Command Reference

ddm hopt+schuler

582 Display - Encoder

Purpose oft the Document

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing.

This document serves as a summary of our Device Portal documentation for the 881, containing relevant commands. It does not claim to be exhaustive. The complete document is available in our Test Software “Device Portal”.

You can download the “Device Portal” with the following Link: https://ddmhoptschulergmbh.sharepoint.com/:f:/g/EiZmm_gzJgNOqQZUqAOqK5YBUOMNVJCsvGpycwlZ661pkQ

Note: To use the Device Portal on Windows, the necessary drivers must also be installed.

History

Date	Rev	Note
11.09.2025	1.00	Initial release

Content

1. Deviceportal User Guide

1.1 Introduction

The purpose of this manual is to explain the usage of the ddm testing and demo tool DevicePortal. DevicePortal is an application which can connect to several different ddm devices for demo, testing and maintenance purpose. This manual also contains details about the different devices, their configuration and the interface definition for developers to use it as reference for custom software development.

1.2 Installation

DevicePortal is running on Windows and Linux. If you are running Windows, just execute the provided executable to run setup. Follow the instructions of the setup to install DevicePortal. If you are running Linux, the installation depends on the Linux distribution. Install the provided package with commands depending on the distribution you are running.

1.3 Connecting to a Device

If you are running DevicePortal you will see an application window like this:

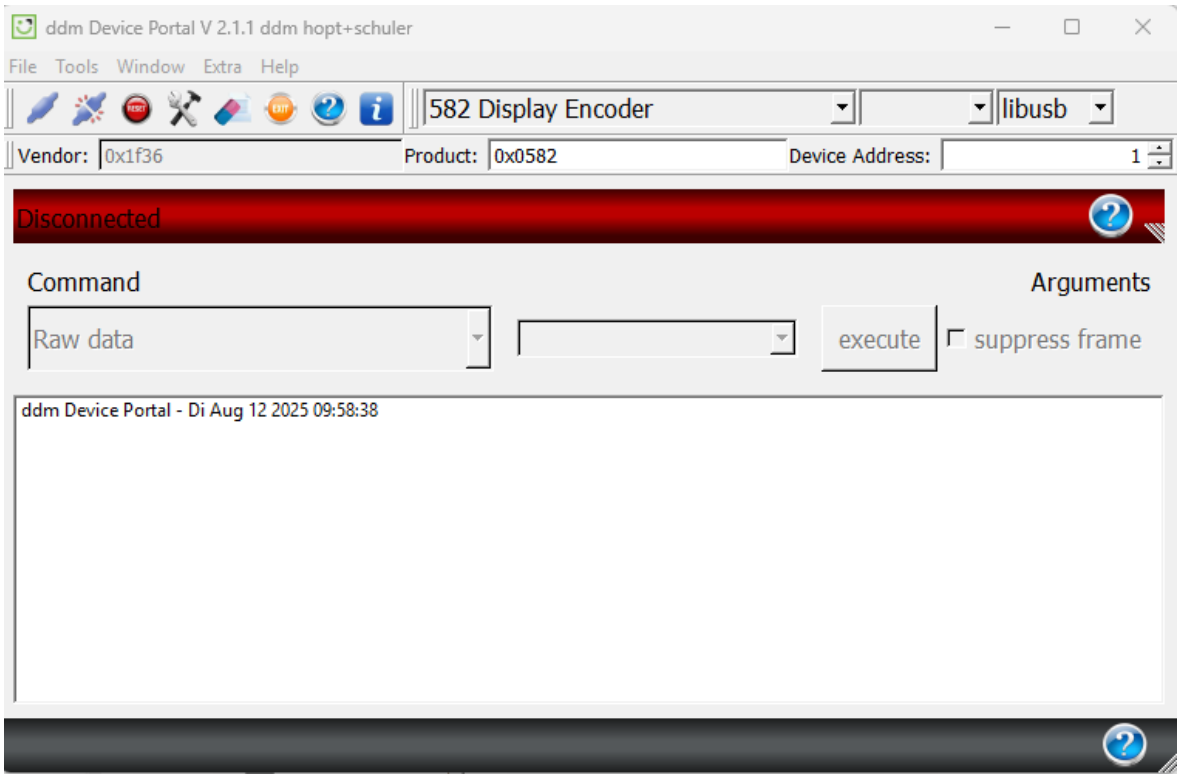

Figure 1: DevicePortal main window

On the top right of the window you can select the device to connect and the interface for the connection. If you select USB or libusb there is nothing more to do (for libusb you need to install drivers when using Windows). After device and interface is configured, click connect on the right side of the toolbar. If the device is connected and the parameters are correct, the application connects to the device and two new windows open. One window shows the status of the device, the other one shows controls. The status line of the main window shows 'Connected to the device ¡device¿'. The upper pane shows basic device information while the lower pane shows the communication between the device and the host.

1.4 Device Interface Specification

All ddm devices are using the same structure for commands and responses sent to and received from the device. These structures are defined as follows.

1.4.1 Frame Format

A frame surrounds the communication data between the host and the device. Data from host to device consists of one command byte followed by command parameters if necessary. Data from device to host consists of one status byte followed by data, depending on the preceeding command. The structure of all frames is defined as shown here:

SOH	ADDR	LEN	DATA	BCC

Figure 2: ddm interface frame format

While transmitting a frame, a maximum of 500ms between two consecutive characters is allowed.

SOH Fixed start byte. Must be set to 0x01.

ADDR One byte, defining the device address. For devices connected via RS232 or USB, this value is always 1. For TCP/IP or UDP connections the address allows to communicate with several different devices in the network. The address has to be cponfigured in the network configuration of the device.

LEN Two bytes (MSB first), defines the length of the following data in bytes.

DATA The command or response data.

BCC A block check character calculated as exclusive OR over all preceding bytes (including SOH).

1.4.2 Checksum calculation

The checksum used by the command/response frame and other things like the device configuration calculates an exclusive or (XOR) over all bytes. The checksum can be calculated with the following code example:

```
uint8_t CalculateBcc(const uint8_t *ptrSource, uint16_t DataLength)
{
        uint8_t BccResult = 0x00U;
        while( DataLength-- )
                BccResult ^= *ptrSource++;
        return BccResult;
}
```

1.4.3 Encoder events

event_Encoder	Send	Len
Sent by the device	0xD0: event ENCODER 1 Type: 1 MENU KEY LEFT 0x00 MENU KEY RIGHT 0x01 MENU KEY PRESS 0x02 MENU KEY LONG PRESS 3S 0x03 MENU KEY LONG PRESS 10S 0x04 Position	1 1 4
Description	Indicates, that an encoder was turned or pushed	

1.5 Lcd control

For controlling the Lcd display, the command 0x44 is used. Currently for Lcd there are the subcommands fill, select, show and upload.

1.5.1 Lcd fill

LCD_CMD	Send	Len
Transmit	0x44	1
	0x01 LCD_FILL	1
	Color:	1

LCD_CMD	Send	Len
	black 0x01	
	red 0x02	
	green 0x03	
	blue 0x04	
	cyan 0x05	
	magenta 0x06	
	yellow 0x07	
	white 0x08	
Receive	0x00 (stat OK) or error code	1
Description	Fills the specified color into the selected area	

1.5.2 **Lcd select**

LCD_CMD	Send	Len
Transmit	0x44	1
	0x02 LCD_SELECT	1
	X	2
	Y	2
	width	2
	height	2
Receive	0x00 (stat OK) or error code	1
Description	Selects a region of the display region	

1.5.3 **Lcd show**

LCD_CMD	Send	Len
Transmit	0x44	1
	0x03 LCD_SHOW	1
	id (MSB)	4
Receive	0x00 (stat OK) or error code	1
Description	Displays image with id on display. If image with id does not exist, the display remains in current state.	

1.5.4 **Lcd upload**

To upload images, the image must be in gif format. The data is sent using subsequent commands, because the image is too large to send to the device at once. Instead the image is sent in chunks of 1024 bytes. The command sequence contains a serial byte, that have to be incremented for every command sent until the complete image data is uploaded.

The first command to send looks like this:

LCD_CMD	Send	Len
Transmit	0x44	1
	0x04 LCD_UPLOAD	1
	0x00 (first serial)	1
	image id (MSB first)	4
	metadata	4
Receive	0x00 (stat OK) or error code	1
Description	Start upload of gif file	

After sending this first command, the data chunks are sent subsequently while incrementing the serial byte. The last chunk contains just the remaining bytes and don't have to be padded to 1024.

LCD_CMD	Send	Len
Transmit	0x44	1
	0x04 LCD_UPLOAD	1
	serial	1
	data chunk	1-1024
Receive	0x00 (stat OK) or error code	1
Description	upload chunks of gif file	

1.5.5 **Lcd brightness**

LCD_CMD	Send	Len
Transmit	0x44	1
	0x05 LCD_BRIGHTNESS	1
	Brightness (1-100%)	1
Receive	0x00 (stat OK) or error code	1
Description	Adjusts the brightness	

1.5.6 **LCD get information**

This command returns the id of the currently displayed image

LCD_CMD	Send	Len
Transmit	0x44 0x06 LCD GET INFO	1 1
Receive	0x00 (stat OK) or error code id the id of current image	1
Description	returns the id of the currently displayed image.	

1.5.7 **Upload Navigation**

To upload navigation information, for every image id the ids for images to display if button is turned or pressed can be defined.

Navigation_CMD	Send	Len
Transmit	0x43 n times (picture id from picture id to show on left picture id to show on right picture id to show on push picture id to show on long push 3s picture id to show on long push 10s)	1 1 1 1 1 1 1
Receive	0x00 (stat OK) or error code	1

Navigation_CMD	Send	Len
Description	upload navigation information	