Communications Interface Document for the Doppler Systems Multiple Position Tracker (MPT)

Table of Contents

Introduction

The Doppler Systems Multiple Position Tracker (MPT) interfaces to a network or PC via an Ethernet connection. The MPT has a web server allowing the user to change the MPT configuration and an Ethernet binary serial interface that allows a PC to communicate with the MPT using low level commands.

Version

This document applies to DDF7000 firmware version 2.15 and later.

Physical Interface

The computer interfaces to the MPT via an Ethernet connection.

DHCP

The Ethernet software in the MPT ships with a DHCP client enabled, so if connected to a network with a DHCP server, the MPT will obtain an ip address from the server. Once it has done this it begins broadcasting its IP address on the network as a UDP broadcast message on port 9007. Two messages are transmitted. The first message consists of 27 bytes in the following format

'D','o','p','p','l','e','r',' ',
'D','D','F','6','2','8','0',*ip0,ip1,ip2,ip3,portlo,porthi,mac0,mac1,mac2,mac3,mac4,mac5*

Where the italicized values represent binary values.

The second message has the following structure

Bytes	Type	Parameter
0 – 3	UINT32	IP Address
4 – 7	Float	Latitude[1]
8 – 11	Float	Longitude[1]
12	UCHAR	Number of Connections
13	UCHAR	Major Version
14	UCHAR	Minor Version
15	UCHAR	Bits 0 – 3 receiver type Bit 4: 1 means GPS connected Bit 5: 1 means compass connected
16 – 19	UINT32	0xFFFFFFFF Delimiter

[1] If GPS is not installed latitude = 100, longitude = 190

These messages are transmitted every 2 seconds.

A Windows based program, Doppler DF Discover, is provided with the MPT unit and will display the ip address the port number and the mac address of the device.

Static IP Address

The MPT may also be connected directly to the computer using a crossover cable. The MPT is shipped with a default address of 10.0.0.100. If you are going to connect it directly to your computer you must set your computer TCP/IP properties as shown in Figure 1 below. Once it is connected you can use the web server or the binary serial interface to change the ip address and the ip port number.

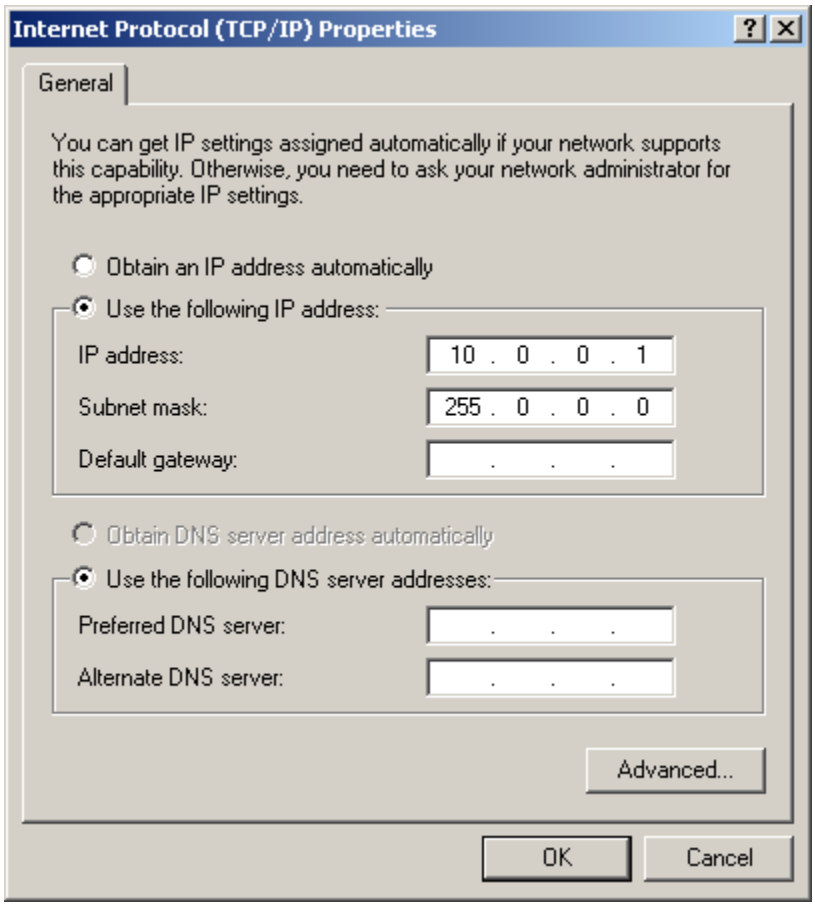

Figure 1: TCP/IP Properties for Direct Connection to the MPT

Web Server

The MPT web server allows the user to change the settings of the direction finder. Any browser can be used to display the web server by typing in the ip address of the MPT.

Binary Serial Interface

The default port of the binary serial interface is 2101. This can be changed using the web server or the binary serial interface.

Command Structure

Table 1: Structure of message

Byte #	Definition	Comments
0	0x02	STX indicates the start of the message
1	Length (LSB)	LSB of the total length of the message (from byte 3 through byte n)
2	Length (MSB)	MSB of the total length of the message
3	Message ID (LSB)	LSB of the message identifier
4	Message ID (MSB)	MSB of the message identifier
5 – n	Message data	Any data sent with the message
n + 1	CRC (LSB)	LSB of CRC check sum
n + 2	CRC (MSB)	MSB of CRC check sum
n + 3	0x03	ETX indicates the end of the message

* CRC is defined as CRC16 standard using 8005 for the generating polynomial see "A Painless Guide to CRC Error Detection Algorithms" for implementation details. CRC calculated on bytes 1 through n

Commands

The commands accepted by the MPT are listed in Table 2.

Table 2: Commands

Command Name	Command Number	Data Length	Range	Comments
Poll for Bearing	0x0000	0	0	Applies to DF modes 8, 9, 10 ignored otherwise
Sweep Rate	0x0001	byte	0-4	0=250,1=500.**2=1000**,3=2000
Averages	0x0002	byte	1-20	Default is **2**
Attenuator	0x0003	byte	0 – 1	**0 = off**, 1 = on
Audio Volume	0x0004	byte	0 – 96	0 = max volume, 96 = muted, **default 20**
Tone Volume	0x0005	byte	0 – 96	0 = max volume, 96 = muted, **default 20**
Receiver Type	0x0006	byte	0 – 5	0 = ICOM R8500 1 = ICOM PCR1500 2 = ICOM PCR2500 3 = AOR AR8600 4 = AOR AR5000 **5 = AOR SR2200** 6 = Motorola MCS2000 7 = ICOM R9500 If the receiver requires a serial port and no serial port has been assigned for the receiver then serial port 0 will be assigned to the receiver
Sample Time	0x0007	short	10 – 10000	Sample time in ms or pulse width for gated mode. Maximum value in modes 1, 8, 9, 10 is 1500 ms **Default 500 ms**
DF Mode	0x0008	3 bytes	Byte 1 0 = manual **1 = continuous** 2 = gated on rising edge	Manual allows sweep to be turned on and off and the directions switched by manual command. Currently only modes 0 through 3 are

			3 = gated on falling edge 4 = gated on positive pulse 5 = gated on negative pulse 6 = continuous CW sweep only 7 = continuous CCW sweep only 8 = send last bearing on poll 9 = send next bearing on poll 10 = start bearing calculation on poll Byte 2-3 XX = Sample time in ms 10 – 10000 applies to modes 1, 2, 3, 6, 7, and 9	implemented. Other modes are for future use. **Default 500 ms**
Threshold	0x0009	short	10 – 10000	Sets the maximum separation of consecutive bearing measurements that will be used in the bearing calculation. Setting this number low will result in some missed bearings. Setting it high will result in more false positives. **Default 2000**
Antenna	0x000A	byte	**0 = VHF** 1= UHF 2 = THF 3= Auto	In Auto mode the antenna switches will be controlled by the receiver frequency as shown below (receiver frequency must be controlled by the direction finder) 100 < f < 250 MHz - VHF Antenna 250 ≤ f < 500 MHz – UHF Antenna 500 ≤ f < 1000 MHz – THF antenna
Echo Type	0x000B	byte	0 = no echo **1 = echo with data** 2 = echo with OK	Sets the way the DF will respond when it receives a command

Direction Cosines	0x000C	4 INT16	-10000 - 10000	Range is -1 to 1 (divide by 10000) Each receiver has calibrated direction cosines.
Calibrate	0x000D	INT16	0 – 3599	Range is 0-359.9 (divide by 10) Calibrate the bearing to a given angle
Indentify Hardware	0x000E	0		Causes hardware version response
Identify Software	0x000F	0		Causes software version response
List Serial Ports	0x0010	0		Lists each serial port connected to the system and its parameters
Configure Serial Port	0x0011	5 bytes	Byte 0 : port number Byte 1 : Baud Rate 0 = 1200 1 = 2400 **2 = 4800** 3 = 9600 4 = 19200 5 = 38400 Byte 2 : Stop Bits **0 = 1** 1 = 1.5 2 = 2 Byte 3 : Data Bits 0 = 7 **1 = 8** Byte 4 : Parity **0 = none** 1 = even 2 = odd	
Assign Serial Port	0x0012	2 bytes	Byte 0 : port number Byte 1 : device **0 = NMEA**	Port number 0 – 3

			1 = Receiver 2 = pass through 3 = radio modem 4 = radio modem pass through	Pass through means all data flows from the serial port to the Ethernet port and vice versa. Data sent from computer to radio modem and vice versa
Send DF Settings	0x0013	0		Causes DF settings response
Set Frequency	0x0014	UINT32	$0 – 2x10^9$	Hz
Set Squelch	0x0015	byte	0 – 255	**Default 0**
Set Rx Volume	0x0016	byte	0 – 255	**Default 128**
Receiver Pass Through	0x0017	Various		Sends command directly to receiver
Sweep	0x0018	byte	Nibble 0 : Direction 0 = CW 1 = CCW Nibble 1: Sweep State 0 = Sweep Off 1 = Sweep On	Manual command to start the sweep
Calibrate S Meter	0x0019	byte	0 – 9	
Filter	0x001A	byte	0 = Off, **1 = On**	
Auto Output	0x001B	byte	0 = Off, **1 = On**	
Hold Time	0x001C	byte	Hold time in seconds	Time DF holds the bearing calculation DF Sends 360 when hold time expires **Default 5 seconds**
DHCP	0x001D	byte	0 = do not use DHCP **1 = use DHCP**	
IP Address	0x001E	UINT32	Sets the IP Address	If using DHCP this sets the default IP address
IP Gateway	0x001F	UINT32	Sets the IP Gateway	
IP Subnet Mask	0x0020	UINT32	Sets the IP Subnet Mask	
IP DF Port	0x0021	UINT16	Sets the port for the DF	**Default 2101**

			communications	
Reserved	0x0022	0	Reserved	
Set to Default	0x0023	0	Sets all parameters to their default values (IP parameters excluded)	
Reset Processor	0x0024	0	Restarts the processor (All ip connections will be terminated)	
Reserved	0x0025			
Serial Pass Through	0x0026	Various	byte 0: port Number remaining bytes serial port data	
Send Serial Number	0x0027	0	Sends the serial number of the unit	
Stream Audio	0x0028	UINT16 UINT32 UINT8	Port number to stream audio to **0 = disable streaming** IP Address to stream audio to On LAN will stream to IP connection. 1 = send time stamp	
Streaming Audio Squelch	0x0029	UINT16	Audio must exceed squelch level in order for audio to be streamed	**Default 50**
Compress Audio	0x002A	UINT8	**0 = no compression** 1 = A law compression	
Send Radio Modem	0x002B	Various	Byte 0: port number Remaining bytes serial port data	
Set Standard Deviation	0x002C	UINT8	Standard Deviation x 10	**Default = 20 (standard devation = 2)**
Set Unit ID	0x002D	UINT8	ID of the unit on a radio modem network	**Default = 0.** Must be set to 1 or higher when used with radio modem
Set latitude	0x002E	Float	Latitude of a fixed site if no GPS is connected to unit	**Default = 91.0.** 4 bytes.
Set longitude	0x002F	Float	Longitude of a fixed site if no	**Default = 181.0.** 4 bytes

			GPS is connected to the unit	
Self-Test	0x0030	UINT8	Controls the self-test	0 – Cancel self-test 1 – Start self-test
Reserved	0x0031			
Save Measured Antenna Readings as Baseline	0x0032	UINT8	Antenna Number	0 – VHF 1 – UHF 2 – THF
Reserved	0x0033	UINT8		
Reserved	0x0034	9 bytes		
Set Receiver Mode	0x0035	1 byte	Mode of receiver	0 = FM, 1 = AM

Responses

With the exception of commands that specifically ask for data to be sent the response to the commands depends on the echo type setting (0x0B). If it is set to no echo, no response will be sent. If it is set to echo with data it will echo the data back in the exact format it was sent with the actual settings of the device. If it is set to echo with ok a message with an ACK (0x06) in the data field will be sent back if the data was accepted and a NAK (0x15) if the data was rejected.

Table 3: Responses to Commands Requesting Data

Response Name	Response Number	Format	Comments
Bearing Message	0x0000	ASCII	Comma delimited string Bearing (0-359.9), S meter (0-255), Number of Averages (0-20), Audio Level (0-2047) Hour[1], Minute[1], Second[1], Direction of Rotation[2]
Identify Hardware	0x000E	ASCII	String will be of the form *major version .minor version*
Identify Software	0x000F	ASCII	String will be of the form *major version .minor version*
Direction Cosines	0x000C	16 Bytes	Bytes 0-3 10000 * cos theta CW Bytes 4-7 10000 * sin theta CW Bytes 8-11 10000 * cos theta CCW Bytes 12-15 10000 * sin theta CCW All in Little-Endian format
List Serial Ports	0x0010	ASCII	Comma separated list of the connected port data Port Number (0-3), Status, Baud Rate (1200 – 38400), Data bits (7 or 8) Stop bits (1, 1.5, or 2 Parity (N = None, O=Odd, E=Even, M=Mark, S=Space, Function (NMEA,RECEIVER,PASS_THROUGH,RADIO_MODEM, RADIO_MODEM_PASS_THROUGH) Each set of port parameters is separated by a carriage return character
Send DF Settings	0x0013	ASCII	Settings will be sent as a block of ASCII data with each setting having

			the following format *Cmd number,setting<CR>*
S Meter Cal Constants	0x0019	ASCII	Two comma separated strings representing the slope and intercept constants in exponent format (e.g. 1.23e-1,1.045e3)
Receiver Data	0x0017	Bytes	Data received from the receiver that is passed through.
Send RX Settings	0x0022	ASCII	Settings will be sent as a block of ASCII data with each setting having the following format *Cmd number,setting<CR>*
Serial Port Data	0x0026	Bytes	Data from the serial ports Byte 0 = port number Remaining bytes is the message data
Serial Number	0x0027	ASCII	Serial number
Radio Modem Data	0x002B	Bytes	Data from radio modem passing through MPT Byte 0 = port number Remaining bytes is the message
Self-Test Complete	0x0030	ASCII	Comma separated strings with gains from each of the eight antennas *XHF,Frequency<CR>* (X = V, U, T) *Ant#,Baseline Measurement,Measurment* (x8)

[1] Time is sent if GPS is connected to the processor
[2] CW or CCW is sent if the averages are set to 1.

Audio Streaming Format

The audio stream is sent via UDP. If the send time stamp option is disabled (see command 0x0028) then the audio is sent as raw PCM data sampled at 7816 Hz. If the compressed option is turned on (command 0x002A) then each byte is a PCM value and can be decompressed using the algorithm presented in Appendix B. If the compressed option is turned off then the PCM values are two bytes (16 bit) values. Compression off gives the best fidelity and is normally preferred unless there is an extreme limitation on bandwidth.

If the time stamp option is turned on then each audio sample is followed by a 9 byte trailer with the following format

Byte	Type	Parameter
0	UCHAR	Index
1	UCHAR	Hour
2	UCHAR	Min
2-6	Float	Second
7 – 8	UINT16	0xFFFF

The index is an unsigned byte and is sent to help the receiving software align the audio packets in the event a packet is delayed or dropped by the network. This is usually not an issue on local area networks (LAN) but can be problematic when the MPT is used on over the Internet or other wide area network. The time tag is used to synchronize the audio with the bearing data when a GPS is connected to the MPT.

Appendix A: C function for implementing the CRC calculation

```
u16    CRCTable[] =
   {
   0x0000, 0xC0C1, 0xC181, 0x0140, 0xC301, 0x03C0, 0x0280, 0xC241, 0xC601, 0x06C0, 0x0780,
   0xC741, 0x0500, 0xC5C1, 0xC481, 0x0440, 0xCC01, 0x0CC0, 0x0D80, 0xCD41, 0x0F00, 0xCFC1,
   0xCE81, 0x0E40, 0x0A00, 0xCAC1, 0xCB81, 0x0B40, 0xC901, 0x09C0, 0x0880, 0xC841, 0xD801,
   0x18C0, 0x1980, 0xD941, 0x1B00, 0xDBC1, 0xDA81, 0x1A40, 0x1E00, 0xDEC1, 0xDF81, 0x1F40,
   0xDD01, 0x1DC0, 0x1C80, 0xDC41, 0x1400, 0xD4C1, 0xD581, 0x1540, 0xD701, 0x17C0, 0x1680,
   0xD641, 0xD201, 0x12C0, 0x1380, 0xD341, 0x1100, 0xD1C1, 0xD081, 0x1040, 0xF001, 0x30C0,
   0x3180, 0xF141, 0x3300, 0xF3C1, 0xF281, 0x3240, 0x3600, 0xF6C1, 0xF781, 0x3740, 0xF501,
   0x35C0, 0x3480, 0xF441, 0x3C00, 0xFCC1, 0xFD81, 0x3D40, 0xFF01, 0x3FC0, 0x3E80, 0xFE41,
   0xFA01, 0x3AC0, 0x3B80, 0xFB41, 0x3900, 0xF9C1, 0xF881, 0x3840, 0x2800, 0xE8C1, 0xE981,
   0x2940, 0xEB01, 0x2BC0, 0x2A80, 0xEA41, 0xEE01, 0x2EC0, 0x2F80, 0xEF41, 0x2D00, 0xEDC1,
   0xEC81, 0x2C40, 0xE401, 0x24C0, 0x2580, 0xE541, 0x2700, 0xE7C1, 0xE681, 0x2640, 0x2200,
   0xE2C1, 0xE381, 0x2340, 0xE101, 0x21C0, 0x2080, 0xE041, 0xA001, 0x60C0, 0x6180, 0xA141,
   0x6300, 0xA3C1, 0xA281, 0x6240, 0x6600, 0xA6C1, 0xA781, 0x6740, 0xA501, 0x65C0, 0x6480,
   0xA441, 0x6C00, 0xACC1, 0xAD81, 0x6D40, 0xAF01, 0x6FC0, 0x6E80, 0xAE41, 0xAA01, 0x6AC0,
   0x6B80, 0xAB41, 0x6900, 0xA9C1, 0xA881, 0x6840, 0x7800, 0xB8C1, 0xB981, 0x7940, 0xBB01,
   0x7BC0, 0x7A80, 0xBA41, 0xBE01, 0x7EC0, 0x7F80, 0xBF41, 0x7D00, 0xBDC1, 0xBC81, 0x7C40,
   0xB401, 0x74C0, 0x7580, 0xB541, 0x7700, 0xB7C1, 0xB681, 0x7640, 0x7200, 0xB2C1, 0xB381,
   0x7340, 0xB101, 0x71C0, 0x7080, 0xB041, 0x5000, 0x90C1, 0x9181, 0x5140, 0x9301, 0x53C0,
   0x5280, 0x9241, 0x9601, 0x56C0, 0x5780, 0x9741, 0x5500, 0x95C1, 0x9481, 0x5440, 0x9C01,
   0x5CC0, 0x5D80, 0x9D41, 0x5F00, 0x9FC1, 0x9E81, 0x5E40, 0x5A00, 0x9AC1, 0x9B81, 0x5B40,
   0x9901, 0x59C0, 0x5880, 0x9841, 0x8801, 0x48C0, 0x4980, 0x8941, 0x4B00, 0x8BC1, 0x8A81,
   0x4A40, 0x4E00, 0x8EC1, 0x8F81, 0x4F40, 0x8D01, 0x4DC0, 0x4C80, 0x8C41, 0x4400, 0x84C1,
   0x8581, 0x4540, 0x8701, 0x47C0, 0x4680, 0x8641, 0x8201, 0x42C0, 0x4380, 0x8341, 0x4100,
   0x81C1, 0x8081, 0x4040
   };
```

```
// DESCRIPTION:   Calculates the CRC-16 value of a buffer of data
//
// INPUT:         *data       Pointer to the data buffer
// INPUT:         NumBytes    Number of bytes to read for CRC

u16 CalcCRC16 (u8 *Data, u16 NumBytes)
{
   unsigned int   i;
   u8  CRCLoByte;
   u16 CRCResult;


   CRCResult = 0x0000;
   for (i = 0; i < NumBytes; i++)
   {
      CRCLoByte = (u8)(CRCResult & 0x00FF);
      CRCResult = ((CRCResult & 0xFF00) >> 8) ^ CRCTable[*Data++ ^ CRCLoByte];
   }

   return CRCResult;
}
```

Appendix B: A law decompression code

```
short decode(byte alaw)
{
    //Invert every other bit, and the sign bit (0xD5 = 1101 0101)
    alaw ^= 0xD5;
    //Pull out the value of the sign bit
    int sign = alaw & 0x80;
    //Pull out and shift over the value of the exponent
    int exponent = (alaw & 0x70) >> 4;
    //Pull out the four bits of data
    int data = alaw & 0x0f;

     //Shift the data four bits to the left
     data <<= 4;
     //Add 8 to put the result in the middle of the range (like adding a half)
     data += 8;

     //If the exponent is not 0, then we know the four bits followed a 1,
     //and can thus add this implicit 1 with 0x100.
     if (exponent != 0)
         data += 0x100;
     /* Shift the bits to where they need to be: left (exponent - 1) places
     * Why (exponent - 1) ?
     * 1 2 3 4 5 6 7 8 9 A B C D E F G
     * . 7 6 5 4 3 2 1 . . . . . . . . <-- starting bit (based on exponent)
     * . . . . . . . Z x x x x 1 0 0 0 <-- our data (Z is 0 only when exponent is 0)
     * We need to move the one under the value of the exponent,
     * which means it must move (exponent - 1) times
     * It also means shifting is unnecessary if exponent is 0 or 1.
     */
     if (exponent > 1)
          data <<= (exponent - 1);

     return (short)(sign == 0 ? data : -data);
 }
```