Signal Delay Unit TBD2K manual

Issue 1.0
2016/12/01

TABLE OF CONTENTS

Change Record

Issue	Date	Section/ Paragraph Affected	Reasons / Remarks	Responsible
1	30/11/2016	All	Created	Ziegleder

Scope

This document is the user and service manual of the signal delay unit TBD2K. The first part is the user manual, which describes the hardware, software and how to use it. This part is addressed to the user, which has to setup and operate the device.
The second part is the service manual, which describes advanced configurations and calibration methods. This is addressed to experienced technicians, which have to commission, re-configure or re-calibrate the device. The information in this part have to be handled with care, misuse could harm hardware or humans.

1 Literature

The following list shows documents, which help to understand and operate the device.

No.	Author, Title	Number & Issue
AD 1		
AD 2		
AD 3		
AD 4		
AD 5		
AD 6		
AD 7		
AD 8		
AD 9		
AD 10		
AD 11		
AD 12		
AD 13		
AD 14		
AD 15		
AD 16		
AD 17		

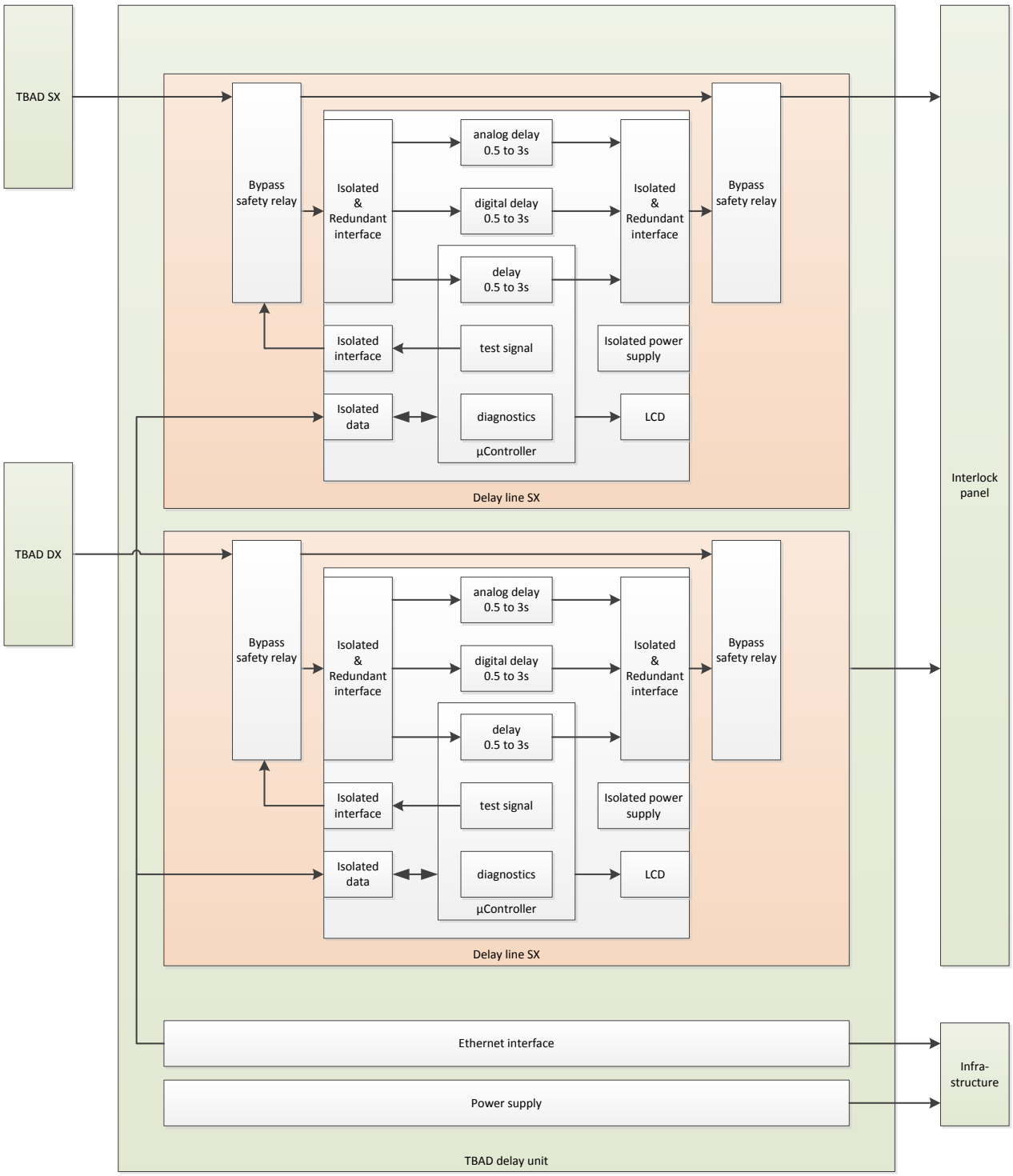
2 Overview

The signal delay unit TBD2K is an Ethernet featured 2 channel digital delay unit. It is designed to delay the falling edge of a 5V digital signal for up to 3.25s with an accuracy of 0.01s.

Two independent delay lines (SX/DX) are built in one enclosure with one common remote interface and power supply. The remote interface cannot influence the delays, but it can monitor, bypass and test them. To prevent the risk of a false-on event, every delay line contains 3 independent delays. An analog, a crystal oscillator based timer and a microcontroller based timer. All three delays are combined to a single output signal. The fastest delay "wins" and will trigger the falling edge of the output signal.

A LCD and a push button are used to show the status locally, a bright white LED is on while the interlock signal is delayed.

In power off conditions or in test mode the delay lines are bypassed by a safety relay. The signal output stage is designed to shut down in case of a malfunction of this relay.



2.1 Technical specifications

3 Setting up

3.1 Electrical connections

4 Remote operation

The unit is equipped with an Ethernet interface. The following chapters will describe the configuration and the command set.

4.1 Configurator

The configurator is a simple configuration tool, which is accessible with the serial interface of the PCD (Debug port). It's used to configure the Ethernet interface and some basic parameters.

The Ethernet interface requires several parameters configured:

- IP address
- Network mask
- Gate address
- Port
- MAC address

Additionally, the configurator allows to configure:

- Debug mode
- Device diagnostics
- Configurator timeout

When the debug mode is enabled, the controller will transmit status information using the serial interface.
The available diagnostic modes are:

Parameter value	Mode
0	All diagnostics off
1	Do a system check once during startup
2	Check device status permanently
3	Do a system check once during startup and check device status permanently

The configurator supports the following keywords:

Keyword	Description
SHOW	Show the current configuration.
IP=XXX.XXX.XXX.XXX;	IP address
GATE=XXX.XXX.XXX.XXX;	Gate address
MASK=XXX.XXX.XXX.XXX;	Network mask
PORT=XX;	Port
MAC=XX.XX.XX.XX.XX.XX;	MAC address
DIAG=X;	Device diagnostic
DEBUG=X;	Debug mode
TIMEOUT=X;	The time in milliseconds you have to start the configuration
PIN=X;	EEPROM protection PIN number

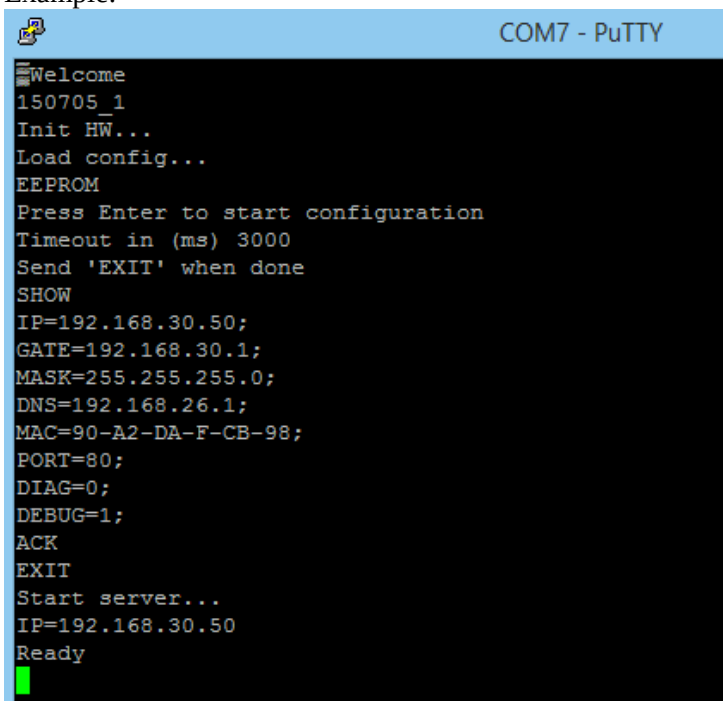
Be aware of the parameter syntax. The parameters have to be terminated by a semicolon.
Caution: MAC, DIAG, DEBUG, PIN and TIMEOUT are usually preconfigured and shall not be modified.

How to start the configuration:

- Connect the power supply controller to a serial port.
- Restart the controller either by power cycling, or by pressing the reset button.
- Wait for the message "Press Enter to start configuration"

- Configure device by using the keywords descripted above
- Send “EXIT” to leave the configuration mode. The firmware will start up the interface.

Example:



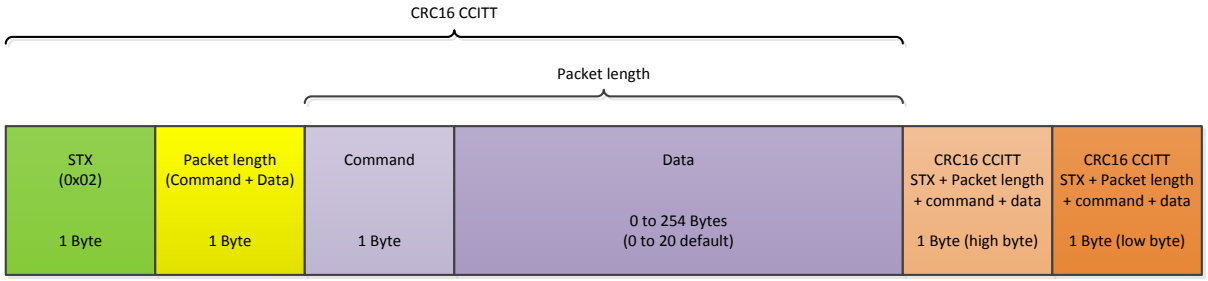
4.2 Remote control

4.2.1 Data protocol

The communication protocol is simple, machine friendly and uses CRC16 to detect transmission errors, check validity and to annoy human users. Depending of the used computer architecture (Big- or Little endian) and CRC library, the CRC-CCITT (0xFFFF) might have to be substituted by CRC-CCITT (XModem) with start value 0xFFFF. Use the examples to verify your calculations.
Commands, which do not return any data, will return Acknowledge ACK: 0x020106F10B
If a command was faulty, the return code is “not acknowledge” NAK: 0x020115D359
Commands, which return data, use the protocol. Command byte is identical to the received command, data is command specific.

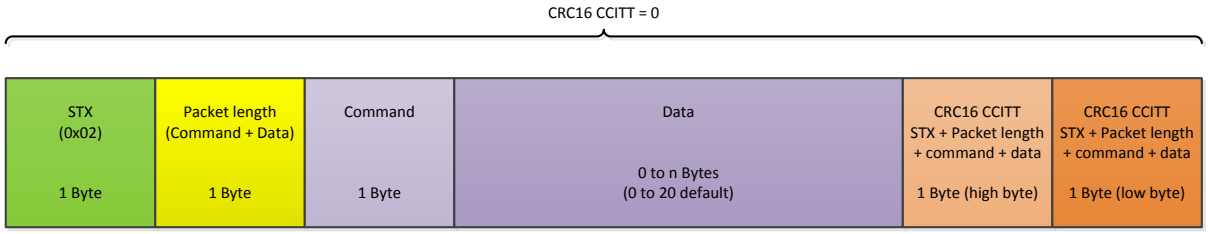
4.2.2 Encode

The CRC-CCITT (0xFFFF) checksum of the package built from STX, packet length, command and the data block is concatenated with high byte first and low byte last.



4.2.3 Decode

The CRC-CCITT (0xFFFF) checksum of the complete received package has to be 0.



The current PCD firmware supports a total datagram length of 50 bytes. Everything else will be discarded.

4.2.4 Examples

Take the test command from above to understand the protocol:
0x0201F16EF3

02	ASCII code STX
01	Packet length:1. Only 1 command will follow.
F1	This command returns ACK, and does nothing else.
6EF3	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

The return value:
0x020106F10B

02	ASCII code STX
01	Packet length:1. Only 1 byte will follow.
06	ASCII code ACK
F10B	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

More examples:
The command F0 will return ASCII NAK:
0x0201F07ED2 -> 0x020115D359

To test and verify the CRC calulations use this online calculator:
http://www.lammertbies.nl/comm/info/crc-calculation.html
There's also a suitable ANSI C/C++ CRC library available:
http://www.lammertbies.nl/comm/software/

4.2.5 Test commands

The following commands are implemented to test the interface:

F0
Returns NAK. Basic communication test.
0x0201F07ED2 -> 0x020115D359

F1
Returns ACK. Basic communication test.
0x0201F16EF3 -> 0x020106F10B

F2
Echo the received data. Basic communication test.

Example:
0x0203F2BEEF72A2 -> 0x0203F2BEEF72A2

02	ASCII code STX
03	Packet length: 1 Command and 2 byte data.
F2	command
BE	Data byte 1
EF	Data byte 2
72A2	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

F3

Converts a received floating number to a string. This is to check if the byte order of the host system matches the order used by the firmware.

Example:

0x0206 F300 79E9 F642 1EA2 -> 0x0209 F300 3132 332E 3435 36EF 07

02	ASCII code STX
06	Packet length: 1 Command and 5 byte data.
F3	Command
00	This value will be echoed. In typical application this position is used for function identifiers, sub-commands etc.
79E9 F642	123.456 single precision floating point value casted into bytes.
1EA2	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

02	ASCII code STX
09	Packet length: 1 Command and 8 byte data.
F3	Repetition of the received command
00	This value will is echoed. In typical application this position is used for function identifiers.
3132 332E 3435 36	123.456 in ASCII code
EF 07	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

F4

Returns the floating number 123.456. This is to check if the byte order of the host system matches the order used by the firmware.

0x0201 F43E 56 -> 0x0205 F479 E9F6 423A 3F

02	ASCII code STX
01	Packet length:1. Only 1 byte will follow.
F4	Command
3E 56	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

02	ASCII code STX
05	Packet length: 1 Command and 4 byte data.
F4	Repetition of the received command
79E9 F642	123.456 single precision floating point value casted into bytes.
3A3F	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

F7

Returns installed firmware version as a string.

0x 0201 F70E 35 -> 0x 0209 F731 3531 3132 345F 31F9 D1

02	ASCII code STX
01	Packet length:1. Only 1 byte will follow.
F7	Command
0E 35	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

02	ASCII code STX
05	Packet length: 1 Command and 4 byte data.
F7	Repetition of the received command
31 35 31 31 32 34 5F 31	“151124_1” in ASCII code
F9D1	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

4.3 Commands

The commands are bundled into several groups, depending of their functionality and abstraction level.
Every command will respond in one of 3 possible ways:
Commands, which do not return any data, will return Acknowledge ACK: 0x020106F10B
If a command was faulty, the return code is “not acknowledge” NAK: 0x020115D359
Commands, which return data, use the protocol. Command byte is identical to the received command, data is command specific.

4.3.1 Analog interaction (A)

4.3.1.1 A0 – Analog measurements

Read analog values.
Command:

Command	Parameters
0xA0	0x71 Onboard temperature sensor

Return:

Command	Parameters
0xA0	4 Bytes floating point value casted into bytes.

4.3.2 Physicist proof commands (B)

Highly abstracted commands, recommended for operator use.

4.3.2.1 B0 – power down

This brings the unit into the following status:

- All internal tasks stopped.
- Delay lines are bypassed.

Command:

Command	Parameters
0xB0	none

Return:
ACK or NAK.

4.3.2.2 B1 – Power up

This brings the unit into the following state:

- Delay line module communication is running
- Delay lines are bypassed.

Command:

Command	Parameters
0xB1	none

Return:
ACK or NAK.

4.3.2.3 B2 – Set up

This brings the unit into the following status:

- All internal tasks are running
- Delay lines are bypassed.

Command:

Command	Parameters
0xB2	none

Return:
ACK or NAK.

4.3.2.4 B3 – Start up

This brings the unit into the following status:

- All internal tasks are running
- Delay lines are online.

Command:

Command	Parameters
0xB3	none

Return:
ACK or NAK.

4.3.2.5 BF – Interlock signals

Returns the incoming and outgoing interlock signals, combined in 1 byte.
This command is intended to be polled at high repetition rate (>10ms pause in between).

Command:

Command	Parameters	Parameters
0xBF		

Return:

Command	Parameters
0xBF	Byte 1: Bit0 : IN1 (SX) Bit1 : OUT1 (SX) Bit4 : IN2 (DX) Bit5 : OUT2 (DX)
	Byte 2 = Byte 1 (dublicated)

Example:
0x0201 BFC7 F9 -> 0x 0203 BF03 0379 AD

02	ASCII code STX
03	Packet length: 3
BF	Command
03	Bit 0 and Bit 1 are set : TBAD signal is on, delay output signal is on.
03	Duplication of the first data byte.
79AD	Check sum CRC-CCITT (0xFFFF) of the preceding bytes.

4.3.2.6 BB – Query state

Query the current state of the unit.
The states are recognized by the following characteristics: (changes marked bold)

State	State name	Characteristics
0xB0	Power down	
0xB1	Power up	
0xB2	Set up	
0xB3	Start up	

Command:

Command	Parameters
0xBB	none

Return:

Command	Parameters
0xBB	NAK command failed due to hardware issues 0xB0 power down 0xB1 Power up 0xB2 Set up 0xB3 Start up 0x00 Unknown

4.3.2.7 BC – bypass

Enable or bypass a delay module.

Command:

Command	Parameters	Parameters
0xBC	0x00 Channel 0 (SX) 0x01 Channel 1 (DX)	0x01 enable 0x00 bypass

Return:
ACK or NAK.

4.3.2.8 BD – trigger test

Trigger a test of a delay module.
If the device is in a state where changing not allowed, the return value will be 0.
Not possible in state “start up”.

Command:

Command	Parameters	Parameters
0xBD	0x00 Channel 0 (SX) 0x01 Channel 1 (DX)	

Return:

Command	Parameters
0xBD	ACK NAK

4.3.3 Delay modules (D)

Commands, related to the delay modules.

4.3.3.1 DD – query data

Get unsigned int data. Parameters, which are marked as "byte" will return a single byte cast to unsigned int. Byte order is little endian for unsigned int. E.g. 0xAE08 = 2222.
This command is intended to be polled at low repetition rate.

Command:

Command	Parameters	
0xDD	SX Module: 0x13 Setpoint 1 0x14 Setpoint 2 0x15 Setpoint 3 0x16 measured delay 1 0x17 measured delay 2 0x18 measured delay 3 0x19 measured output delay 0x1A ADC raw value 0x1B port 0, byte 0x1C port 1, byte 0x1D port 2, byte 0x1E Error code 0x1F status, byte 0x91 input/output port status byte Bit0 : bypass (0) / online (1) Bit1 : testing (1) Bit2 : delay 3 output signal Bit4 : input signal DX Module: 0x23 Setpoint 1 0x24 Setpoint 2 0x25 Setpoint 3 0x26 measured delay 1 0x27 measured delay 2 0x28 measured delay 3 0x29 measured output delay 0x2A ADC raw value 0x2B port 0, byte 0x2C port 1, byte 0x2D port 2, byte 0x2E Error code 0x2F status, byte 0x92 input/output port status byte Bit0 : bypass (0) / online (1) Bit1 : testing (1) Bit2 : delay 3 output signal Bit4 : input signal	

Return:

Command	Parameters
0xDD	NAK Value (2 bytes unsigned int, cast into bytes)

4.3.3.2 D7 – Get module's firmware version

Returns installed firmware version as a string.

Command:

Command	Parameters
0xF7	0x00 Channel 0 (SX) 0x01 Channel 1 (DX)

Return:

Command	Parameters
0xF7	Up to 16 bytes of ASCII code, representing the firmware version string.

4.3.3.3 D8 – Get module's serial number

Returns the serial number of a delay module.

Command:

Command	Parameters
0xF8	0x00 Channel 0 (SX) 0x01 Channel 1 (DX)

Return:

Command	Parameters
0xF8	Byte 1: Serial number (byte) Byte 2: Serial number (byte) (dublicated)

4.3.4 Task scheduler and control system (C)

Mostly engineering commands are low level test and debug commands.

4.3.4.1 C0 – set scheduler mode

4.3.5 Engineering commands (E)

Mostly engineering commands are low level test and debug commands.

4.3.5.1 E0

4.3.5.2 E1

4.3.5.3 E2

4.3.5.4 E3 – Load data from EEPROM

Read calibration and text data from EEPROM. Calibration data and text data is written as a block of bytes into the controller's EEPROM. Accordingly, the load data command will return the data, as it is stored in the EEPROM. See chapter calibration data for further details.

Command:

Command	Parameters
0xE3	

Return:

Command	Parameters
0xE3	NAK 38bytes of data, See chapter calibration data.

4.3.5.5 E4 – Write data to EEPROM

Write calibration and text data to EEPROM. Calibration data and text data is written as a block of bytes into the controller's EEPROM. The write data command will write the data to the EEPROM as it is. There is no further validation beside the protocol's check sum.
See chapter calibration data for further details.

Command:

Command	Parameters
0xE4	

Return:
ACK or NAK.

4.3.5.6 E5 – Get port expander port

4.3.5.7 E6 – Set port expander port

4.3.6 Test and diagnostics commands (F)

4.3.6.1 F0 – NAK

Returns NAK. Basic communication test.

Command:

Command	Parameters
0xF0	none

Return:

NAK

4.3.6.2 F1 – ACK

Returns ACK. Basic communication test.

Command:

Command	Parameters
0xF1	none

Return:
ACK

4.3.6.3 F2 – Echo data

Echo the received data. Basic communication test.

Command:

Command	Parameters
0xF2	Up to 45 bytes

Return:

Command	Parameters
0xF2	Up to 45 bytes

4.3.6.4 F3 – Convert a floating number to a string

Converts a floating point number into a string (FTOA). This is to check if the byte order of the host system matches the order used by the firmware.

Command:

Command	Parameters
0xF3	0x00, single precision floating point value casted into bytes

Return:

Command	Parameters
0xF3	0x00, Floating point number converted to a string

4.3.6.5 F4 – Return a floating number

Returns the floating number 123.456. This is to check if the byte order of the host system matches the order used by the firmware.

Command:

Command	Parameters
0xF4	none

Return:

Command	Parameters
0xF4	123.456 single precision floating point value casted into bytes.

4.3.6.6 F5 – Get error report

Read error code. An error will be cleared after reading it. If the error source is persistent, like an over temperature error, the error will occur again.

Command:

Command	Parameters
0xF5	none

Return:

Command	Parameters	
0xF5	4 Bytes representing an unsigned long, every bit represents an error.	See error code table for further details.

4.3.6.7 F6 – Diagnostics

Check a single analog for validity.

Command:

Command	Parameters
0xF6	

Return:
ACK or NAK.

4.3.6.8 F7 – get firmware version

Returns installed firmware version as a string.

Command:

Command	Parameters
0xF7	none

Return:

Command	Parameters
0xF7	Up to 9 bytes of ASCII code, representing the firmware version string.

4.3.6.9 F8 – read bad CRC counter

Every time a data package with bad CRC is received it will be counted. This command returns the counter value.

Command:

Command	Parameters
0xF8	none

Return:

Command	Parameters
0xF8	Bad CRC counter (2 bytes unsigned int, cast into bytes)

4.3.6.10 Basic configuration data

	Name	Size	Description
	parameter ID	1 Byte	Assigned id, not used
	IP address	Unsigned long integer (4 Bytes)	
	gate	Unsigned long integer (4 Bytes)	
	Mask	Unsigned long integer (4 Bytes)	
	DNS	Unsigned long integer (4 Bytes)	
	MAC	6 Bytes	
	Port	1 Byte	
	Debug	1 Byte	
	Timeout	1 Byte	
	Diag	1 Byte	
	Aux	10 Bytes	

4.4 List of acronyms

µC	Microcontroller
A	Amperé
AC	Alternating current
ADC	Analog to Digital Converter
ARGOS	Advanced Rayleigh guided Ground layer adaptive Optics System
CRC	Cyclic redundancy check
CS	Chip Select
D	Depth
DAC	Digital to Analog Converter
DC	Direct Current, =
DIN	Deutsches Institut für Normung, German Institute for Standardization
EEPROM	Electrically Erasable Programmable Read-Only Memory
HOH	Hero Of Hyrule
HP	Horizontal Pitch, with 1 HP being 5.08mm (0.20 inches)
ID	Identification
IP	Internet Protocol
EMI	Electro Magnetic Interference
MAC	Media Access Control address or Ethernet Hardware Address
MISO	Master In Slave Out
MOSI	Master Out Slave In
MPE	Max Planck Institut für extraterrestrische Physik
MPI	Max Planck Institut
P.E.	Protective Earth
PCB	Printed Circuit Board
PCD	Pockels Cell Driver
PSU	Power Supply Unit
SEP	Somebody Else's Problem
SPI	Serial Peripheral Interface
Tbd	To Be Defined
TNB	Trinitrobenzene
TP	Test Point
U	Unit or Rack Unit, with 1 U being 44.4mm (1.75 inches)
V	Volt
VDR	Voltage Dependent Resistor
0x	Indicating a hexadecimal number
EMC	ElectroMagnetic Compatibility
DC/DC	Direct Current to Direct Current converter
RTD	Resistance Temperature Detector
TC	ThermoCouple
SSDM	Stochastic Signal Density Modulation
°C	Degree Celsius, a metric a scale and unit of measurement for temperature
PWM	Pulse Width Modulation

Notes: