Giant Meterwave Radio Telescope

National Center for Radio Astrophysics

Tata Institute of Fundamental Research

Communication protocol

between Teleset and ABCcom

Description and implementation

Author : Laurent Pommier

Project Supervisor : Prof. Pramesh Rao

Date : 10/01/06

Table of Contents

Illustration Index

1. Introduction

The Giant Meterwave Radio Telescope (GMRT) is a radio telescope located in Khodad, at 80 km from Pune (Maharastra, India). It is composed of an array of 30 antennas spread over distances up to 25 km. Each antenna is 45 m in diameter, and has been designed to operate at a range of frequencies from 50 to 1500 Mhz.

GMRT telescope is a complex assembly of electronic and electro-mechanical subunits. It is divided in different systems, each in charge of a specific task (Local Oscillator, Intermediary Frequency, Front End, Feed Positioning System, Servo, BaseBand and Correlator Systems).
The GMRT Telemetry System is responsible for controlling and monitoring other GMRT systems. From the control room of the central building, an operator uses the Telemetry system to perform operations for a specific observation, and to watch the behavior of the telescope. More precisely he sets in a coordinated manner all various parameters of each GMRT systems, and continuously monitors them during the observation.

Inside the Telemetry system, a chain of various elements connects the central building to each antenna. A long term project for a telemetry improvement is to replace some elements of this chain. In particular a new ABC has been developed: the **ABCcom** software on Linux PC. This software technical report is in ref. [4].
Communicating with this new ABC, another software is developed on the operator end: **Teleset** software on Linux PC.
At this date, ABCcom development is completed. Teleset is more complex, a basement design is implemented (multi-user processes, ABCs communication, shared memory, log files...) but some parts remain (correlator, base band, final user interface). Its actual state is documented in ref. [5].

This document presents the details of the communication protocol between Teleset and ABCcom programs. It first analyzes the structure of messages exchanged on this link. Then it describes the contents of all commands and answers implemented for this communication. This report is thus a complement of ABCcom and Teleset documentation.

2. Packet structure and data format

Data exchanged on the Teleset / ABCcom link are packed according to a specific protocol. Next two paragraphs detail the packet structure in both directions, and the last part explains in which format data are put inside packets according to their type.

2.1. Packets from Teleset to ABCcom

Packets from Teleset to ABCcom follow the format illustrated below. A header is composed of 8 bytes and gives information about the full packet (size, ABC addresses, ...). After can come several command sub packets and finally a check sum byte (CS).

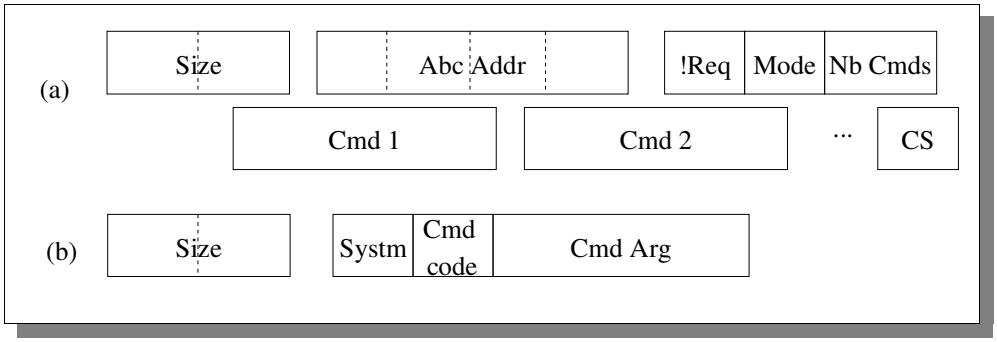

Illustration 1 : (a) Full packet structure from Teleset to ABCcom, (b) Command sub-packet structure

Header:

- ABC addresses are coded on 4 * 8 = 32 bits. A packet can be addressed to several ABC PCs on the link. There are 30 ABC units in GMRT, each of them has a specific address bit among those 32. If a bit is raised in a packet, the corresponding ABCcom will accept this Teleset packet.
- !Req byte indicates whether an answer is expected from ABCcom for this packet. !Req = 0 means all ABC addressed units have to answer.
- Mode byte is 0 in AbcShell mode, and 1 in AbcCom mode (cf. [4]).
- Nb Cmds byte indicates how many command sub-packets are following.

Command sub-packet:

Each command sub packet is dedicated to a specific ABCcom system. A command is coded on one byte, and can have several bytes argument (cf. section 3).

- Systm byte can be:

System	ABC	SV	LO	IF	FE	FPS	EXP
Code	1	2	3	4	5	6	7

Illustration 2 : System code numbers

2.2.Packets from ABCcom to Teleset

Packets from ABCcom to Teleset follow the format illustrated below. A header is composed of 6 bytes and gives information about the full packet (size, ABC address, ...). After can come several answer sub packets and finally a check sum byte (CS).

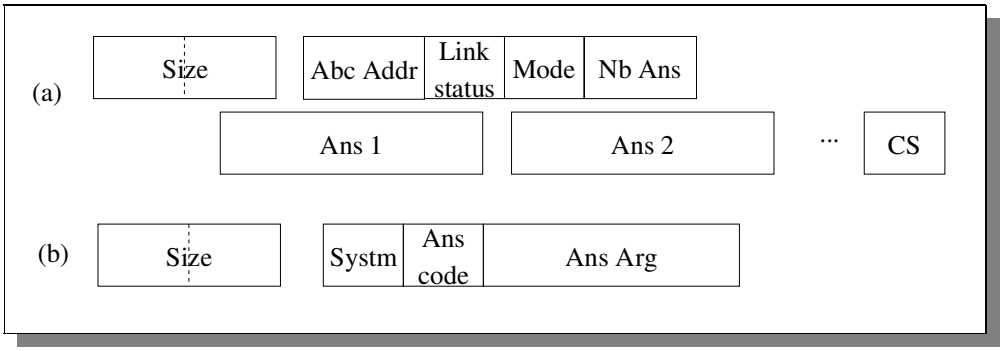

Illustration 3 : (a) Full packet structure from ABCcom to Teleset, (b) Answer sub-packet structure

Header:

- ABC address is one byte, directly the ABC number.
- Link status is a number N. It indicates if there has been an size / CS error in the N^{ieme} previous Teleset packet. N is usually 1 unless a Teleset packet with !Req = 1 has been sent after the wrong packet.
- Mode byte is 0 in AbcShell mode, and 1 in AbcCom mode (cf. [4]).
- Nb Ans byte indicates how many answer sub-packets are following.

Answer sub-packet:

Each answer sub packet comes from a specific ABCcom system. An answer is coded on one byte, and can have several bytes argument (cf. section 4).

2.3.Data Format

A Packet exchanged between Teleset and ABC is implemented in C as a chain of unsigned characters *Msg [i]*. It is stored in a MSG structure along with its current size (cf. ***common.h*** files in ABCcom and Teleset programs). An implementation as a simple string is not possible since a packet can sometimes contain an ASCII delimiter.

Depending on its type, a data inside packets is coded with a specific and fixed format. For instance, the basic data is a integer byte, it is coded as one unsigned character in binary. The table below resumes data types and their respective code in Teleset – ABCcom communication.

Data type	*Link format*		*Programs code*
integer int x	0 < . < 255	Binary	Msg[i] = x;
	can be > 255	Binary 2 bytes: lower - higher	Msg[i] = 0x00ff & x; Msg[i+1] = x >> 8;
float param	ASCII: sign 12 digits (.4) coma ex: “ +234.9200,”		sprintf((char *)&Msg[i], "%+12.4f,", param);
unsigned char x	Binary		Msg[i] = x;
string char* str	ASCII: string and coma ex: “word,”		for (j=0;j<strlen(str);j++) Msg[i++] = str[j]; Msg[i] = ',';
coma	ASCII coma		Msg[i] = ',';

Illustration 4 : Data format table

With respect to this format protocol, ABCcom and Teleset programs pop data from a received packet, and push data into a packet to be transmitted. Programs must then handle the same type of data at the same place inside a packet.
In ABCcom, Combase is the class coding those pop and push functions (cf. ***combase.h*** file, ref. [4]). In Teleset, pop functions are implemented in RxPop class and push functions are in CmdBase class (cf. ***decode.h*** and ***cmdbase.h*** files, ref. [5]).

☞*Notes:*

- *Pop and push functions are C++ polymorphic functions. Therefore to avoid confusion, one byte Msg[i] is coded as an unsigned character and strings are treated only as chain of signed characters .*
- *Since some packets are transmitted by ABCcom directly to SSC or a MCM, this format protocol is also convenient to enter directly data in SSC / MCM communication format (ref. [1] and [3]).*

3. Telemetry commands protocol

In the communication between GMRT control room and antennas, the telescope operator enters telemetry commands with Teleset program which sends them to one or several ABCcom PCs.
As noticed in the previous section, a command is specific to one antenna system. It is coded on one byte followed by some arguments. This chapter resumes all telemetry commands implemented until now in Teleset and ABCcom programs. An exhaustive list is given in Appendix A.

By convention, a command code N is an even numbers, N+1 being the corresponding answer code number. For each system, 0x00 code is for a 'state' command, and 0x02 for a 'askparam' command. A state command asks the current state of a system (operation status, system answers...), and a askparam command asks the current parameter values of a system.

3.1.ABC commands

These commands target ABC system. They are composed in CmdAbc class, ***cmdbase*** Teleset files, and decoded in Comabc class, ***abccom*** ABCcom files.

Command code and name		***Argument***	***Description***
00	state	--	Ask all ABCcom Systems state
02	askparam	--	Ask ABC System parameters
04	settime	Time and date string ex: “Wed Nov 30 01:35:57 IST 2005”	Set time and date in ABCcom PC
06	exit	ABCcom number (1byte integer)	ABCcom exits.
08	startlocal	--	Start ABCcom local Teleset mode
0a	stoplocal	--	Stop ABCcom local Teleset mode
0c	startexp	Valid MCM number (1byte integer)	Start MCM Expert in ABCcom
0e	stopexp	--	Stop MCM Expert in ABCcom

Illustration 5 : ABC System commands

3.2.Servo commands

These commands target Servo system. They are composed in CmdSsc class, ***cmdbase*** Teleset files, and decoded in Comssc class, ***combase*** ABCcom files.

Command		*Argument*	*Description*
00	state	--	Ask Servo System state
02	askparam	--	Ask Servo System parameters
04	trkon	Source RA and DEC (floats)	Start tracking mode in Servo System
06	trkoff	--	Stop Servo tracking
08	trkset	ABC no, AZ offset, EL offset, Latitude, Longitude,[1]	Set Loading parameters for one antenna
0a	trkaz	OutTrack integer 0 or 1	Set AZ OutTrack flag (Setting parameter)
0c	trkrate	RA rate, DEC rate, Time ref (floats)	Set rate Setting parameters
0e	cmd	SSC command [2]	Send direct command to SSC

[1] ABC no is an integer entered by the operator, rest are floats from ***trkset.ant*** file.
[2] cf. [1] for SSC commands. Programs take care of coma positions that are slightly different in this protocol.

Illustration 6 : Servo System commands

3.3. Common commands to LO / IF / FE

These commands target either LO, IF or FE system. They are composed in CmdSys class, ***cmdsystm*** Teleset files, and decoded in Comsys class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
00	state	--	Ask System state
02	askparam	--	Ask System parameters
04	reboot	--	Reboot system
06	setnew	N parameters (floats) [1]	Set new Setting parameters
08	doset	--	Redo the last settings
0a	domon	--	Monitor System
0c	montime	time interval: 2-bytes integer	Enter interval time for auto operations

[1] N depend on the system. Those parameters are taken from ***set1*** file.

Illustration 7 : LO / IF / FE common commands

3.4.LO specific commands

These commands target LO system. They are composed in CmdLo class, ***cmdsystm*** Teleset files, and decoded in Comlo class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
0e	setaage	--	Do the 'setaage' setting
10	setpiche	--	Do the 'setpiche' setting

Illustration 8 : LO specific commands

3.5.IF specific commands

These commands target IF system. They are composed in CmdIf class, ***cmdsystm*** Teleset files, and decoded in Comif class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
0e	loadGA	Abc no (integer), 88 float parameters [(1)]	Send IF loading parameters
10	setGA	4 float parameters	Set post gain and pre att. (ch1 and ch2)

[(1)] Those parameters are post gains and pre attenuations for IF system, taken from ***ifload_i_ant*** file.

Illustration 9 : IF specific commands

3.6.FE specific commands

These commands target FE system. They are composed in CmdFe class, ***cmdsystm*** Teleset files, and decoded in Comfe class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
0e	seton	--	Switch FE system on
10	setoff	--	Switch FE system off
12	setNGon	--	Switch Noise calibration on
14	setNGoff	--	Switch Noise calibration off
16	setNG50	--	Switch Noise calibration on at 50%
18	setNG25	--	Switch Noise calibration on at 25 %

Illustration 10 : IF specific commands

3.7.FPS commands

These commands target FPS system. They are composed in CmdFps class, ***cmdbase*** Teleset files, and decoded in Comfps class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
00	state	--	Ask FPS System state
02	askparam	--	Ask System parameters
04	init	--	Init FPS system
06	loadcnt	ABC no, Encoder count parameters (1)	Load counter positions
08	mvpos	feed parameter (float format)	Move FPS to corresponding loaded count
0a	montime	time interval: 2-bytes integer	Enter interval time for auto operations
0c	cmd	FPS command (2)	Enter direct command to FPS

(1) ABC no is an integer entered by the operator, rest are floats from ***fpspos.ant*** file.

(2) The 1st byte is the FPS command number, transmitted along with corresponding arguments to ABCcom which will form packets according to FPS protocol (cf. [2]).

Illustration 11 : FPS commands

3.8.Expert commands

These commands target EXP system. They are composed in CmdEXP class, ***cmdbase*** Teleset files, and decoded in Comexp class, ***comsys*** ABCcom files.

Command		*Argument*	*Description*
00	state	--	Ask EXP System state
04	cmd	MCM command (1)	Enter direct command to MCM

(1) The 1st byte is the MCM command number, transmitted along with corresponding arguments to ABCcom which will form packets according to MCM protocol (cf. [3]).

Illustration 12 : EXP commands

4. Telemetry answers protocol

Once Teleset has sent a telemetry command message (with !req = 0) to some ABC PCs, each of them has to return a telemetry answer message. ABCcom forms a message packet compliant to the structure in Illustration 3, with one or several answer sub-packets.
An answer sub-packet corresponds to a specific command sub-packet. Its code is then N+1 when the command code was N. They both deal with the same system whose code byte is at the beginning of their sub-packets. This chapter describes answers code and arguments used for this Teleset – ABCcom communication.

As seen in the previous section, 0x00 and 0x02 are respectively codes of 'state' and 'askparam' commands, for every system. Corresponding state and askparam answers are detailed in next paragraphs. To know about ABCcom and its systems evolution, Teleset regularly sends a state command to ABC PCs used for the current telescope observation.
Rest of commands most of time corresponds to complex operations that can take some time to get executed. For this reason, ABCcom just return for them an acknowledgment answer. The operator will come to know his command effects throughout following state answers.

4.1.Acknowledgment answer

For a command with a code N > 0x02, ABCcom returns an acknowledgment answer with N+1 code.
An acknowledgment answer also has a 1-byte argument: 1 if the command format was correct, 0 if not. This part of the code is in ***abccom*** files for ABCcom program, and ***decode.h*** file for Teleset.

4.2.Askparam answer

For an 'askparam' command (code 0x02), ABCcom returns an askparam answer. Its code byte is then 0x03, and its arguments are:

- 1-byte N: number of Setting parameters
- N float parameters
- 1-byte M: number of loading parameters
- M float parameters

Parameters of each system are described in ref. [4]. In ABCcom, askparam answers are coded in ***comsys*** and ***combase*** files. In Teleset, the decoding of these answers is implemented in ***decode.h*** file.

4.3.State answers

For a 'state' command (code 0x00), ABCcom returns an state answer. Its code byte is then 0x01, and its arguments depend on the targeted system. Next paragraphs detail state answer arguments system after system.

4.3.1.ABC state answer

The ABC system state answer contains just one argument:

- ABC PC time (string format)

After receiving an ABC state command, ABCcom composes an answer message not only with ABC state answer sub-packet, but also with the state answer sub-packet of every other system configured in ABCcom. Thus the answer message contains the entire state of ABCcom systems. This command is the one sent automatically by Teleset every few seconds.
These answers are composed in ***abccom*** files of ABCcom program. In Teleset, they are decoded in ***decode*** files.

4.3.2.Servo state answer

The Servo State answer consists in several answer sub-packets. Their answer code are all 0x01, they differ by their (first) arguments. When a Servo state command is received, the return message contains:

(i) a track sub packet, when some changes occurred in Track object
(ii) SSC event sub packets, if SSC has sent any.
(iii) SSC answer sub packets, when SSC commands have been sent.
(iv) SSC time outs and link error sub packets, if any.
(v) SSC display answer sub packets, last 3 replies from SSC to ABCcom automatic display commands.

Arguments of SSC time-out and link error sub-packets:

- 0x00
- link byte: 0x00 – link timeout, 0x01 – command timeout
- command byte: 0 – command timeout, else command code
- code byte: 0x00 – link timeout, 0x15 – link NAK, 0x01 – operational command timeout, 0x03 – display command timeout

Arguments of SSC event, answer and display answer sub-packets:

- data code, from SSC packet according to protocol ref. [1].

☞*Note: SSC data code is a SSC packet without the 5 control codes (DLE, STX, data, DLE ETX, BCC). Data code always starts by 2 bytes that can be* 0x01*,* 0x02 *or* 0x03 *for the different transmitter and receiver state machine.*

Arguments of Track sub-packets:

- 0x04 (to differ from other servo state sub packets which can start with 0..3)
- then same arguments than for an 'askparam' answer

These packets are composed in ***servo*** files of ABCcom, and decoded in ***decssc*** files of Teleset.

4.3.3.LO / IF / FE state answer

Those systems are implemented on the same model, their state answers are similar in their structure and contain the following arguments:

- 1-byte: N number of MCM units in the system, then for each of them (N=1 or 2):
    - 1-byte: MCM address
    - 1-byte: total number of time outs since ABCcom started
- 4 command state bytes (CmdState[] for monitor, set, reboot, and syscmd)
- 1 system command byte (CmdSys)
- 1 system state byte (StateSys)
- Last monitoring time (Montime string), if not composed of eight 0 then:
    - 1-byte error checking (MonResult), if not 0 then:
        - 2-bytes DataPlus length
        - DataPlus.Msg[] bytes (error code)
    - 1-byte: N, number of voltages
    - N voltages values (float format)

The nature of those arguments is detailed in ref. [4]. These packets are composed in ***comsys*** files of ABCcom, and decoded in ***decode*** files of Teleset.

4.3.4.FPS state answer

The FPS system state answer argument are:

- 1-byte: N number of MCMs in the system (N=1)
- 1-byte: MCM address (14)
- 1-byte: total number of time outs since ABCcom started
- 3 command state bytes (CmdState[] for init, mvpos, and directcmd)
- Last monitoring time (Montime string), if not composed of eight 0 then:
    - 2-byte EncCount
    - 1-byte Rpm
    - 1-byte AnsStatus code
    - 1-byte FpsStatus code
- 2-bytes DataPlus length
- DataPlus.Msg[] bytes (2^{nd} logical packet of last FPS answer)

The nature of those arguments is detailed in ref. [4]. These packets are composed in ***comsys*** files of ABCcom, and decoded in ***decfps*** files of Teleset.

4.3.5.EXP state answer

The EXP system state answer argument are:

- 1-byte: N number of MCMs in the system (N=1)
- 1-byte: MCM address
- 2-bytes DataPlus length
- DataPlus.Msg[] bytes (MCM answer)

The nature of those arguments is detailed in ref. [4]. These packets are composed in ***comsys*** files of ABCcom, and decoded in ***decexp*** files of Teleset.

5. Shell mode protocol

The Shell mode is when Teleset and ABCcom programs do not perform their telemetry tasks. Instead, Teleset communicates with just one ABCcom to transfer some files, or to remotely execute one Linux command in an ABC PC shell terminal. TeleShell is the shell mode main object of Teleset program, and AbcShell is the one of ABCcom.
The protocol for Shell mode communication is very close to the one used in normal telemetry mode. Headers are the same, with Mode byte equal to 0. Shell mode packets are limited to one sub-packet. A check sum byte ends the message. Next paragraphs describe command and answer sub-packets in shell mode communication.
The shell mode communication is implemented in ***teleshell*** files on Teleset side, and in ***mainshell*** files on ABCcom end. It is described in ref. [4] (part 7.1), and in ref. [5] (Appendix B).

5.1. Special operations

Some TeleShell commands are addressed to AbcShell itself. Their packets are composed of:

- 1-byte: ASCII letter 's' (for system)
- N-bytes: ASCII operator text command.

In this case, the operator text is one of the following:

operator text	*description*
put filename	Transfer file from Teleset to ABCcom in source directory
get filename	Transfer file from ABCcom to Teleset in source directory
exit	Exit TeleShell, an exit command is sent to ABCcom
newabc	Exit current ABCcom and restart with new one

Illustration 13 : AbcShell commands

For the two lasts, AbcShell sends an answer packet:

- 2-bytes: ASCII letter 'se' (for system end)
- ASCII text 'exit rx.' or 'newabc rx.'.

After a 'put filename' operation, TeleShell sends several messages until the file is fully transmitted:

- 2-bytes: ASCII 'fp' (for file process)
- maximum number of bytes: file data

Or:

- 2-bytes: ASCII 'fe' (for file end)
- N bytes: last file data, (can be empty)

AbcShell can send an interruption message if an error occurs:

- 2-bytes: ASCII 'fi' (for a file interruption)
- ASCII error message ('format error' or 'time out')

Or after getting a 'fe' command packet, AbcShell sends a final acknowledgment:

- 2-bytes: ASCII 'se' (for system end)
- ASCII text 'file ok.'

Roles are simply inverted in a 'get filename' operation, but without final acknowledgment or error message in interruptions since the intelligence is on Teleset end.

5.2.Linux Shell Terminal operations

When a command is not recognized as one of those described above, it is supposed to be a linux shell terminal command:

- 1-byte: ASCII letter 's' (for system)
- N-bytes: ASCII operator text command.

AbcShell executes the text command faithfully in a Shell terminal. It then sends several messages as:

- 2-bytes: ASCII 'sp' (for system process)
- maximum number of bytes: answered data

Or:

- 2-bytes: ASCII 'se' (for system end)
- N bytes: last answered data, (can be empty)

The answered data is either 'cmd unknown' if the execution failed, or the faithful Linux shell text answer. Teleset can interrupt the transmission if an error occurs:

- 2-bytes: ASCII 'si' (for a file interruption)

☞*Note: This answer process is similar to 'get file' since AbcShell stores the shell answer in* **shellout** *file before sending it.*

6. Conclusion

The Telemetry System is a complex system of the Giant Meterwave Radio Telescope. Its function is to configure every GMRT components according to the operator plans and to control and monitor them regularly during all the radio observation process.

A current project to improve GMRT telemetry is to replace ABC and CEB components by a new generation of units. These units are mainly Linux PCs running two softwares which have been developed in laboratories since last 2 years: ABCcom and Teleset. A new communication protocol has been implemented between those two programs, it defines which data is exchanged on the link and under which format.

Data packets have been designed in a worry of efficiency and simplicity. Indeed it is very important to have a fast link between those units in order to improve the overall telescope performances. Packets also had to follow an organization easily usable by both program applications.

This report is a complement to the ABCcom [4] and Teleset [5] software documentations.

References and label index

References:

[1] BARC, *Servo software BARC: SSC interface details*, April 1991.

[2] Mukund Gadgil, *Feed Positioning System Software*, April 1992, GMRT technical report.

[3] Laurent Pommier, *Mcmcom program, a Linux PC / MCM communication*, September 2004, GMRT technical report.

[4] Laurent Pommier, *ABCcom software of GMRT Telemetry System*, January 2006, GMRT technical report.

[5] Laurent Pommier, *Teleset software of GMRT Telemetry System* , March 2006, GMRT technical report.

Index table:

Label	***Full Name***
ABC	Antenna Based Computer
CS	Check Sum byte
COMH	Communication Handler
EXP	Expert system
FE	Front End system
FILO	First In Last Out list
FPS	Feed Positioning System
GMRT	Giant Metrewave Radio Telescope
IF	Intermediary Frequency system
LO	Local Oscillator system
MCM	Monitor and Control Module
PIU	Plug In Unit
SSC	Station Servo Computer

Appendix A: Table of Teleset commands

To Teleset	***To AbcCom***	***To Servo***			
• Defabc • Showabc • Noans • Ncmd • Exit • Adduser • Rmuser	• State • Askparam • Settime • Exit • Startlocal • Stoplocal • StartExpert • StopExpert	• State • Askparam • Trkon • Trkoff • Trkset • Trkaz • Trkrate	*Direct Commands:*		
		• Coldstart • Position • Track • Hold • Stop	• Close • Stow • Stow release • Abort • Reset HW	• Read Angles • Read Anl Vars • Read Dig Vars • Read setparameters • Read Antenna status	• Read Error status • Set time day • Set stow angles • Set S/W Hi limit • Set S/W Lo limit • Set windvel limit
To LO / IF / FE	***only to LO***	***To FPS***			
• State • Askparam • Reboot • Setnew • Doset • Domon • Montime	• Setaage • Setpiche	• State • Askparam • Init • Loadcounts • Mvpos	*Direct Commands:*		
		• Null cmd • Set turning point • Set ramp down cnt • Set lower rpm limit • Set brake down diff • Set ramp up cnt • Set stop time cnt	• Set max PWM cnt • Set max angle • Set min angle • Read turning point • Read ramp down cnt • Read lower rpm limit • Read brake cnt diff	• Read ramp up cnt • Read stop time cnt • Read max PWM cnt • Read max angle • Read min angle • Run to calibrate	• Free run • Run to preset • Run to fine tune • Password run • Reboot • Stop
only to IF	***only to FE***	***To Expert***			
• LoadGA • SetGa	• Seton • Setoff • SetNGon • SetNGoff • SetNG50 • SetNG25	• State • Askparam	*Direct Commands:*		
		• Set MCM Address • Null cmd • Set Idle Mode • Set Scan Mode • Set Mean Mode	• Set Anl Mask • Set Dig Mask 16b • Set Dig Mask 32b • Set Dig Mask 64b • Read Anl Mask	• Read Dig Mask 16b • Read Dig Mask 32b • Read Dig Mask 64b • Read Version • Read Mode	• Reboot • Feed Mcm (old) • Feed Mcm (new) • Fe Box monitor • Common Box monitor