PASSMARK® SOFTWARE

USB Power Delivery Tester

Communication Protocol Guide

Model PM110 Rev 1

Document Edition: 1.0
Date: 14 May 2018
Web site: www.passmark.com

PASSMARK® SOFTWARE

Table of Contents

PASSMARK® SOFTWARE

Introduction

This document is targeted at experienced software developers wishing to develop their own software to interact with the USB PD tester.

Further details about the USB PD hardware can be found here,
https://www.passmark.com/products/usb-power-delivery-tester.htm

The USB PD Tester communicates with the host using a serial over USB interface. This interface is made using FTDI's FT230x chip. For more details and documentation on the interface chip see the FTDI web site.
http://www.ftdichip.com/

This chip has two interface options. A classic serial port (which gives limited control) and the more advanced D2XX interface. High level languages can access to the Passmark USB Power Delivery tester using the FTDI's proprietary "direct" driver interface. This interface, D2XX, is provided via a proprietary DLL (FTD2XX.DLL). Please refer to the "D2XX Programmer's Guide" for detailed explanation of the functions available in the DLL.
http://www.ftdichip.com/Support/Documents/ProgramGuides.htm

Search in D2XX devices

D2XX driver allows you to get a list of all FTDI devices connected. Below is an example code for searching in D2XX devices to find the Passmark USB Power Delivery testers:

```
DWORD       devcount = 0;
UCHAR       serial[50];
FT_STATUS   ftStatus;

ftStatus = FT_ListDevices(&devcount, NULL, FT_LIST_NUMBER_ONLY);
for (DWORD curDevice = 0; curDevice < devcount; curDevice++)
{
        ftStatus = FT_ListDevices((PVOID)curDevice, serial, FT_LIST_BY_INDEX
        | FT_OPEN_BY_SERIAL_NUMBER);
        if(ftStatus == FT_OK)
        {
                if(!strncmp((char*)serial, "PMPD" , 4))
                {
                        // Add this serial number to the list of testers.
                }
        }
}
```

Open a device

The below code opens a device by its serial number and returns a handle that will be used for subsequent accesses.

```
FT_HANDLE devHandle;
FT_STATUS  ftStatus;

ftStatus = FT_OpenEx(serial, FT_OPEN_BY_SERIAL_NUMBER, &devHandle);
if(ftStatus == FT_OK)
{
        FT_SetBaudRate(devHandle, 115200);
        FT_SetDataCharacteristics(devHandle, FT_BITS_8, FT_STOP_BITS_1,
        FT_PARITY_NONE);
        FT_SetTimeouts(devHandle, 1, 100);
}
```

Send and Receive data

```
FT_STATUS  ftStatus;
ftStatus = FT_Write(devHandle, buf, len, &NumBytesWritten);
ftStatus = FT_Read(devHandle, buf, 1, &NumRecvBytes);
```

Example Command - Get Connection Status

```
FT_HANDLE devHandle;
FT_STATUS  ftStatus;
DWORD      NumBytesWritten;
DWORD      NumRecvBytes;
UCHAR      cmd_buf[8];
UCHAR      recv_buf[64];

cmd_buf[0] = 0x02;   // Header
cmd_buf[1] = 0x01;   // Length
cmd_buf[2] = 0x0A;   // Command
cmd_buf[3] = 0x0A;   // Checksum
cmd_buf[4] = 0x03;   // Flag

ftStatus = FT_Write(devHandle, cmd_buf, 5, &NumBytesWritten);
if (ftStatus == FT_OK)
{
        ftStatus = FT_Read(devHandle, &recv_buf, 15, &NumRecvBytes);
        if (ftStatus == FT_OK)
        {
                PortType = recv_buf[4];
                SelectedVoltage = ((WORD)recv_buf[6] << 8) | recv_buf[5];
                DetectedMaxCurrent = ((WORD)recv_buf[8] << 8) | recv_buf[7];
        }
}
```

PASSMARK® SOFTWARE

Communication Protocol

Communication Format

Following is the frame structure for the commands sent by the host and response received from the tester.

Header	Length	Command	Data	Checksum	Closing Flag

Header: This is a single byte that indicates the beginning of the frame.
Length: This byte indicates the total number of bytes between Length and Checksum.
Command: This byte is used to instruct the tester which operation to perform. For the response packets, this indicates which command's response.
Data [Payload]: Data bytes are the parameters of a command or response. The least significant byte is always sent and received first (LSB First).
Checksum: This byte is used on the host as well as the tester to check the validity of the packet and to trap any data corruption. This is calculated by XORing all the bytes between Length and Checksum.
Closing Flag: This is a single byte that indicates the end of the frame.

Commands

Command Overview

Table below summarizes the list of USB PD Tester commands.

Command	Description
0x0A	Get Connection Status
0x0B	Get Capabilities
0x0C	Get Statistics
0x0D	Change Power Profile
0x10	Set Load
0xE0	Change Config
0xE1	Get Config

Get Connection Status

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	0x01	0x0A	-	0x0A	0x03

Response:

Header	Length (bytes)	0x02
Length	1	0x0B
Command	1	0x0A
Connection Status	1	0x00: Not Connected 0x01: Connected
Port Type	1	0x01: SDP (Standard Downstream Port) 0x02: CDP (Charging Downstream Port) 0x03: DCP (Dedicated Charging Port) 0x04: Type-C (with PD support)

		0x05: Proprietary Charger 0x06: Type-C (without PD support)
Selected Voltage	2	Selected Voltage in millivolts
Max Current	2	Maximum current for the selected voltage
Max Power	4	Maximum power for the selected voltage
Checksum	1	To be calculated
Flag	1	0x03

Get Capabilities

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	0x01	0x0B	-	0x0B	0x03

Response:

Header	Length (bytes)	0x02
Length	1	0x2d
Command	1	0x0B
Num Power Profiles	2	Number of the power profiles
Power Data Objects	35	Profile Index (1 byte) Voltage in millivolts (2 bytes) Max Current in milliamperes (2 bytes) . Next Profile Data Object
Checksum	1	To be calculated
Flag	1	0x03

Get Statistics

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	0x01	0x0C	-	0x0C	0x03

Response:

Header	Length (bytes)	0x02
Length	1	0x09
Command	1	0x0C
Data	8	Internal Temperature in Celsius (1 byte) Heatsink Temperature in Celsius (1 byte) Voltage in millivolts (2bytes) Set Current in milliamperes (2bytes) Current in milliamperes (2bytes)
Checksum	1	To be calculated
Flag	1	0x03

Change Power Profile

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	0x02	0x0D	Profile Index	To be calculated	0x03

Profile Indexes can be optioned by sending “Get Capabilities” command.

Response:

Header	Length (bytes)	0x02
Length	1	0x02
Command	1	0x0D
Data	1	Status Byte 0x00: Successful 0x01: Failed
Checksum	1	To be calculated
Flag	1	0x03

Set Load

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	0x03	0x10	Load in milliamperes	To be calculated	0x03

Response:

Header	Length (bytes)	0x02
Length	1	0x02
Command	1	0x10
Data	1	Status 0x00: Successful 0x01: Failed
Checksum	1	To be calculated
Flag	1	0x03

Change Config

Command:

Header	Length	Command	Data	Checksum	Flag
0x02	Depends on the parameter	0xE0	See Table below	0x10	0x03

Parameter	Data Bytes
Loopback Port	Disable: 0x00 0x00 Enable: 0x00 0x01
Current Limit	Enforce Limits: 0x01 0x00

	Allow 20% over current: 0x01 0x01 Force Limit: 0x01 0x02 MaxCurrent in milliamperes (2 Bytes) Example: Set maximum current to 3000mA 0x01 0x02 0xB8 0x0B
Max SDP Current	0x02 MaxCurrent in milliamperes (2 Bytes) Example: Set maximum current to 900mA 0x02 0x84 0x03

Response:

Header	Length (bytes)	0x02
Length	1	0x02
Command	1	0xE0
Data	1	Status 0x00: Successful 0x01: Failed
Checksum	1	To be calculated
Flag	1	0x03

Get Config

Command

Header	Length	Command	Data	Checksum	Flag
0x02	0x02	0xE1	See Table below	0x10	0x03

Parameter	Data Bytes
Loopback Port	0x00
Max SDP Current	0x02

Response:

Header	Length (bytes)	0x02
Length	1	Depends on the parameter
Command	1	0xE0
Data	Depends on the parameter	Loopback Port: 0x00: Disabled 0x01: Enables Max SDP Current: 2 bytes LSB First
Checksum	1	To be calculated
Flag	1	0x03