 bytes.

The format of the command is as follows;

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	34h	the command code
byte 3	'X'	this field will specify either ‘H’ or ‘V’ where ... H specifies that the controller drive the polarization to the horizontal polarization position associated with the satellite that was the last target of an auto move operation. V specifies that the controller drive the polarization to the vertical polarization position associated with the satellite that was the last target of an auto move operation.
byte 4	ETX	
byte 5	checksum	

The reply to this command will be the same as the reply to the status poll query command except the command code will be 34h. Note that the NAK reply will be sent back to the host if there are no satellites available in the RC3000’s memory or if the Polarization Type is set to Circular (No Pol Control). Note also that if the Polarization Type is set to DUAL (2 Port Feed) there is only one polarization position associated with the satellite and receipt of this command with either the ‘H’ or ‘V’ argument will result in a move to the single polarization position associated with the satellite.

If TRACK mode is active and a peaking or search operation is in progress this command will not be executed until after the peaking or search operation terminates. If this occurs the normal acknowledgment will be sent to the host.

3.6 Query Name Command

This query command instructs the RC3000 to send back to the host computer the name of a satellite stored in non-volatile memory (via the controller's STORE mode) and the total number of satellites stored in non-volatile memory. The command contains the index of the desired entry in the satellite list. A maximum of 50 satellites can be stored in memory.

This query command contains 7 bytes and the format is:

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	35h	the query name command code
bytes 3,4	'XX'	where XX is the index of the satellite name being requested. Normally this would be '01' the first time through and then incremented until the 'YY' (YY being the last entry in the list) satellite name is read. The maximum possible range for XX and YY is 1 through 50.
byte 5	ETX	
byte 6	checksum	the checksum

The normal response to this query command contains 19 bytes and the format is as follows;

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	35h	the query name command code
bytes 3,4	'XX'	where XX is the index of the satellite name being requested.
bytes 5,6	'YY'	where YY is the total number of satellite names contained in the list. Repeat this command YY times to download the names of all stored satellites.
bytes 7-16:	sat name	This field will contain the satellite name. The name will be in capital letters and normally be left justified. The only time the satellite name will not be left justified is if the user selected the USER entry from STORE mode and manually entered blank characters before the satellite name.
byte 17:	ETX	
byte 18 :	checksum	the checksum

Note: If entry 'XX' does not exist in the list (or the list has no entries) the NAK reply will be sent back to the host.

3.7 Miscellaneous Command

This command performs miscellaneous functions. Here is the format of the command.

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	36h	the miscellaneous command code
byte 3	'X'	the sub-command code
byte 4	'Y'	the sub-command parameter
byte 5	ETX	
byte 6	checksum	

The sub-command code 'X' can have the following values:

'X' = 'R' This specifies the azimuth or elevation drive reset command. This accomplishes the same function as the DRIVE RESET mode of the RC3000: it allows the user to reset the azimuth, elevation, or polarization alarms. When the sub-command code is 'R', the sub-command parameter 'Y' must be either 'A', 'E', or 'P' (for azimuth, elevation, or polarization respectively) to specify which axis will be reset. If the 'P' command is specified, the command will be accepted only if the Pol Control Equipment Code CONFIG mode item is set to 1 (ONE PORT) or 2 (TWO PORT).

'X' = 'T' This sub-command is used to reset track mode errors (sub-command parameter 'Y' = R). When the TRACK mode ERROR sub-mode is active this command will cause the ERROR sub-mode to terminate. The controller will react as if TRACK mode was activated via RECALL mode. Note that if a system error is active (an error message flashing on the bottom row of the display) the condition which generated the system error must be rectified or the controller will probably return to the TRACK mode ERROR sub-mode. This sub-command can also be used to switch frequency bands when a dual band satellite is being tracked. A sub-command parameter of 'C' will specify C band and a sub-command parameter of 'K' will specify K band. The reply to this command will be a NAK if TRACK mode is not active, the satellite being tracked was not specified as a dual band satellite (when the track was initiated via SETUP mode), or if track polarization movement is not allowed (see byte 32 of the device status poll command). If polarization movements are not allowed the controller is either peaking the antenna or performing a search. Changing the band during a peaking operation or search can cause the antenna to not accurately peak the antenna.

‘X’ = ‘S’ This sub-command is used to initiate an automatic antenna STOW via the RC3000. NOTE: On mounts with no stow function, reply to this subcommand will be a NAK.

‘X’ = ‘D’ This sub-command is used to initiate an automatic antenna DEPLOY via the RC3000. NOTE: On mounts with no deploy function, reply to this subcommand will be a NAK.

‘X’ = ‘A’	This sub-command is used to initiate an automatic search pattern via the RC3000. The sub-command parameter 'Y' must be either an 'E' to initiate an expanding spiral or 'F' to initiate a "flat" spiral pattern. NOTE: Be sure the RC3000 is configured correctly for automatic searching.	
‘X’ = ‘K’	The Keypad Input sub-command sends a keypad value to the RC3000. The RC3000 will react to the keypad value as if the corresponding key on the RC3000’s front panel was pushed. The required values for the 'Y' subcommand are:	
'Y'	30-39h 41-47h	key codes as defined in the following table

CODE	KEY
30h	0/Speed
31h	1/Pol CCW
32h	2/N/EL UP
33h	3/Pol CW
34h	4/E/AZ CCW
35h	5
36h	6/W/AZ CW
37h	7/H
38h	8/S/EL DN
39h	9/V
3A-3Fh	-- unused –
41h	Stop/decimal pt.
42h	+/-/BKSP
43h	Mode
44h	Scroll Up/Yes
45h	Scroll Dn/No
46h	Enter
47h	Mode Group Change*

*the 47h key code can be used to initiate a RC3000 mode group change which normally requires the Mode key to be held down for five seconds continuously.

The reply to the miscellaneous command will be the same as the reply to the status poll query except the command code will be '36h'.

3.8 Reflect Display Command

This command requests the RC3000 to send the 160 (4 rows x 40 columns) characters currently displayed on the LCD. The command format is:

byte 0	STX	
byte 1	A	where A is the RC3000's address
byte 2	37h	reflect display command code
byte 3	ETX	
byte 4	checksum	

The response to this command will be to send the 160 displayed characters in ASCII format plus cursor status. The response format is:

byte 0	ACK	
byte 1	A	where A is the RC3000's address
byte 2	37h	reflect display command code
byte 3-42	row 1	40 characters displayed on row 1 of the LCD
byte 43-82	row 2	40 characters displayed on row 2 of the LCD
byte 83-122	row 3	40 characters displayed on row 3 of the LCD
byte 123-162	row 4	40 characters displayed on row 4 of the LCD
byte 163	cursor row	cursor row position (1-4)
byte 164	cursor col_tens	tens digit of cursor column (0 if column <10)
byte 165	cursor col_ones	ones digit of cursor column
byte 166	cursor status	0 = cursor not blinking, 1 = cursor blinking
byte 167	ETX	
byte 168	checksum	

Since the reply is lengthy, the request to reflect the display should be limited to a frequency less than 1 Hz. This will make the reflected display at the M&C software a little "jumpy" but should allow the operator to see what is happening at the RC3000.

3.9 Load Signal Strength Command

NOTE: This command is not yet implemented on the RC3000.

This command instructs the RC3000 to load supplied data into a signal strength register to be used in the tracking algorithm. The command contains 2 information bytes that hold ten bits of signal strength, one bit that indicates a modem lock and 5 placeholder bits. This command consists of a total of 7 bytes. The format is:

byte 0:	STX	
byte 1:	A	where A is the RC3000 address
byte 2:	38h	the Signal Strength command code

byte 3: This byte contains the five least significant bits of the 10 bit signal strength as reported by the system and a one-bit modem lock indicator.

7	6	5	4	3	2	1	0
0	1	k	s4	s3	s2	s1	s0

where,

k	meaning
0	modem is unlocked
1	modem is locked

and

s4 s3 s2 s1 s0 are the 5 least significant bits of a 10 bit signal strength

byte 4: This byte contains the five most significant bits of the 10 bit signal strength as reported by the system.

7	6	5	4	3	2	1	0
0	1	0	s9	s8	s7	s6	s5

where,

s9 s8 s7 s6 s5 are the 5 most significant bits of a 10 bit signal strength.

byte 5:	ETX	
byte 6:	checksum	the checksum

The normal response to this query command contains 6 bytes and the format is as follows;

byte 0:	ACK	
byte 1:	A	where A is the RC2000 address
byte 2:	38h	the query name command code
byte 3:	‘0’	ASCII ‘0’, for online.
byte 4:	ETX	
byte 5:	checksum	the checksum

Satellite Data Commands

The next six commands allow for the transfer of preset (user-defined) satellite data to and from the RC3000.

-- the Read/Write Satellite Data commands transfer basic data required for any sat (name, longitude, inclination, RF band, ephemeris availability, polarization offset).
-- RC3000 may hold basic data for 20 satellites
-- the first 10 sat data sets may have associated ephemeris (tle or iess) data
-- tle and iess read/write commands allow for storing of ephemeris data
-- index into stored tle and iess arrays is same as basic sat data index

3.10 Write Satellite Data Command

This command downloads basic satellite data into the RC3000's list of user defined satellites.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	39h	Write sat data command code
byte 3	Index Tens	Tens digit of index that sat data set is to be stored (0 if index < 10) NOTE: index value may be between 1 to 20
byte 4	Index Ones	Ones digit of index that sat data set is to be stored
bytes 5-14	Sat Name	10 character satellite name to be associated with index
bytes 15-20	Longitude	Nominal satellite longitude -179.9 to 179.9 (West longitude negative) Left Justify and pad with blanks
bytes 21-22	Inclination	Satellite inclination 0 to 19 Left Justify and pad with blanks
byte 23	Band	RF Band (0-C, 1-Ku, 2-C/Ku, 3-L, 4-X, 5-Ka, 6-S)
byte 24	Ephem	Ephemeris Data Present (0-none, 1-TLE, 2-IESS-412)
bytes 25-29	Pol Offset	Polarization Offset -90.0 to 90.0 negative = counterclockwise Left Justify and pad with blanks
byte 30	ETX	
byte 31	Checksum	

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	39h
byte 3	ETX
byte 4	Checksum

3.11 Read Satellite Data Command

This command uploads a stored set of satellite data.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	3Ah	Read Satellite Data command code
byte 3	Index Tens	Tens digit of sat data index (0 if index < 10, NOTE: index value between 1 & 20)
byte 4	Index Ones	Ones digit of sat data index
byte 5	ETX	
byte 6	Checksum	

Reply

byte 0	ACK or NAK	
byte 1	address	
byte 2	3Ah	
byte 3	Index Tens	Tens digit of sat data index (0 if index < 10, NOTE: index value between 1 & 20)
byte 4	Index Ones	Ones digit of stored TLE index

NOTE: Bytes 3 & 4 will contain 7Fh when no valid data is stored for the requested index.

bytes 5-14	Sat Name	10 character satellite name to be associated with index
bytes 15-20	Longitude	Nominal satellite longitude -179.9 to 179.9 (West longitude negative) Left Justify and pad with blanks
bytes 21-22	Inclination	Satellite inclination 0 to 19 Left Justify and pad with blanks
byte 23	Band	RF Band (0-C, 1-Ku, 2-C/Ku, 3-L, 4-X, 5-Ka, 6-S)
byte 24	Ephem	Ephemeris Data Present (0-none, 1-TLE, 2-IESS-412)
bytes 25-29	Pol Offset	Polarization Offset -90.0 to 90.0 negative = counterclockwise Left Justify and pad with blanks
byte 30	ETX	
byte 31	Checksum	

3.12 Write Two Line Element Data Command

This command downloads NORAD Two Line Element (TLE) ephemeris data into the RC3000.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	3Bh	Write TLE Data command code
byte 3	Index Tens	Tens digit of index that TLE set is to be stored (0 if index < 10) NOTE: index value may be between 1 to 10
byte 4	Index Ones	Ones digit of index that TLE set is to be stored
bytes 5-73	TLE Line 1	69 characters (including checksum) of TLE Line 1
bytes 74-142	TLE Line 2	69 characters (including checksum) of TLE Line 2
byte 143	ETX	
byte 144	Checksum	

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	3Bh
byte 3	ETX
byte 4	Checksum

3.13 Read Two Line Element Data Command

This command uploads a stored set of Two Line Element (TLE) data.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	3Ch	Read TLE Data command code
byte 3	Index Tens	Tens digit of stored TLE index (0 if index < 10, NOTE: index value between 1 & 10)
byte 4	Index Ones	Ones digit of stored TLE index
byte 5	ETX	
byte 6	Checksum	

Reply

byte 0	ACK or NAK	
byte 1	address	
byte 2	3Ch	
byte 3	Index Tens	Tens digit of stored TLE index (0 if index < 10, NOTE: index value between 1 & 10)
byte 4	Index Ones	Ones digit of stored TLE index

NOTE: Bytes 3 & 4 will contain 7Fh when no valid data is stored for the requested index.

bytes 5-73	TLE Line 1	69 characters (including checksum) of TLE Line 1
bytes 74-142	TLE Line 2	69 characters (including checksum) of TLE Line 2
byte 143	ETX	
byte 144	Checksum	

3.14 Write IESS Data Command

This command downloads IESS-412 ephemeris data into the RC3000.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	3Dh	Write IESS data command code
byte 3	Index Tens	Tens digit of index that IESS data set is to be stored (0 if index < 10) NOTE: index value may be between 1 to 10
byte 4	Index Ones	Ones digit of index that IESS data set is to be stored

--- TBD

byte TBD	ETX	
byte TBD	Checksum	

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	3Dh
byte 3	ETX
byte 4	Checksum

3.15 Read IESS Data Command

This command uploads a stored set of IESS data.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	3Eh	Read IESS Data command code
byte 3	Index Tens	Tens digit of IESS data index (0 if index < 10, NOTE: index value between 1 & 20)
byte 4	Index Ones	Ones digit of sat data index
byte 5	ETX	
byte 6	Checksum	

Reply

byte 0	ACK or NAK	
byte 1	address	
byte 2	3Eh	
byte 3	Index Tens	Tens digit of IESS data index (0 if index < 10, NOTE: index value between 1 & 10)
byte 4	Index Ones	Ones digit of stored TLE index

NOTE: Bytes 3 & 4 will contain 7Fh when no valid data is stored for the requested index.

---- TBD

byte TBD	ETX
byte TBD	Checksum

3.16 Read Pulse Count Command

The Read Pulse Count command returns the current value of azimuth and elevation pulse or resolver counts. The message format for this command will be:

byte 0	STX	
byte 1	A	where A is the RC3000 address
byte 2	3Fh	3F hex - the read pulse count command code
byte 3	ETX	
byte 4	checksum	

The reply to this query will consist of 15 bytes ...

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	3Fh	the read pulse count command code
bytes 3-7:		Azimuth pulse or resolver count
bytes 8-12:		Elevation pulse or resolver count
byte 13	ETX	
byte 14 :	checksum	

3.17 Extended Device Status Poll Command

The description of the Extended Device Status Poll command (code 40h) is extremely lengthy and is therefore documented at the end of this appendix.

3.18 Remote Locate Command

This command requests the RC3000 to perform a LOCATE operation based on the satellite data supplied. The command is designed to allow an M&C system to simulate entering satellite data manually or selecting a satellite from the user's preset list stored in the RC3000.

NOTE: The M&C system is required to have confidence that the preset list is programmed correctly. The Write Satellite Data command (39h) and Read Satellite Data command (3Ah) may be used to gain confidence that the preset satellite list is correct.

The RC3000 will automatically sequence through the LOCATE operation. Any action that normally requires user action from the front panel will be automatically initiated.

The command contains 37 bytes with the following format:

byte 0	STX	
byte 1	A	RC3000 address
byte 2	41h	Remote Locate command code
byte 3	Preset Flag & Preset Index Tens	

```
7 6 5 4   3 2 1 0
0 1 0 A $ 0 0 0 B
```

A - 1 = perform LOCATE to an indexed satellite from the user preset list stored in the RC3000.
Bytes 5-28 may be left blank.
NOTE: this option is required to reference an inclined orbit satellite that has ephemeris data associated with it.

B - Tens digit of preset satellite index (if applicable)
(0 if index < 10): index value may be between 0 to 19

A - 0 = perform LOCATE to a satellite using name, longitude, inclination and band data supplied in bytes 5-28.

byte 4	Index Ones	Ones digit of preset satellite index (if applicable)
bytes 5-14	Sat Name	10 character satellite name
bytes 15-20	Longitude	Nominal satellite longitude -179.9 to 179.9 (West longitude negative) Left Justify and pad with blanks
bytes 21-22	Inclination	Satellite inclination 0 to 19 Left Justify and pad with blanks
byte 23	Band	RF Band (0-C, 1-Ku, 2-C/Ku, 3-L, 4-X, 5-Ka, 6-S)

bytes 24-28	Pol Offset	Satellite Polarization Offset -90.0 to 90.0 (relative to equatorial plane) negative = counterclockwise Left Justify and pad with blanks **NOTE: Future expansion - polarization offset data is not currently used.**
byte 29	'X'	Receive Polarization Selection H - horizontal V - vertical N - none NOTE: Not applicable if feed type is circular.
byte 30	Position Update	A - Determine mount position (lat/lon/heading) automatically according to how the RC3000 is configured U - Force an update of position via GPS and compass
byte 31-34	Spare Bytes	-for future expansion
byte 35	ETX	
byte 36	Checksum	

The reply to the Remote Locate command will consist of 5 bytes:

byte 0	ACK or NAK	ACK implies that LOCATE operation will be initiated. Progress of the LOCATE operation may be monitored via the Extended Device Status Poll command. NAK implies an error in the supplied satellite data
byte 1	address	
byte 2	41h	
byte 3	ETX	
byte 4	Checksum	

3.19 Remote Store Command

This command requests the RC3000 to perform a STORE operation based on the satellite data supplied.

The RC3000 will automatically sequence through the STORE operation. Any action that normally requires confirmation from the front panel will be automatically initiated. If a particular satellite name has already been STOREd, it's data will be overwritten as a result of the Remote Store command.

NOTE: It is assumed that the satellite has been positively identified and is currently peaked up in azimuth and elevation prior to performing a STORE operation. It is also assumed that Horizontal and Vertical polarization positions have been confirmed.

The command contains 48 bytes with the following format:

byte 0	STX	
byte 1	A	RC3000 address
byte 2	42h	Remote Locate command code
byte 3	Preset Flag & Preset Index Tens	
		`7 6 5 4   3 2 1 0` `0 1 0 A $ 0 0 0 B`
		A - 1 = perform STORE of a satellite defined from the user preset list stored in the RC3000. NOTE: this option is required to reference an inclined orbit satellite that has ephemeris data associated with it.
		A - 0 = perform STORE of a satellite using name, longitude, inclination and band data supplied in bytes 5- 39.
		B - Tens digit of preset satellite index (0 if index < 10): index value may be between 1 to 20
byte 4	Index Ones	Ones digit of preset satellite index
bytes 5-14	Sat Name	10 character satellite name
bytes 15-20	Longitude	Nominal satellite longitude -179.9 to 179.9 (West longitude negative) Left Justify and pad with blanks
bytes 21-22	Inclination	Satellite inclination 0 to 19 Left Justify and pad with blanks
byte 23	Band	RF Band (0-C, 1-Ku, 2-C/Ku, 3-L, 4-X, 5-Ka, 6-S)

bytes 24-28	Pol Offset	Satellite Polarization Offset -90.0 to 90.0 negative = counterclockwise Left Justify and pad with blanks **NOTE: Future expansion - polarization offset data is not currently used.**
byte 29		Polarization Selection C - use calculated H,V values NOTE: Requires that a LOCATE function has been preformed immediately prior to the Remote Store. S - use H,V values supplied in bytes 30-39 H - use current polarization position as Horizontal & calculate Vertical position 90 degrees away V - use current pol position as Vertical & calculate Horizontal position 90 degrees away
bytes 30-34		Horizontal Polarization Position -90.0 to 90.0
bytes 35-39		Vertical Polarization Position -90.0 to 90.0 NOTE: Polarization Selection, Horizontal and Vertical Positions are not applicable if feed type is circular
byte 40		Track Polarization Selects which Polarization position to use when TRACK initiated (applicable to inclined orbit satellites only) H - Horizontal V - Vertical
bytes 41-45	Spare Bytes	-for future expansion
byte 46 ETX		
byte 47	Checksum	

The reply to the Remote Store command will consist of 5 bytes:

byte 0	ACK or NAK	ACK implies that STORE operation will be initiated. NAK implies an error in the supplied satellite data
byte 1	address	
byte 2	42h	
byte 3	ETX	
byte 4	Checksum	

3.20 Write Signpost Data Command

NOTE: This command is only applicable for controllers with the integrated DVB receiver option installed.

This command downloads signpost data into the RC3000's list of user defined signposts.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	43h	Write signpost data command code
byte 3	Index Tens	Tens digit of index that signpost data set is to be stored (0 if index < 10) NOTE: index value may be between 1 to 20
byte 4	Index Ones	Ones digit of index that signpost data set is to be stored
bytes 5-10	Longitude	Nominal satellite longitude -179.9 to 180.0 (West longitude negative) Left Justify and pad with blanks
bytes 11-15	Frequency	10700 to 12750 Left Justify and pad with blanks
bytes 16-20	Symbol Rate	1000 to 30000 Left Justify and pad with blanks
byte 21	FEC	Forward Error Correction Code type 1 - 9 1 = 1 / 2, 2 = 2 / 3, 3 = 3 / 4, 5 = 5 / 6, 6 = 6 / 7, 7 = 7 / 8, 9 = AUTO
byte 22	Polarization	H = horizontal, V = vertical L = LHCP, R = RHCP
bytes 23 - 28	Identification	6 character ID string
byte 29	Priority	0 - 9 relative search priority
bytes 30 - 33	Spare	pad with zeros
byte 34	ETX	
byte 35	Checksum	

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	43h
byte 3	ETX
byte 4	Checksum

3.21 Read Signpost Data Command

NOTE: This command is only applicable for controllers with the integrated DVB receiver option installed.

This command uploads a stored set of signpost data.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	44h	Read Signpost Data command code
byte 3	Index Tens	Tens digit of signpost data index (0 if index < 10, NOTE: index value between 1 & 20)
byte 4	Index Ones	Ones digit of signpost data index
byte 5	ETX	
byte 6	Checksum	

Reply

byte 0	ACK or NAK	
byte 1	address	
byte 2	44h	
byte 3	Index Tens	Tens digit of signpost data index (0 if index < 10, NOTE: index value between 1 & 20)
byte 4	Index Ones	Ones digit of signpost data index

NOTE: Bytes 3 & 4 will contain 7Fh when no valid data is stored for the requested index.

bytes 5-10	Longitude	Nominal satellite longitude -179.9 to 180.0 (West longitude negative) Left Justify and pad with blanks
bytes 11-15	Frequency	10700 to 12750 Left Justify and pad with blanks
bytes 16-20	Symbol Rate	1000 to 30000 Left Justify and pad with blanks
byte 21	FEC	Forward Error Correction Code type 1 - 9 1 = 1 / 2, 2 = 2 / 3, 3 = 3 / 4, 5 = 5 / 6, 6 = 6 / 7, 7 = 7 / 8, 9 = AUTO
byte 22	Polarization	H = horizontal, V = vertical L = LHCP, R = RHCP
bytes 23 - 28	Identification	6 character ID string
byte 29	Priority	1 - 9 relative search priority
bytes 30 - 33	Spare	pad with zeros
byte 34	ETX	
byte 35	Checksum	

3.22 Read Navigation Data Command

This command uploads the current values of navigation data.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	45h	Read Navigation Data command code
byte 3	ETX	
byte 4	Checksum	

Reply

byte 0	ACK or NAK	
byte 1	address	
byte 2	45h	
byte 3	latitude/longitude source	

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 X X X
```

where 'XXX' is ...
001 – currently no lat/lon data
010 – GPS
011 – Manually Entered
100 – User Preset Location List
101 – Saved Position
111 – Remotely Entered

bytes 4-9	latitude	dd.mm(N/S) format
bytes 10-12	reserved	spare bits for additional lat seconds data set to blanks
bytes 13-19	longitude	ddd.mm(E/W) format
bytes 20-22	reserved	spare bits for additional lon seconds data set to blanks
bytes 23-28	reserved	spare bits for altitude data set to blanks
byte 29	true heading source	

```
7 6 5 4   3 2 1 0
0 1 0 0 $ 0 X X X
```

where 'XXX' is ...
001 – Currently no true heading data
010 – Automatically from fluxgate compass
011 – Manually from fluxgate compass
100 – Manually entered
101 – Fix by user
110 – Fix by DVB Receiver
111 – Remotely entered

bytes 30-34	true heading	0.0 – 359.9

byte 35	magnetic variation status		
		0x41h if magvar ready (calculated) 0x40h is magvar not ready	
bytes 36-41	magnetic variation	-99.9 to 99.9 or blanks if magvar not ready (westerly variation negative)	
	byte 42	reserved	spare for time status
	bytes 43-50	reserved	spare for time data
	bytes 51-58	reserved	spare for date data
	byte 59	reserved	spare for platform pitch/roll status
	bytes 60-64	reserved	spare for pitch data
	bytes 65-69	reserved	spare for roll data
byte 70	ETX		
byte 71	Checksum		

3.23 Write Navigation Data Command

This command downloads antenna position data into the RC3000.

byte 0	STX	
byte 1	A	RC3000 address
byte 2	46h	Write Navigation Data command code
bytes 3-7	Latitude	sddmm (+/-degrees/minutes) format minus = South, positive (North) implied, pad with zeros 01234 = 12 degrees 34 minutes N -1234 = 12 degrees 34 minutes S 00123 = 1 degree 23 minutes
bytes 8-13	Longitude	sdddmm (+/-degrees/minutes) format minus = West, positive (East) implied, pad with zeros 001234 = 12 degrees 34 minutes E -01234 = 12 degrees 34 minutes W 000123 = 1 degree 23 minutes E
bytes 14-18	True Heading	123.4 format (000.0 to 359.9) True Heading of mount azimuth 0.0
bytes 19-30	reserved	spare for future expansion (fill with blanks)
byte 31	ETX	
byte 32	Checksum	

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	46h
byte 3	ETX
byte 4	Checksum

NOTE: Valid lat/lon/heading data will trigger an ACK reply. Upon reception of valid data, the RC3000 will overwrite any existing position data and invalidate (DELETE) any STOREd satellites.

3.24 Azimuth/Elevation/Polarization Jog Command (with minimal reply)

This command jogs the antenna in azimuth, elevation or polarization. It is functionally the same command as described in section 3.4 but with a much shorter reply. The command contains 11 bytes and the format of the command is exactly the same as described in 3.4 except that the command code is 47h vs. 33h.

Rather than sending the full status reply, this command only sends the position (at the time the command is received) of the axis to be jogged.

The reply to this query will consist of 12 bytes ...

byte 0	ACK	
byte 1	A	where A is the RC3000 address
byte 2	47h	the command code
byte 3:	"A/E/P"	Axis jogged: A(zimuth), E(levation) or P(olarization)
bytes 4-9	axis position	This field will contain the formatted axis position in the range -180.0 to 180.0. If the analog to digital converter detects an error this field will contain '******'.
byte 10	ETX	
byte 11 :	checksum	

3.25 Remote Keypress Command

NOTE: This command provides the same functionality as the Miscellaneous - Keypad Input sub-command (section 3.7). This command provides a minimal reply compared to the reply from the Miscellaneous command.

This command sends a keypad value to the RC3000. The RC3000 will react to the keypad value as if the corresponding key on the RC3000's front panel was pushed.

Here is the format of the command.

byte 0	STX		
byte 1	A	where A is the RC3000 address	
byte 2	48h	the Remote Keypress command code	
byte 3	key code	30-39h 41-47h	key codes as defined in the following table
byte 4	ETX		
byte 5	checksum		

CODE	KEY
30h	0/Speed
31h	1/Pol CCW
32h	2/N/EL UP
33h	3/Pol CW
34h	4/E/AZ CCW
35h	5
36h	6/W/AZ CW
37h	7/H
38h	8/S/EL DN
39h	9/V
3A-3Fh	-- unused –
41h	Stop/decimal pt.
42h	+/-/BKSP
43h	Mode
44h	Scroll Up/Yes
45h	Scroll Dn/No
46h	Enter
47h	Mode Group Change*

*the 47h key code can be used to initiate a RC3000 mode group change which normally requires the Mode key to be held down for five seconds continuously.

Reply

byte 0	ACK or NAK
byte 1	address
byte 2	48h
byte 3	ETX
byte 4	Checksum

Extended Device Status Poll Command

This command is an extension of the Device Status Poll Command. The reply to this command provides all the information of the Device Status Poll Command along with information about the current mode and state of the RC3000.

The Extended Device Status Poll command consists of 5 bytes with the following format:

byte 0 STX
byte 1 A where A is the RC3000 address
byte 2 40h the extended device status poll query command code
byte 3 ETX
byte 4 checksum

The response to this command will consist of 52 bytes, which will be a combination of ASCII and binary data fields. The binary data will be placed in the lower nibble of a byte whose higher nibble will be initialized to a value that will make the result an ASCII character. The format of the response is:

byte 0 ACK
byte 1 A where A is the RC3000 address
byte 2 40h the extended status poll query command code

bytes 3-44 the same information as contained in bytes 3-44 of the Device Status Poll command

byte 45 Current Mode

This byte contains a value reflecting the present mode of the RC3000:

0-31 (0h-1Fh) Unused to avoid use of control character value

31-127 (20h - 7Fh) current mode value as defined by the mode_name enumeration list at the end of this definition

byte 46 Current State

This byte contains a value reflecting the present state within the current_mode

0-31 (0h-1Fh) Unused to avoid use of control character value

31-127 (20h - 7Fh) current_state value as defined by the enumeration list for the current_mode at the end of this definition

byte 47 Last Exited Mode

This byte contains a value reflecting the previous mode of the RC3000. This data is to be used with the Exit Status byte to determine how the previously accomplished mode terminated.

0-31 (0h-1Fh) Unused to avoid use of control character value

31-127 (20h - 7Fh) current mode value as defined by the mode_name enumeration list at the end of this definition

byte 48 Exit Status

This byte contains a value reflecting the termination status of the previously performed mode.

0-31 (0h-1Fh) Unused to avoid use of control character value

31-127 (20h - 7Fh) exit_status value as defined by the enumeration list for the current_mode at the end of this definition

byte 49 extended azimuth position

For mounts with the ability to generate azimuth position to 0.01 degrees, this byte contains the digit for the one hundredth of a degree. This digit is to be added to the rest of the azimuth position contained in bytes 14-19.

byte 50 extended elevation position

For mounts with the ability to generate elevation position to 0.01 degrees, this byte contains the digit for the one hundredth of a degree. This digit is to be added to the rest of the elevation position contained in bytes 20-25.

bytes 51-55 spare bytes

byte 56:ETX
byte 57:checksum

EXTENDEND STATUS REPLY ENUMERATION LISTS

The following lists define the values to be placed in bytes 45-48 of the Extended Device Status Poll reply.

The lists are provided in the structure of a “C” language enumeration type. The first name in a list is assigned the value 0. The values for subsequent names are incremented by 1. Note that all lists will have placeholder names for values 0 through 31 (0-1Fh). These values will not be used in order to avoid using a control character value.

When applicable, values within lists that only apply to certain mount types or RC3000 options will be noted by “C” style comments (//----).

Names for modes, states or exit_status in lists are provided as descriptions. An M&C system may map a value to any description of their choosing.

The first list provided enumerates values for the RC3000’s operating modes. Values from this list will be used in the Current Mode (byte 45) and Last Exited Mode (byte 47) fields.

Lists for the Current State (byte 46) and Exit Status (byte 48) fields are group together per the applicable RC3000 mode.

```
/*---------- MODE NAMES --------------------*/
enum mode_names {

U0,U1,U2,U3,U4,U5,U6,U7,U8,U9,            // - don't use 0-1F to avoid
conflict with
U10,U11,U12,U13,U14,U15,U16,U17,U18,U19,        // reserved message control
values used by
U20,U21,U22,U23,U24,U25,U26,U27,U28,U29,        // remote control protocol
U30,U31,

MANUAL_MODE,
AUTO_MODE,
POSITION_MODE,
TRUCK_POS_MODE,
TRUCK_HDG_MODE,
LOCATE_MODE,
SPIRAL_REMOTE_MODE,
STORE_SAT_MODE,
TRACK_MODE,
AUX_MODE,
FAIRING_MODE,
POS_CONFIRM_MODE,
DEPLOY_REQUEST_MODE,
HEADING_FIX_MODE,
SETTINGS_MODE,
STOW_MODE,
DEPLOY_MODE,
RECALL_MODE,
REMOTE_MODE,

DEFINE_MODE,
```

```
INIT_MODE,
SETUP_MODE,
PACK_MODE,

RESET_MODE ,
DELETE_MODE ,

CONFIG_MENU_MODE,              /* START OF PROGRAMMING MODES */
ANTENNA_CONFIG_MODE,
AUTOPOL_CONFIG_MODE,
SAT_PRESET_CONFIG_MODE,
TRUCK_PRESET_CONFIG_MODE,
EXPERT_CONFIG_MODE,
RESET_DEFAULTS_CONFIG_MODE,
AZIM_CAL_CONFIG_MODE,
SYS_COMP_CONFIG_MODE,
AZIM_POT_DRIVE_CONFIG_MODE,
ELEV_POT_DRIVE_CONFIG_MODE,
AZIM_PULSE_CAL_CONFIG_MODE,
ELEV_PULSE_CAL_CONFIG_MODE,
POL_POT_DRIVE_CONFIG_MODE,
PULSE_LIMITS_MODE,
TRACK_CONFIG_MODE,
AGC_CONFIG_MODE,
LIMITS_CONFIG_MODE,
TLE_1_CONFIG_MODE,
TLE_2_CONFIG_MODE,

DUMP_FLUXGATE_DATA_MODE,
DUMP_GPS1_NMEA_DATA_MODE,
DUMP_GPS1_RAW_DATA_MODE,
DUMP_GPS2_RAW_DATA_MODE,
DUMP_GPS3_RAW_DATA_MODE,
COMPASS_CAL,
GPS_DIAG_MODE,
AD_DIAG_MODE,
LIMITS_DIAG_MODE,
AGC_OFFSET_MODE,
DIAG_MENU_MODE,
TIMEDATE_DIAG_MODE,
SHAKE_MODE,
AZEL_MODE,
GPSHDG_MODE,

UNDEFINED_MODE

}; /*end of mode_names enumeration */
```

```
//==========================================================
enum simple_mode_states {

O0,O1,O2,O3,O4,O5,O6,O7,O8,O9,                    // - don't use 0-1F to
avoid conflict with
O10,O11,O12,O13,O14,O15,O16,O17,O18,O19,  // reserved message control values
used by
O20,O21,O22,O23,O24,O25,O26,O27,O28,O29,  // remote control protocol
O30,O31,

INITIALIZING_MODE,
WAITING_FOR_USER_INPUT,

LAST_SIMPLE_MODE_STATE } ;

// used for: POSITION_MODE, AUX_MODE, HEADING_FIX_MODE, SETTINGS_MODE,
DEFINE_MODE
//           TRUCK_HDG_MODE, REMOTE_MODE, RESET_MODE, DELETE_MODE, all
configuration modes
//           NOTE: no maintenance modes are instrumented
//==========================================================

enum simple_exit_status {

M0,M1,M2,M3,M4,M5,M6,M7,M8,M9,                    // - don't use 0-1F to
avoid conflict with
M10,M11,M12,M13,M14,M15,M16,M17,M18,M19,  // reserved message control values
used by
M20,M21,M22,M23,M24,M25,M26,M27,M28,M29,  // remote control protocol
M30,M31,

MODE_NORMAL_EXIT,

LAST_SIMPLE_EXIT_CONDITION };


//==========================================================
```

```
enum locate_states      {        // M&C

A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,                      // - don't use 0-1F to
avoid conflict with
A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,  // reserved message control values
used by
A20,A21,A22,A23,A24,A25,A26,A27,A28,A29,  // remote control protocol
A30,A31,                                                  // - this values
placed directly into mode byte


ENTERING_LOCATE_MODE,
INITIALIZING_LOCATE_MODE,
LOCATE_BEGINNING_ANTENNA_DEPLOYMENT,
LOCATE_ANTENNA_DEPLOYMENT_OPENING_FAIRING,       // SWD only
LOCATE_ANTENNA_DEPLOYMENT_ELEV_MOVE,
LOCATE_ANTENNA_DEPLOYMENT_AZIM_MOVE,
LOCATE_UPDATING_DISPLAY,
LOCATE_CALCULATE_MAGVAR_FOR_CALCULATE_MODE,  // VSFX only
LOCATE_CALCULATING_MAGNETIC_VARIATION,
LOCATE_SYNCHRONIZING_SYSTEM_CLOCK_TO_UTC,
LOCATE_WAITING_FOR_LAT_LON,
LOCATE_WAITING_FOR_HEADING,
LOCATE_WAITING_FOR_SAT_DATA,
LOCATE_UNDEFINED_PARAMETER_ERROR,
LOCATE_READY_TO_LOCATE,
LOCATE_AZIMUTH_RANGE_ERROR,  // ** if commanded remotely, action required
LOCATE_ELEVATION_RANGE_ERROR,  // ** if commanded remotely, action required
LOCATE_PERFORMING_MANUAL_SAT_DATA_ENTRY,
LOCATE_PERFORMING_PRESET_SAT_DATA_ENTRY,
LOCATE_PERFORMING_SATLIST_DATA_ENTRY,

LOCATE_WAITING_FOR_POLARIZATION_SELECTION,
LOCATE_WAITING_FOR_CONFIRMATION_TO_CONTINUE,
LOCATE_PITCH_CALCULATION_FIRST_ELEV_MOVEMENT,   // Ephemeris Tracking
LOCATE_PITCH_CALCULATION_AZIM_MOVEMENT,            // Ephemeris Tracking
LOCATE_PITCH_CALCULATION_SECOND_ELEV_MOVEMENT,               // Ephemeris
Tracking
LOCATE_CALCULATING_PITCH,
LOCATE_ROLL_CALCULATION_AZIM_MOVEMENT,             // Ephemeris Tracking
LOCATE_ROLL_CALCULATION_ELEV_MOVEMENT,            // Ephemeris Tracking
LOCATE_CALCULATING_ROLL,                               // Ephemeris Tracking
LOCATE_CALCULATE_NORAD_ELEV_ANGLE,              // Ephemeris Tracking
LOCATE_CALCULATE_NORAD_AZIM_ANGLE,              // Ephemeris Tracking
LOCATE_FIRST_ELEV_MOVEMENT,
LOCATE_POL_MOVEMENT,
LOCATE_AZIM_MOVEMENT,
LOCATE_SECOND_ELEV_MOVEMENT,

LOCATE_BEGINNING_SCAN,
LOCATE_MOVING_TO_INITIAL_AZIMUTH_SCAN_POSITION,
LOCATE_MOVING_TO_INITIAL_ELEVATION_SCAN_POSITION,
LOCATE_PERFORMING_SMOOTH_AZIMUTH_SCAN,
LOCATE_ADJUST_ELEV_DURING_SMOOTH_AZIMUTH_SCAN,
LOCATE_AZIMUTH_STEP_SCAN,
```

```
LOCATE_ADJUST_ELEVATION_DURING_AZIMUTH_STEP_SCAN,
LOCATE_MOVING_TO_SCAN_PEAK,
LOCATE_NO_PEAK_FOUND_MOVING_TO_NOMINAL_AZIMUTH,
LOCATE_SCAN_FINAL_ELEVATION_ADJUSTMENT,
LOCATE_SCAN_WAITING_FOR_EXIT_COMMAND_AFTER_NO_PEAK_FOUND,
BEGINNING_SPIRAL_SEARCH,
SPIRAL_MOVING_TO_NOMINAL_AZIMUTH_STARTING_POSITION,
SPIRAL_SEARCH_STEPPING_CW_IN_AZIMUTH,
SPIRAL_SEARCH_ADJUSTING_ELEV_CW_AZIMUTH_STEP,
SPIRAL_SEARCH_SAMPLING_SIGNAL_AZ_CW_STEP,
SPIRAL_SEARCH_ADJUSTING_AZIMUTH_DURING_UP_ELEVATION_STEP,
SPIRAL_SEARCH_STEPPING_UP_IN_ELEVATION,
SPIRAL_SEARCH_SAMPLING_SIGNAL_EL_UP_STEP,
SPIRAL_SEARCH_STEPPING_CCW_IN_AZIMUTH,
SPIRAL_SEARCH_ADJUSTING_ELEV_CCW_AZIMUTH_STEP,
SPIRAL_SEARCH_SAMPLING_SIGNAL_AZ_CCW_STEP,
SPIRAL_SEARCH_ADJUSTING_AZIMUTH_DURING_DOWN_ELEVATION_STEP,
SPIRAL_SEARCH_STEPPING_DOWN_IN_ELEVATION,
SPIRAL_SEARCH_SAMPLING_SIGNAL_EL_DOWN_STEP,
SPIRAL_SEARCH_NO_PEAK_MOVING_TO_AZIMUTH,
SPIRAL_SEARCH_NO_PEAK_MOVING_TO_ELEVATION,


LAST_LOCATE_STATE } ;  // end of locate_states enumeration

//=========================================================


enum locate_exit_conditions {

B0,B1,B2,B3,B4,B5,B6,B7,B8,B9,                    // - don't use 0-1F to
avoid conflict with
B10,B11,B12,B13,B14,B15,B16,B17,B18,B19,  // reserved message control values
used by
B20,B21,B22,B23,B24,B25,B26,B27,B28,B29,  // remote control protocol
B30,B31,


LOCATE_CANNOT_INITIATE_MOVEMENT_BELOW_DOWN_LIMIT,  // CPS only
LOCATE_ANTENNA_DEPLOYMENT_FAIRING_NOT_CLEAR,  // SWD only
LOCATE_ANTENNA_DEPLOYMENT_ELEV_MOVE_STOP_KEY,
LOCATE_ANTENNA_DEPLOYMENT_ELEV_MOVE_MODE_CHANGED,
LOCATE_ANTENNA_DEPLOYMENT_AZIM_MOVE_STOP_KEY,
LOCATE_ANTENNA_DEPLOYMENT_AZIM_MOVE_MODE_CHANGED,
LOCATE_PITCH_CALC_FIRST_ELEV_MOVEMENT_STOP_KEY,
LOCATE_PITCH_CALC_FIRST_ELEV_MOVEMENT_MODE_CHANGED,
LOCATE_PITCH_CALC_AZIM_MOVEMENT_STOP_KEY,
LOCATE_PITCH_CALC_AZIM_MOVEMENT_MODE_CHANGED,
LOCATE_PITCH_CALC_SECOND_ELEV_MOVEMENT_STOP_KEY,
LOCATE_PITCH_CALC_SECOND_ELEV_MOVEMENT_MODE_CHANGED,
LOCATE_ROLL_CALC_AZIM_MOVEMENT_STOP_KEY,
LOCATE_ROLL_CALC_AZIM_MOVEMENT_MODE_CHANGED,
LOCATE_ROLL_CALC_ELEV_MOVEMENT_STOP_KEY,
LOCATE_ROLL_CALC_ELEV_MOVEMENT_MODE_CHANGED,
LOCATE_FIRST_ELEV_MOVEMENT_STOP_KEY,
LOCATE_FIRST_ELEV_MOVEMENT_MODE_CHANGED,
LOCATE_POL_MOVEMENT_STOP_KEY,
```

```
LOCATE_POL_MOVEMENT_MODE_CHANGED,
LOCATE_AZIM_MOVEMENT_STOP_KEY,
LOCATE_AZIM_MOVEMENT_MODE_CHANGED,
LOCATE_SECOND_ELEV_MOVEMENT_STOP_KEY,
LOCATE_SECOND_ELEV_MOVEMENT_MODE_CHANGED,
LOCATE_FUNCTION_COMPLETED,
LOCATE_INITIAL_AZIMUTH_SCAN_POSITION_STOP_KEY,
LOCATE_INITIAL_AZIMUTH_SCAN_POSITION_MODE_CHANGED,
LOCATE_INITIAL_ELEVATION_SCAN_POSITION_STOP_KEY,
LOCATE_INITIAL_ELEVATION_SCAN_POSITION_MODE_CHANGED,
LOCATE_ADJUST_ELEV_SMOOTH_AZIMUTH_SCAN_STOP_KEY,
LOCATE_ADJUST_ELEV_SMOOTH_AZIMUTH_SCAN_MODE_CHANGED,
LOCATE_AZIMUTH_STEP_SCAN_STOP_KEY,
LOCATE_AZIMUTH_STEP_SCAN_MODE_CHANGED,
LOCATE_ADJUST_ELEV_AZIMUTH_STEP_SCAN_STOP_KEY,
LOCATE_ADJUST_ELEV_AZIMUTH_STEP_SCAN_MODE_CHANGED,
LOCATE_SCAN_FINAL_AZIMUTH_MOVE_STOP_KEY,
LOCATE_SCAN_FINAL_AZIMUTH_MOVE_MODE_CHANGED,
LOCATE_SCAN_FINAL_ELEVATION_ADJUSTMENT_STOP_KEY,
LOCATE_SCAN_FINAL_ELEVATION_ADJUSTMENT_MODE_CHANGED,
LOCATE_FINISHING_WITH_NO_PEAK_FOUND,
SPIRAL_MOVING_TO_AZIM_STARTING_STOP_KEY,
SPIRAL_MOVING_TO_AZIM_STARTING_MODE_CHANGED,
SPIRAL_STEPPING_CW_IN_AZIMUTH_STOP_KEY,
SPIRAL_STEPPING_CW_IN_AZIMUTH_MODE_CHANGED,
SPIRAL_ADJUSTING_ELEV_CW_AZIMUTH_STEP_STOP_KEY,
SPIRAL_ADJUSTING_ELEV_CW_AZIMUTH_STEP_MODE_CHANGED,
SPIRAL_ADJUSTING_AZIM_UP_ELEV_STEP_STOP_KEY,
SPIRAL_ADJUSTING_AZIM_UP_ELEV_STEP_MODE_CHANGED,
SPIRAL_STEPPING_UP_IN_ELEVATION_STOP_KEY,
SPIRAL_STEPPING_UP_IN_ELEVATION_MODE_CHANGED,
SPIRAL_STEPPING_CCW_IN_AZIMUTH_STOP_KEY,
SPIRAL_STEPPING_CCW_IN_AZIMUTH_MODE_CHANGED,
SPIRAL_ADJUSTING_ELEV_CCW_AZIMUTH_STEP_STOP_KEY,
SPIRAL_ADJUSTING_ELEV_CCW_AZIMUTH_STEP_MODE_CHANGED,
SPIRAL_ADJUSTING_AZIM_DOWN_ELEVATION_STEP_STOP_KEY,
SPIRAL_ADJUSTING_AZIM_ELEVATION_STEP_MODE_CHANGED,
SPIRAL_STEPPING_DOWN_IN_ELEVATION_STOP_KEY,
SPIRAL_STEPPING_DOWN_IN_ELEVATION_MODE_CHANGED,
SPIRAL_NO_PEAK_MOVING_TO_AZIMUTH_STOP_KEY,
SPIRAL_NO_PEAK_MOVING_TO_AZIMUTH_MODE_CHANGED,
SPIRAL_NO_PEAK_MOVING_TO_ELEVATION_STOP_KEY,
SPIRAL_NO_PEAK_MOVING_TO_ELEVATION_MODE_CHANGED,
SPIRAL_NO_PEAK_FOUND,

LAST_LOCATE_EXIT_CONDITION } ;

//=========================================================
```

```
enum store_states {

C0,C1,C2,C3,C4,C5,C6,C7,C8,C9,                    // - don't use 0-1F to
avoid conflict with
C10,C11,C12,C13,C14,C15,C16,C17,C18,C19,  // reserved message control values
used by
C20,C21,C22,C23,C24,C25,C26,C27,C28,C29,  // remote control protocol
C30,C31,


STORE_ENTERING_MODE,
STORE_WAITING_FOR_OVERWRITE_CONFIRMATION,
STORE_WAITING_FOR_SAT_DATA_CONFIRMATION,
STORE_WAITING_TO_MODE_OUT_DUE_TO_INCORRECT_SAT_DATA,
STORE_WAITING_FOR_POLARIZATION_ADJUSTMENT,
STORE_CROSS_POL_MOVE,
STORE_WAITING_FOR_TRACK_POL_SELECTION,
STORE_MOVING_POL_FOR_TRACK,


LAST_STORE_STATE } ;

//=========================================================


enum store_exit_conditions {

D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,                    // - don't use 0-1F to
avoid conflict with
D10,D11,D12,D13,D14,D15,D16,D17,D18,D19,  // reserved message control values
used by
D20,D21,D22,D23,D24,D25,D26,D27,D28,D29,  // remote control protocol
D30,D31,


STORE_INVALID_REGION,  // CPS only
STORE_LIST_FULL,
STORE_SAT_NAME_ALREADY_STORED,
STORE_USER_INDICATES_SAT_DATA_INCORRECT,
STORE_CROSS_POL_MOVE_MODE_CHANGED,
STORE_USER_MODED_OUT,
STORE_SAT_DATA_STORED_OK,
STORE_USER_MODED_OUT_AT_TRACK_POL_SELECTION,
STORE_USER_MODED_OUT_IN_TRACK_POL_MOVE,
STORE_PROCEEDING_TO_TRACK_MODE,


LAST_STORE_EXIT_CONDITION };

//=========================================================
```

```
enum track_states {

E0,E1,E2,E3,E4,E5,E6,E7,E8,E9,                    // - don't use 0-1F to
avoid conflict with
E10,E11,E12,E13,E14,E15,E16,E17,E18,E19,  // reserved message control values
used by
E20,E21,E22,E23,E24,E25,E26,E27,E28,E29,  // remote control protocol
E30,E31,


TRACK_ENTERING_MODE,
TRACK_WAITING_FOR_C_OR_KU_SELECTION,
TRACK_INITIALIZING_DISPLAY,
TRACK_DEPLOYING_ELEVATION_AXIS,
TRACK_POSITIONING_POLARIZATION,
TRACK_WAITING_FOR_EXIT_CONFIRMATION,  // LNR-only
TRACK_INITIALIZING_PARAMETERS,
TRACK_STEP_PEAKING,
TRACK_STEP_WAITING_FOR_SIGNAL_TO_RETURN,
TRACK_STEP_IDLE,
TRACK_SEARCH_PERFORMING_SEARCH_PATTERN,
TRACK_SEARCH_MOVING_TO_FOUND_PEAK,
TRACK_SEARCH_WAITING_TO_SEARCH_AGAIN,
TRACK_MANUAL_SEARCH_NOMINAL_ELEV_MOVE,
TRACK_MANUAL_SEARCH_NOMINAL_AZIM_MOVE,
TRACK_MANUAL_SEARCH_ACTIVE,
TRACK_MEMORY_IDLE,
TRACK_MEMORY_PEAKING,
TRACK_MEMORY_REPOSITION,
TRACK_ERROR_CREEP_JAMMED,
TRACK_ERROR_CREEP_LIMIT,
TRACK_ERROR_CREEP_DRIVE,
TRACK_ERROR_PEAK_LIMIT,
TRACK_ERROR_SCALE_FACTOR,
TRACK_ERROR_TRACK_GEO,
TRACK_ERROR_TRACK_SYSTEM,
TRACK_ERROR_TRACK_CHECKSUM,
TRACK_ERROR_UNDEFINED_STATUS,
TRACK_MENU_WAITING_FOR_SELECTION,
TRACK_MENU_VIEW,
TRACK_MENU_MODIFY,
TRACK_TLE_IDLE,
TRACK_TLE_REPOSITION,

TRACK_MANUAL_SEARCH_JOG_AZIM_CCW,
TRACK_MANUAL_SEARCH_JOG_AZIM_CW,
TRACK_MANUAL_SEARCH_JOG_ELEV_DOWN,
TRACK_MANUAL_SEARCH_JOG_ELEV_UP,
TRACK_MANUAL_SEARCH_JOG_POL_CW,
TRACK_MANUAL_SEARCH_JOG_POL_CCW,
TRACK_MANUAL_SEARCH_AUTO_POL_MOVE,
TRACK_MANUAL_SEARCH_IDLE,


LAST_TRACK_STATE };
```

```
//==========================================================
```

```
enum stow_states {

F0,F1,F2,F3,F4,F5,F6,F7,F8,F9,                    // - don't use 0-1F to
avoid conflict with
F10,F11,F12,F13,F14,F15,F16,F17,F18,F19,  // reserved message control values
used by
F20,F21,F22,F23,F24,F25,F26,F27,F28,F29,  // remote control protocol
F30,F31,

STOW_INITIALIZING_MODE ,
STOW_WAITING_FOR_CONTINUE_CONFIRMATION ,
STOW_CONTINUING_OPERATION ,
STOW_MOVING_TO_INITIAL_CW_POSITION ,  // SWD only
STOW_WAITING_TO_CONFIRM_INVALID_STOW_SWITCH ,  // SWD only
STOW_MOVING_TO_AZIM_STOW ,
STOW_SEARCHING_FOR_AZIM_STOW_SWITCH ,
STOW_WAITITNG_CANNOT_FIND_AZ_STOW_SWITCH ,
STOW_MOVING_TO_POL_STOW ,
STOW_SEARCHING_FOR_POL_STOW_SWITCH ,
STOW_WAITING_CANNOT_FIND_POL_STOW_SWITCH ,
STOW_WAITING_OUTSIDE_OF_AZIM_STOW_WINDOW ,  // SWD only
STOW_MOVING_TO_ELEV_STOW ,
STOW_CLOSING_FAIRING ,  // SWD only
STOW_WAITING_FAIRING_CANT_MOVE_ELEV_NOT_AT_STOW ,  //SWD only
STOW_WAITING_FAIRING_NOT_AT_STOW ,  // SWD only

LAST_STOW_STATE    } ;

//=======================================================


enum stow_exit_conditions {

G0,G1,G2,G3,G4,G5,G6,G7,G8,G9,                    // - don't use 0-1F to
avoid conflict with
G10,G11,G12,G13,G14,G15,G16,G17,G18,G19,  // reserved message control values
used by
G20,G21,G22,G23,G24,G25,G26,G27,G28,G29,  // remote control protocol
G30,G31,


STOW_MODE_OUT_FROM_CONTINUE_CONFIRMATION,
STOW_MOVING_TO_INITIAL_CW_POSITION_MOVE_STOPPED,  // SWD only
STOW_MOVING_TO_INITIAL_CW_POSITION_MODE_CHANGE,  // SWD only
STOW_INVALID_AZ_STOW_SWITCH,                           // SWD only
STOW_MOVING_TO_AZIM_STOW_MOVE_STOPPED,
STOW_MOVING_TO_AZIM_STOW_MODE_CHANGED,
STOW_SEARCHING_FOR_AZIM_STOW_SWITCH_MOVE_STOPPED,
STOW_SEARCHING_FOR_AZIM_STOW_SWITCH_MODE_CHANGED,
STOW_CANNOT_FIND_AZ_STOW_SWITCH,
STOW_MOVING_TO_POL_STOW_MOVE_STOPPED,
STOW_MOVING_TO_POL_STOW_MODE_CHANGED,
STOW_SEARCHING_FOR_POL_STOW_SWITCH_MOVE_STOPPED,
STOW_SEARCHING_FOR_POL_STOW_SWITCH_MODE_CHANGED,
STOW_CANNOT_FIND_POL_STOW_SWITCH,
STOW_OUTSIDE_OF_AZIM_STOW_WINDOW,                      // SWD only
```

```
STOW_MOVING_TO_ELEV_STOW_MOVE_STOPPED,
STOW_MOVING_TO_ELEV_STOW_MODE_CHANGED,
STOW_FINISHED_NORMALLY,
STOW_FINISHED_WITH_ELEV_NOT_AT_STOW,                        // SWD only
STOW_FINISHED_WITH_FAIRING_NOT_AT_STOW,               // SWD only

LAST_STOW_EXIT_CONDITION } ;

//=========================================================

enum deploy_states {

H0,H1,H2,H3,H4,H5,H6,H7,H8,H9,                    // - don't use 0-1F to
avoid conflict with
H10,H11,H12,H13,H14,H15,H16,H17,H18,H19,  // reserved message control values
used by
H20,H21,H22,H23,H24,H25,H26,H27,H28,H29,  // remote control protocol
H30,H31,


DEPLOY_INITIAIZING_MODE,
DEPLOY_WAITING_FOR_CONTINUE_CONFIRMATION,
DEPLOY_OPENING_FAIRING,                          // SWD only
DEPLOY_WAITING_FAIRING_NOT_CLEAR,         // SWD only
DEPLOY_MOVING_ELEVATION,
DEPLOY_MOVING_POL,
DEPLOY_MOVING_AZIMUTH,

LAST_DEPLOY_STATE};

//=========================================================


enum deploy_exit_conditions {

I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,                    // - don't use 0-1F to
avoid conflict with
I10,I11,I12,I13,I14,I15,I16,I17,I18,I19,  // reserved message control values
used by
I20,I21,I22,I23,I24,I25,I26,I27,I28,I29,  // remote control protocol
I30,I31,

DEPLOY_MOVING_ELEVATION_MOVE_STOPPED,
DEPLOY_MOVING_ELEVATION_MODE_CHANGED,
DEPLOY_MOVING_POL_MOVE_STOPPED,
DEPLOY_MOVING_POL_MODE_CHANGED,
DEPLOY_MOVING_AZIMUTH_MOVE_STOPPED,
DEPLOY_MOVING_AZIMUTH_MODE_CHANGED,
DEPLOY_FINISHED_NORMALLY,

LAST_DEPLOY_EXIT_CONDITION} ;


//=========================================================
```

```
enum menu_states {

J0,J1,J2,J3,J4,J5,J6,J7,J8,J9,                    // - don't use 0-1F to
avoid conflict with
J10,J11,J12,J13,J14,J15,J16,J17,J18,J19,  // reserved message control values
used by
J20,J21,J22,J23,J24,J25,J26,J27,J28,J29,  // remote control protocol
J30,J31,

MENU_WAITING_FOR_SELECTION,

LAST_MENU_STATE};

//=========================================================


enum menu_exit_conditions {

K0,K1,K2,K3,K4,K5,K6,K7,K8,K9,                    // - don't use 0-1F to
avoid conflict with
K10,K11,K12,K13,K14,K15,K16,K17,K18,K19,  // reserved message control values
used by
K20,K21,K22,K23,K24,K25,K26,K27,K28,K29,  // remote control protocol
K30,K31,

MENU_MODE_NORMAL_EXIT,

LAST_MENU_EXIT_CONDITION} ;


//=========================================================
enum manual_states {

L0,L1,L2,L3,L4,L5,L6,L7,L8,L9,                    // - don't use 0-1F to
avoid conflict with
L10,L11,L12,L13,L14,L15,L16,L17,L18,L19,  // reserved message control values
used by
L20,L21,L22,L23,L24,L25,L26,L27,L28,L29,  // remote control protocol
L30,L31,


MANUAL_INITIALIZING_MODE,
MANUAL_JOG_AZIM_CCW,
MANUAL_JOG_AZIM_CW,
MANUAL_JOG_ELEV_DOWN,
MANUAL_JOG_ELEV_UP,
MANUAL_JOG_POL_CW,
MANUAL_JOG_POL_CCW,
MANUAL_AUTO_POL_MOVE,
MANUAL_IDLE,

LAST_MANUAL_STATE } ;

// manual mode exit_status  - use simple_exit_status

//=========================================================
```

```
enum pos_confirm_states {

N0,N1,N2,N3,N4,N5,N6,N7,N8,N9,                    // - don't use 0-1F to
avoid conflict with
N10,N11,N12,N13,N14,N15,N16,N17,N18,N19,  // reserved message control values
used by
N20,N21,N22,N23,N24,N25,N26,N27,N28,N29,  // remote control protocol
N30,N31,


POS_CONFIRM_INITIALIZING_MODE,
POS_CONFIRM_WAITING_FOR_SAVE_OR_CLEAR_DECISION,

LAST_POS_CONFIRM_STATE } ;

// uses simple_exit_status


//=========================================================

enum fairing_mode_states {

P0,P1,P2,P3,P4,P5,P6,P7,P8,P9,                    // - don't use 0-1F to
avoid conflict with
P10,P11,P12,P13,P14,P15,P16,P17,P18,P19,  // reserved message control values
used by
P20,P21,P22,P23,P24,P25,P26,P27,P28,P29,  // remote control protocol
P30,P31,

FAIRING_WAITING_ELEV_NOT_STOWED,
FAIRING_WAITING_FOR_USER_INPUT,
FAIRING_OPENING_TO_CLEAR,
FAIRING_OPENING_TO_SERVICE,
FAIRING_CLOSING_TO_STOW,
FAIRING_CLOSING_TO_CLEAR,
FAIRING_OPENING,
FAIRING_CLOSING,

LAST_FAIRING_STATE } ;

// uses simple_exit
//=========================================================

enum truck_pos_mode_states {

Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,                    // - don't use 0-1F to
avoid conflict with
Q10,Q11,Q12,Q13,Q14,Q15,Q16,Q17,Q18,Q19,  // reserved message control values
used by
Q20,Q21,Q22,Q23,Q24,Q25,Q26,Q27,Q28,Q29,  // remote control protocol
Q30,Q31,

TRUCK_POS_INITIALIZING_MODE,
TRUCK_POS_WAITING_FOR_MANUAL_PRESET_GPS_SELECTION,
TRUCK_POS_CALCULATING_MAGVAR,  //LNR only
TRUCK_POS_MANUAL_LAT_ENTRY,
```

```
TRUCK_POS_MANUAL_LON_ENTRY,
TRUCK_POS_WAITING_FOR_PRESET_ENTRY,
TRUCK_POS_WAITING_FOR_GPS_ENTRY,

LAST_TRUCK_POS_STATE } ;

// uses simple exit

//=========================================================

enum init_mode_states {

R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,                      // - don't use 0-1F to
avoid conflict with
R10,R11,R12,R13,R14,R15,R16,R17,R18,R19,  // reserved message control values
used by
R20,R21,R22,R23,R24,R25,R26,R27,R28,R29,  // remote control protocol
R30,R31,



INIT_INITIALIZING_MODE,
INIT_WAIITNING_FOR_DEPLOY_CONFIRMATION,
INIT_OPENING_FAIRING,  // SWD only
INIT_FAIRING_NOT_CLEAR_WAITING_TO_MODE_OUT,  //SWD only
INIT_MOVING_ELEV_TO_DEPLOY,
INIT_MOVING_AZIM_TO_DEPLOY,
INIT_GETTING_LAT_LON_FROM_GPS,
INIT_SYNCHRONIZING_TIME_TO_UTC,
INIT_GETTING_HEADING_FROM_COMPASS,
INIT_CALCULATING_MAGVAR,

LAST_INIT_STATE } ;
//=========================================================
enum init_exit_conditions {

S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,                      // - don't use 0-1F to
avoid conflict with
S10,S11,S12,S13,S14,S15,S16,S17,S18,S19,  // reserved message control values
used by
S20,S21,S22,S23,S24,S25,S26,S27,S28,S29,  // remote control protocol
S30,S31,


INIT_MOVING_ELEV_TO_DEPLOY_MOVE_STOPPED,
INIT_MOVING_ELEV_TO_DEPLOY_MODE_CHANGED,
INIT_MOVING_AZIM_TO_DEPLOY_MOVE_STOPPED,
INIT_MOVING_AZIM_TO_DEPLOY_MODE_CHANGED,
INIT_NORMAL_EXIT,

LAST_INIT_EXIT_CONDITION } ;

//=========================================================
```

```
enum setup_pack_mode_states {

T0,T1,T2,T3,T4,T5,T6,T7,T8,T9,                    // - don't use 0-1F to
avoid conflict with
T10,T11,T12,T13,T14,T15,T16,T17,T18,T19,  // reserved message control values
used by
T20,T21,T22,T23,T24,T25,T26,T27,T28,T29,  // remote control protocol
T30,T31,

SETUP_PACK_WAITING_FOR_USER_TO_INITATE_MVOEMENT,
SETUP_PACK_RETRACTING_AZIM,
SETUP_PACK_EXTENDING_AZIM,
SETUP_PACK_RETRACTING_ELEV,
SETUP_PACK_EXTENDING_ELEV,
SETUP_PACK_WAITING_FOR_CONFIRMATION_OF_ENDING_POSITION,

LAST_SETUP_PACK_STATE } ;

//=========================================================
enum setup_pack_exit_conditions {

V0,V1,V2,V3,V4,V5,V6,V7,V8,V9,                    // - don't use 0-1F to
avoid conflict with
V10,V11,V12,V13,V14,V15,V16,V17,V18,V19,  // reserved message control values
used by
V20,V21,V22,V23,V24,V25,V26,V27,V28,V29,  // remote control protocol
V30,V31,


SETUP_PACK_RETRACTING_AZIM_MOVE_STOPPED,
SETUP_PACK_RETRACTING_AZIM_MODE_CHANGED,
SETUP_PACK_EXTENDING_AZIM_MOVE_STOPPED,
SETUP_PACK_EXTENDING_AZIM_MODE_CHANGED,
SETUP_PACK_RETRACTING_ELEV_MOVE_STOPPED,
SETUP_PACK_RETRACTING_ELEV_MODE_CHANGED,
SETUP_PACK_EXTENDING_ELEV_MOVE_STOPPED,
SETUP_PACK_EXTENDING_ELEV_MODE_CHANGED,
SETUP_PACK_NORMAL_EXIT,

LAST_SETUP_PACK_EXIT_CONDITION } ;

//=========================================================
```

```
enum recall_mode_states {

v0,v1,v2,v3,v4,v5,v6,v7,v8,v9,                    // - don't use 0-1F to
avoid conflict with
v10,v11,v12,v13,v14,v15,v16,v17,v18,v19,  // reserved message control values
used by
v20,v21,v22,v23,v24,v25,v26,v27,v28,v29,  // remote control protocol
v30,v31,

RECALL_ENTERING_MODE,
RECALL_NO_SATS_STORED_WAITING_TO_EXIT,
RECALL_WAITING_FOR_USER_TO_SCROLL_THROUGH_LIST,
RECALL_WAITING_FOR_INVALID_DATA_ACKNOWLEDGEMENT,
RECALL_WAITING_FOR_INITIAL_POLARIZATION_SELECTION,
RECALL_PERFORMING_ELEVATION_MOVE,
RECALL_PERFORMING_POLARIZATION_MOVE,
RECALL_PERFORMING_AZIMUTH_MOVE,


LAST_RECALL_STATE } ;




//=========================================================
enum recall_exit_conditions {

w0,w1,w2,w3,w4,w5,w6,w7,w8,w9,                    // - don't use 0-1F to
avoid conflict with
w10,w11,w12,w13,w14,w15,w16,w17,w18,w19,  // reserved message control values
used by
w20,w21,w22,w23,w24,w25,w26,w27,w28,w29,  // remote control protocol
w30,w31,

RECALL_ELEV_MOVE_STOPPED,
RECALL_ELEV_MOVE_MODE_CHANGED,
RECALL_POL_MOVE_STOPPED,
RECALL_POL_MOVE_MODE_CHANGED,
RECALL_AZIM_MOVE_STOPPED,
RECALL_AZIM_MOVE_MODE_CHANGED,
RECALL_SWITCHING_TO_TRACK_MODE,
RECALL_NORMAL_EXIT,


LAST_RECALL_EXIT_CONDITION } ;
```

4.0 Troubleshooting - Remote Control

No Communication between RC3000 and the remote control computer. There are numerous situations that could cause no communication:

1) The address set in the RC3000 is not being used by the remote commands. Check the address in the REMOTE configuration screen and ensure that address is being sent with the commands. Incorrectly addressed commands will be ignored by the RC3000.

2) The baud rate set in the RC3000 is not being used by the remote commands. Check the baud rate in the REMOTE configuration screen and ensure that it is the same as being used by the remote computer. Commands sent at the incorrect baud rate will not be recognized by the RC3000.

3) The remote computer or RC3000 are not both set to RS-232 or RS-422/RS-485. The remote control system should determine whether it is to work in RS-232, -422 or -485 mode. Check the configuration of the remote jumper and the placement of the cable inside of the RC3000, as described in 2.2.11. Also check the cabling between the RC3000 and the remote computer.

4) The RS-422 adapter is not compatible with the RC3000. Occasionally it has been found that a commercially available RS-422 adapter will just not work with the RC3000. To check for this possibility, temporarily mechanize the interface via RS-232 and see if communications is established.

5) The remote computer is not actually transmitting through the intended communication port. To check for this possibility, mechanize a “loop back” right at the communication port of the remote computer. The receive mode of the remote control software should see an exact reflection of the transmitted command.

Unreliable Communications or ACU Reset. There are some situations that may cause the remote control communications to be unreliable (such as a garbled status reply) or in the extreme situation to cause the ACU to reset.

1) The general recommendation is not to send commands (particularly status requests) at a rate greater than once a second.

2) Don't repeatedly ask for "static" information such as navigation or satellite data.

3) Allow a previous command to ACK or NAK before sending another command.