Programmer's Guide

DLPC8445 and DLPC8445V

TEXAS INSTRUMENTS

ABSTRACT

This guide details the software interface requirements for DLPC8445 and DLPC8445V controller-based systems. This description includes the communication protocol, initialization, common use cases, and command descriptions.

Table of Contents

Trademarks

All trademarks are the property of their respective owners.

1 Scope

This Programmer's Guide (PG) defines all the supported commands through various interfaces that are available to use with DLPC8445 and DLPC8445V controllers (controller).

- I^{2}C and USB are the available interfaces for sending commands to the controller. All commands are accepted across these interfaces.
- I^{2}C is always available, USB availability depends on system configuration.
- USB interface can be made available via a TSTPT configuration.

This PG pertains specifically to using a single controller with the part numbers shown in the table below. This PG does not support dual DMD, dual DLPA PMICs, laser-phosphor color wheel (LPCW), or color overlap.

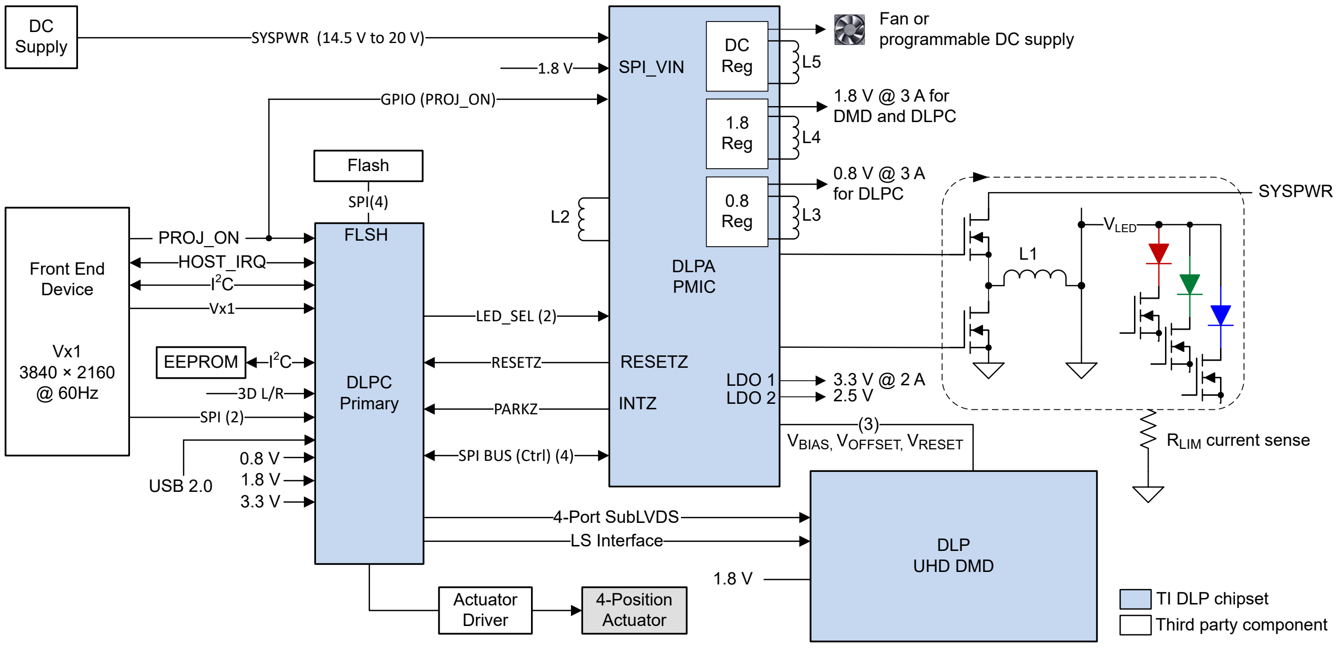
For additional information on chips in the chipset, visit ti.com and look at the device product folder of interest.

2 Introduction

2.1 System Overview

A typical TI DLP® Pico™ chipset consists of a controller, a PMIC, and a DMD. The DMD and PMIC are configured by the controller. Figure 2-1 shows a high level system structure. A DLP controller communicates with an Application Processor through commands supported through various interfaces.

Figure 2-1. Example Application Block Diagram



2.2 System Initialization

2.2.1 Boot ROM

The controller employs a Boot ROM and associated boot software. This resident boot code consists of the minimum code necessary to complete the program loading from Flash to internal RAM at power-up. Therefore, an external flash device is required to store the device firmware.

The Boot ROM also contains software that will receive I^{2}C/USB interface commands at power-up where the commands are specifically for programming or re-programming the controller’s flash device. Before programming or reprogramming, the flash device can have valid firmware, or corrupted firmware, or the flash device can be blank.

Boot ROM commands will be applicable only when the system is in Boot hold mode. The system can get in Boot hold mode when either the Boot hold pin (STAY_IN_BOOT) is set, or flash is empty or has corrupted Secondary Boot firmware, or when main application (MainApp) explicitly goes to Boot ROM for firmware upgrade. In other scenarios boot software will load the valid MainApp and will transfer control to MainApp or into Secondary Boot. Command support will be limited in Boot ROM.

Note

As noted in Section 10.1, the flash program must be reprogrammed whenever the system transitions from MainApp.

2.2.2 Secondary Boot

The Secondary Bootloader is a backup feature added to the DLPC8445 and DLPC8445V embedded software to support modifications to the Boot process. In Boot ROM, it is recommended to use a Secondary Boot for the flash upgrade process.

Secondary Boot can be accessed from Boot ROM by:

1. Initializing into Boot ROM with STAY_IN_BOOT set to 1
   a. For DLPC8445 and DLPC8445V controllers this is TSTPT0
2. Setting the System Type to Flash System through the Write System Type command (see Section 7.1.12)
3. Setting the application to Secondary boot (from Boot ROM) (see Section 8.1.3)
   a. '0x02 - Secondary boot Application' should then be returned in the Read Mode command (see Section 8.1.1)

Alternatively, Secondary Boot can be accessed from the Main Application by issuing a Switch Application command (see Section 8.1.3) to switch to the secondary boot application if STAY_IN_BOOT is set to 0. The system will also transfer to Secondary Boot if the Main Application firmware is found to be corrupted or missing.

Note

The controller's TSTPT0 requires an analog switch in the schematic for external pullup in order to use STAY_IN_BOOT.

2.2.3 Main Application

The Main Application contained in the embedded software will be reached during the initialization process once the boot process is completed normally and STAY_IN_BOOT is set to 0. This mode is the primary use case for the DLP® system being the mode in which images are projected.

2.2.4 DLPC8445 and DLPC8445V Controller Startup

- The controller outputs a signal called HOST_IRQ. At reset HOST_IRQ starts HIGH, then software will set it LOW to indicate it is BUSY and then will pull it HIGH when it is ready to accept commands.
- Auto-initialization at power-up refers to the process of the Boot ROM reading the controller firmware from the flash device and transferring it to IRAM. Then the ARM processor begins to execute the MainApp that was read from Flash. Auto-initialization is complete once the MainApp has been executed to an extent for command handling from the Application Processor. If a valid Flash device and MainApp are not found in the Flash, the BootRom will assert HOST_IRQ high so that I2C or USB commands can be sent to program or reprogram the Flash.
- The controller initialization typically completes (HOST_IRQ goes high) within 500ms of RESETZ being asserted high. However, this time may vary depending on the software version and the contents of the user configurable auto initialization batch file that is executed by the controller at power-up.

3 Software Overview

The controller contains an Arm® Cortex R4F processor along with additional functional blocks for image processing and chipset control. TI provides software as firmware to be stored in an SPI flash memory connected to the controller. The firmware consists of the MainApp code (used by the Arm processor) along with other configuration and operational data required by the system for normal operation. The controller and its accompanying DLP chipset components require this proprietary software to operate.

The firmware must be programmed into the SPI flash memory. The DLPC8445 and DLPC8445V downloads the main application from Flash into the Arm processor instruction RAM at power-up and also periodically accesses the operational data in Flash during runtime. The available controller functions depend on the firmware version in Flash. Different firmware is required for different chipset combinations (such as when using different DMD or PMIC devices). Visit the applicable controller's product folder on ti.com and go to the DLP Pico Firmware Selector, or contact TI for the latest firmware.

3.1 Interface Protocol

3.1.1 Supported Interfaces

The communication interfaces supported for the controller include a serial data bus conforming to the Philips I^{2}C specification up to 400kHz, USB 2.0, and UART (as a debug port) interfaces. The Phillips I^{2}C and USB interfaces support commands while UART only displays the debug log. In addition to control commands, serial flash programming is also supported over these interfaces.

3.1.2 I^{2}C Target

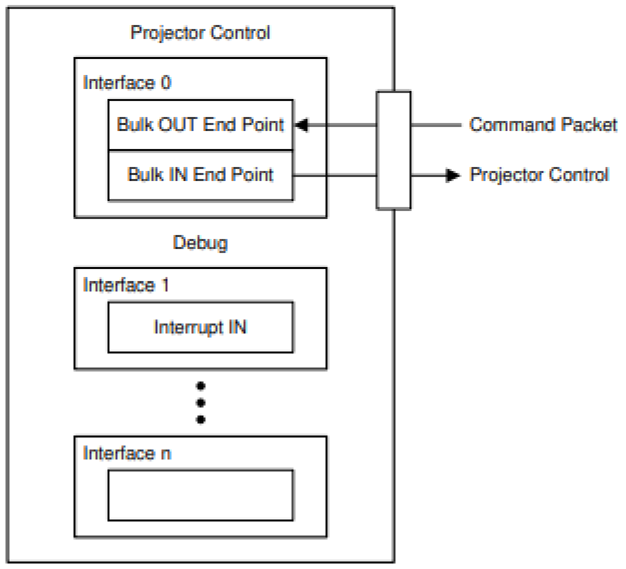
While writing to the controller operating in the I^{2}C target configuration, the first byte following the start condition should be the controller device write address (36h). It is possible to change the device address to either 34h or 36h. This should be indicated correctly to the controller through a TSTPT pin (TSTPT_2) that gets latched up at the startup (If the pin is set LOW then the device address is 36h otherwise if it is set as High then the device address is 34h). The communication will go through if only if TSTPT is set to the value that is matching the address that the user sets in the DLP Tool Suite.

The remaining bytes are sent as specified in the Command Protocol section below. While reading from the controller in I^{2}C target configuration, the first byte following the start condition should be controller device write address +1 (37h default) followed by header and opcode bytes as explained later in the document.

3.1.3 USB

The DLPC8445 and DLPC8445V controller has USB OTG2.0-compliantt hardware. When connected to a USB host, the controller configures as USB device (target) mode operating at high speed (480Mbit/s). The controller enumerates one of the interfaces as a generic WinUSB device with two bulk endpoints. A USB bulk transfer sends the command and response packets through these endpoints. The OUT endpoint is used for the command packet and the IN endpoint is used for the response packet. The USB transfer size can vary from 1 byte to 512 bytes. When the host sends the USB IN request, the controller responds with NAK until there is a response packet ready from the software.

Figure 3-1. USB Core



3.1.4 UART Settings

The following settings are recommended to access the controller debug log over UART:

- Baud Rate: 115200
- Data Bits: 7
- Stop Bits: One
- Parity: None
- Flow Control: None

4 Acronyms

This guide uses the follow acronyms:

Table 4-1. Acronyms used

Acronym	Full Word (Description)
ALPF	Active Lines Per Frame
APPL	Active Pixels Per Line
CCA	Color Coordinate Adjustment
CFI	Common Flash Interface
CLK	Clock
CRC	Cyclic Redundancy Check
DB	DynamicBlack™
DLPA	DLP Power Management IC
DMD	Digital Mircomirror Device
DSI	Display Serial Interface (video interface)
HDR	High Dynamic Range
HLG	Hybrid Log Gamma (HDR transfer function)
HSG	Hue Saturation Gain
Hsync	Horizontal Sync
HW	Hardware

Table 4-1. Acronyms used (continued)

Acronym	Full Word (Description)
I^2C	Inter-Integrated Circuit (command interface)
WRP	Warping
LUT	Look-up Table
3DLR	3D Left/Right Synchronization
PCC	Primary Color Correction
PMIC	Power Management Integrated Circuit
PQ	Perceptual Quantizer (HDR transfer function)
RGB	Red Green Blue (color space)
SPI	Serial Peripheral Interface (command interface)
SSP	Synchronous Serial Port
TPG	Test Pattern Generator
TPPL	Total Pixels Per Line
UART	Universal Asynchronous Receiver-Transmitter
Vsync	Vertical Sync
WPC	White Point Correction
XPR	Extended Pixel Resolution (pixel shifting technology)
VBO	V-by-One
YCbCr	(color space)

5 Command Protocol

This section describes the command protocol implemented for DLPC8445 and DLPC8445V. This is the protocol to be used by the application processor to control DLPC8445 and DLPC8445V using any of the supported commands. This protocol is applicable across all supported peripheral interfaces.

This protocol specifies a flexible length header. The minimum header length is one byte. The first header byte indicates how to interpret the remaining bytes such as opcode, data, and checksum (for error detection). There is also a destination parameter in the header that directs the command to different entities within the projector application.

Use this flexible header length method for application that require a minimum of overhead bytes can opt for the one-byte header. For a more robust application, configure a larger header that includes data length and/or checksum.

5.1 Command Packet

The command packet defines the packet format to follow when commands are sent to the controller. Fields that are always present are indicated in **bold**, and optional fields are indicated in normal font.

The definition of which fields are present is based on the 1-byte header field. The length field is mandatory if a command is defined as having variable data size.

Table 5-1. Command Packet Format

Field	Size (bytes)	Description
Header	1	See Table 5-2 table below
Opcode	1	Command opcode

Table 5-1. Command Packet Format (continued)

Field	Size (bytes)	Description
Length	2 or 0 based on data length present field in the header	Length of the command data in bytes following this byte. Checksum is not included in length. For example, when length being set as 10 means there are 10 bytes of data after this length field. LSB of length shall be sent first followed by MSB.
Data	0-511 (total maximum of 512 bytes in the whole message including header and checksum)	Parameters/data

Checksum	1 or 0 (optional as checksum present field of header)	Checksum of all bytes in the message including header bytes. Fletcher's checksum is implemented as below:
		uint32 SimpleChecksum = 0;
		uint32 SumofSumChecksum = 0;
		uint08 *Addr = (uint08 *) StartAddress;
		while (NumBytes--)
		{
		SimpleChecksum += *Addr++;
		SumofSumChecksum += SimpleChecksum;
		}

Table 5-2. Command Header Byte

Bits	Field Name	Values
0:2	Destination	Numerical destination code of the applicable command
3	Opcode Length	0 = One byte opcode
4	Datalength Present	1 = Length field present in the extended header 0 = No length field present
5	Checksum Present	1 = Checksum present after data bytes 0 = Checksum not present
6	Reply Requested	1 = Device will send a response packet to every write command. This field is applicable only for write commands. 0 = Response packet not sent for write commands
7	Read Command	1 = Read Command 0 = Write Command

Write responses are optional as described in the command header description above. If response is requested, it is imperative to read the response (both Write Response and Read Response) immediately following the respective Command Packet. The response of a command is lost as soon as the controller receives another set of bytes from the host.

An example of a command packet for the Read LED Enable command (see Section 9.6.2) can be seen below:

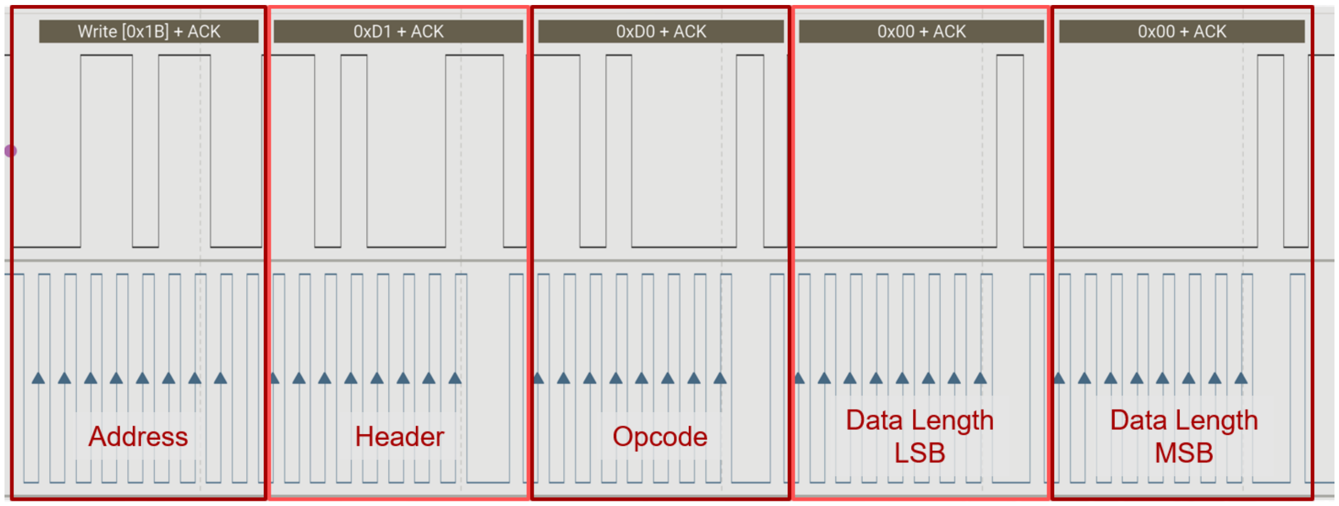


Figure 5-1. Command Packet Example

5.2 Response Packet

The response packet is the format in which the controller replies to the host. The response packet format is followed for both write responses and read responses. For write commands, the response packet is sent only if the "reply requested" bit is set in the command header. The controller matches the response header to the same format as the incoming command packet header. There is however an exception, if the response packet is for a command that expects variable number of data bytes, the response packet will always include the length field (irrespective of whether the command packet had length mentioned or not). Kindly refer to *Support for Variable Data Size* section for information related to variable sized commands.

Similar to the definition of command packet, fields in **bold** represents fields that are always present..

Table 5-3. Response Packet Format

Field	Size (bytes)	Description
Header	1	See Table 5-4 table below
Length	2 or 0 (Optional as per Data Length Preset field in the header)	Length of the command data in bytes following this byte. The checksum is not included in length. For example, when the length is set as 10 means there are 10 bytes of data after this length field. LSB of length shall be sent first followed by MSB.
Data	0-511 (total of max 512 bytes in the whole message including header and checksum)	Response data bytes depend on the command code. If error bit in the header is set, there will only be a single data byte. This byte will indicate the error code that caused the command to nack. The error code definitions are listed in the Table 5-5 table below.
Checksum	1 or 0 (optional as per Checksum Present field of header byte)	Checksum of all bytes in the message including header bytes (Fletcher's checksum)

Table 5-4. Response Header Byte

Bits	Field Name	Values
0:2	Destination	Numerical destination code of the applicable command.
3	Reserved	N/A
4	Datalength Present	1 = Length field present in the extended header 0 = No length field
5	Checksum Present	1 = Checksum present after data bytes 0 = Checksum not present

Table 5-4. Response Header Byte (continued)

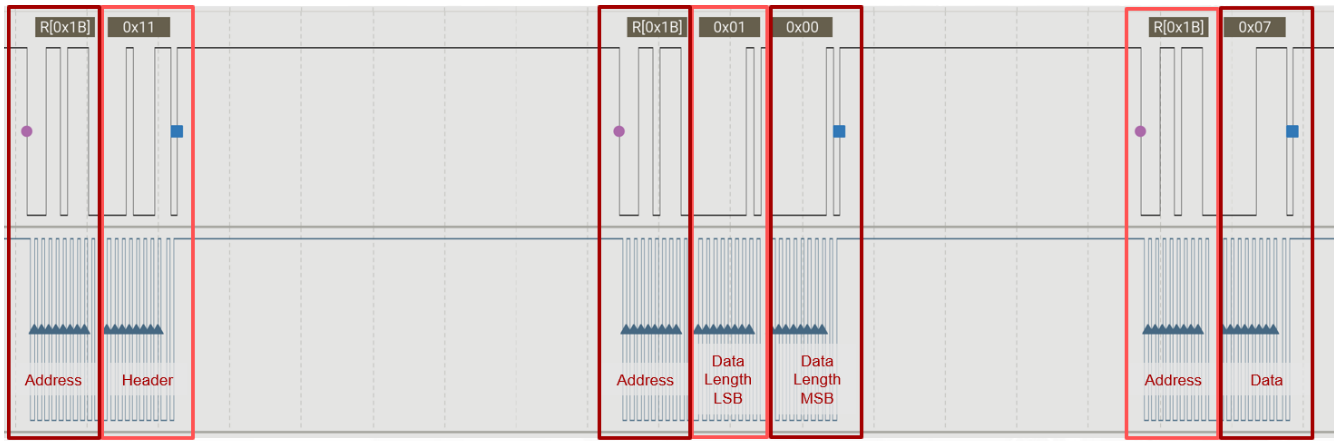
Bits	Field Name	Values
6	Error	1 = Error. First data byte will have the error code that gives more information about the failure 0 = No error
7	Busy	1 = System busy/response not ready 0 = Response ready Applicable only for I^{2}C based communication

Table 5-5. Error Code Definitions

Error Code	Meaning
1	Invalid destination
2	Invalid/Unknown command
3	Invalid length
4	Allocated buffer is not enough to store a command
5	Length information missing for a variable sized command
6	Checksum mismatch
7	Timeout error
8	Read not supported
9	Write not supported
10	Execution failed
11	Invalid response length
12	Buffer Full

An example of a response packet for Read LED Enable (see Section 9.6.2) can be seen below:

Figure 5-2. Response Packet Example



5.3 Error Handling and Recovery

As all physical interfaces support the same protocol, not all start conditions will be the same. Also, depending on payload size, a command packet may be sent over multiple packets. It is important for the controller to know the start of a command to be able to parse and execute the command successfully. This means the host and the controller should always be in sync as will be the case if both the host and the controller reset and power on together. However, if an error occurs on either side, or if one of host/controller asynchronously resets, the sync is lost. Since the start conditions specific to physical interfaces are not monitored, another mechanism to recover when such an error occurs is needed. To support this use case, the controller monitors the time of arrival of each group of bytes. If any group of bytes comes outside of a defined timeout (750 ms) compared to the last group, it is treated as the start of a new command.

The timeout is always measured from the last received group of bytes and not from the group of bytes where it encountered an error. This means if the host keeps sending commands one after the other without a timeout, all of it is discarded.

It is valid to include multiple commands in a single group or send commands back to back without waiting for the defined timeout. Both of these cases are controlled by the command handler which executes all such chained commands in the order they are received.

5.4 System Busy—I^{2}C Scenarios

When using the I^{2}C protocol, the target component pulls the clock line low when it needs to indicate it is busy processing and cannot accept any more data from the host. Be aware that when there are multiple target devices on the same bus, the entire communication on the bus gets halted until the busy peripheral component releases the bus. To prevent this undesirable effect, the controller supports the following options for the host:

5.4.1 HOST_IRQ Implementation

A separate controller pin (HOST_IRQ) reports to the host component that the controller is busy or not busy. Upon power-on-reset, the front-end communication device must wait until the signal goes to a HIGH state. A signal that remains LOW continuously indicates a problem with the controller boot-sequence. The source of the problem must be resolve before proceeding.

When a command is sent, HOST_IRQ is pulled LOW until the command completes execution. If the user attempts to send another command while execution of the first command is ongoing, application processor should confirm whether HOST_IRQ is HIGH or LOW and then takes the decision to send the command accordingly. This process ensures that there is no clock stretching, and other devices on the I^{2}C bus are not affected, but it ensures that the command handler is occupied and no other command can be sent at this point.

5.4.2 Short Status Response

When the I^{2}C host requests a data read, the Busy flag (bit 7 in the header byte) indicates the short status. If it is set the controller is busy and does not have a response to send back yet. The host can use the system busy bit as a check for the controller's availability to receive. When this bit is set, the rest of the bits of the response header shall be treated as don't care and no further bytes to be read. The expectation is that the host will keep reading from the controller for a response until this bit is cleared. When that occurs the response header is valid, and the remaining data is as per the command.

If the host abandons a read command midway or sends another command immediately after sending a read command, the response bytes in the controller buffer get discarded and the new command gets processed.

For the USB communication layer, the controller indicates the busy status by NAK response to the read request.

5.5 Support for Variable Data Size

For large data handling commands such as flash download and flash read, the user can allow some commands to support variable number of data bytes. To support this use case the commands that require variable data size is mandated to include length as part of the command packet header. The command handler uses the given length to decode the received command packet and execute correctly. Similar to the command packet the data in the response packet may also be variable. The command handler includes length in the response packet header for such commands.

The command protocol is designed to support commands up to a length of 65535 bytes (2-byte length field). However, due to memory segmentation, the command handler implementation is limited to a maximum size of 512 bytes in a command packet (this includes all bytes in the command such as header, checksum, and so on).

6 Command Descriptions

Please consider these guidelines applicable to all the command descriptions that follow in this document.

- Byte order: Wherever a parameter is specified as more than 1 byte in length, the order in which it must be sent/read is LSB first and MSB last.
- Parameter for read commands: Not all read commands require a read parameter unless specified.
- When the input parameter(s) to a command is in fixed point format, it is specified such as format = s8.2 or format = u12.4 and so on. where s stands for signed and u stands for unsigned.

Fixed-Point Representation:

This representation has a fixed number of bits for integers and fractions. Negative numbers are represented in two's complement format.

Fixed Point representation - [Integer][Fraction]

Example: Assuming the format is signed and using 32-bit format, with16 bits for the integer part and 16 bits for the fractional part. This will be referred to as s15.16 format.

In this case, -43.625 and 43.625 are represented as follows:

[1111111111010101][1010000000000000] = 0xFFD5A000 = -43.625

[0000000000101011][1010000000000000] = 0x002BA000 = +43.625

7 BootROM Commands

The following commands can be executed while in Boot hold.

7.1 BootROM

BootROM Commands Table	
Command	**Section**
Read Boot Hold Reason	Section 7.1.1
Read Flash ID	Section 7.1.2
Read Get Flash Sector Information	Section 7.1.3
Write Unlock Flash For Update	Section 7.1.4
Read Unlock Flash For Update	Section 7.1.5
Write Erase Sector	Section 7.1.6
Write Initialize Flash Read Write Settings	Section 7.1.7
Write Flash Write	Section 7.1.8
Read Flash Write	Section 7.1.9
Read Checksum	Section 7.1.10
Write Full Flash Erase	Section 7.1.11
Write System Type	Section 7.1.12
Read System Type	Section 7.1.13
Write Clear Error History	Section 7.1.14
Read Error History	Section 7.1.15

7.1.1 Read Boot Hold Reason (12h)

This command returns the code that specifies the reason for being in bootloader mode.

7.1.1.1 Return Parameter(s)

Get Boot Hold Reason	
Return Parameter(s)	
Byte	**Description**
Byte 0	Boot Hold Reason 2 = Main Application programming required 4 = Initialization failure 8 = Stay in Boot Pin Set 16 = Invalid Flash Image 32 = Flash communication failure

7.1.2 Read Flash ID (20h)

This command returns the flash device and manufacturer IDs.

7.1.2.1 Return Parameter(s)

Get Flash ID	
Return Parameter(s)	
Byte	**Description**
Byte 0	Manufacturer ID
Byte 1	Device ID
Bytes 2-3	Capacity ID

7.1.3 Read Get Flash Sector Information (21h)

This command returns the flash sector information read from CFI compliant flash device.

7.1.3.1 Return Parameter(s)

Get Get Flash Sector Information	
Return Parameter(s)	
Byte	**Description**
Byte 0	Flash Size 0 = 4 Mb 1 = 8 Mb 2 = 16 Mb 3 = 32 Mb 4 = 64 Mb 5 = 128 Mb 6 = 256 Mb 7 = 512 Mb
Bytes 1-4	Sector Size (bytes)
Bytes 5-6	Number of Sectors

7.1.4 Write Unlock Flash For Update (22h)

This command unlocks the flash update operation (Download, Erase). By default the flash update operations are locked. This is to prevent accidental modification of flash contents. To unlock the pre-defined key shall be send as the unlock code. Calling this command with any other parameter will lock the flash update commands.

7.1.4.1 Write Parameter(s)

Set Unlock Flash For Update	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Unlock 0 = Lock flash update operations 4154802215 = Unlock flash update operations

7.1.5 Read Unlock Flash For Update (22h)

This command returns whether the flash is in unlocked state.

7.1.5.1 Return Parameter(s)

Get Unlock Flash For Update	
Return Parameter(s)	
Byte	**Description**
Byte 0	Is Unlocked Range = 0 to 1 with step size 1

7.1.6 Write Erase Sector (23h)

This command erases the sector of the flash of the given address. This command is a flash update command and requires flash operations to be unlocked using Unlock Flash for Update command. The sector address shall be specified as an offset from flash start address. For example, in a flash device where all sectors are 64KB of size, sector addresses shall be specified as follows:

Sector 0 = 0

Sector 1 = 0x10000

Sector 2 = 0x20000

...

7.1.6.1 Write Parameter(s)

Set Erase Sector	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Sector Address

7.1.7 Write Initialize Flash Read Write Settings (24h)

This command initializes flash read/write operation. This command shall be called before Flash Write or Flash Read command is sent. Note: For Flash Write/Read, the Start Address and Num Bytes set up shall both be multiple of 4.

7.1.7.1 Write Parameter(s)

Set Initialize Flash Read Write Settings	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Start Address to program data to where Offset 0 refers to first byte in the flash, 1 refers to second byte.
Bytes 4-7	This specifies the number of bytes Flash Write command should expect or the number of bytes Flash Read command should return.This number must be a multiple of 4.

7.1.8 Write Flash Write (25h)

This command is used to program data to flash. This command shall be called only after setting the start address and size using the Initialize Flash Read Write Settings command. This command is a flash update command and requires flash operations to be unlocked using Unlock Flash for Update command. Flash Write commands can be chained until the initialized number of bytes are programmed. The BootROM will auto-increment the address and size for each command. Only the initialized number of bytes will be programmed even if more data is provided. It is important to send only multiple of 4 number of bytes per flash write command to ensure all bytes are written. This is done so that all flash writes are optimized as per the multi-word write supported by the flash device. For SPI communication, SSP buffer size is 64 bytes. The Host needs to send 65 bytes (64 bytes data + 1 dummy byte - needed for mode 0 operation). This command supports variable sized payload. Number of bytes to write is always expected to be a multiple of 4 bytes. The command fails if the number of bytes requested to write is not a multiple of 4 bytes.

7.1.8.1 Write Parameter(s)

Set Flash Write	
Write Parameter(s)	
Byte	**Description**
Bytes 0 - *	Data to write to flash memory

7.1.9 Read Flash Write (25h)

This command is used to read data from flash. This command shall be called only after setting the start address and size using the Initialize Flash Read Write command. Flash Read commands can be chained until the initialized number of bytes are returned. The BootROM will auto-increment the address and size for each command. Only the initialized number of bytes will be returned. Calling the function after returning requested data will result in command failure. This command supports variable sized response. Number of bytes to write is always expected to be a multiple of 4 bytes. Command returns error if it is not multiple of 4.

7.1.9.1 Read Parameter(s)

Get Flash Write	
Read Parameter(s)	
Byte	**Description**
Bytes 0-1	Number of bytes to read in this command

7.1.9.2 Return Parameter(s)

Get Flash Write	
Return Parameter(s)	
Byte	**Description**
Bytes 0 - *	The bytes read from the flash

7.1.10 Read Checksum (26h)

This command computes and returns the checksum starting at the given address for the specified number of bytes. Checksum is calculated as below: -

```
uint32 SimpleChecksum = 0;
uint32 SumofSumChecksum = 0;
uint08 *Addr = (uint08 *) StartAddress;

while (NumBytes--)
{
SimpleChecksum += *Addr++;
SumofSumChecksum += SimpleChecksum;
}
```

7.1.10.1 Read Parameter(s)

Get Checksum	
Read Parameter(s)	
Byte	**Description**
Bytes 0-3	Start Address offset for checksum computation where Offset 0 refers to first byte in the flash, 1 refers to second byte.
Bytes 4-7	Number of bytes to compute checksum. This should be a multiple of 4.

7.1.10.2 Return Parameter(s)

Get Checksum	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	Simple additive checksum
Bytes 4-7	Sum of simple additive checksum calculated at each address

7.1.11 Write Full Flash Erase (28h)

This command does a full chip erase. This command is a flash update command and requires flash operations to be unlocked using the Unlock Flash for Update command.

7.1.11.1 Write Parameter(s)

Set Full Flash Erase
Write Parameter(s)
This command has no write parameters.

7.1.12 Write System Type (03h)

This command can be used to enter the desired system type. System type can only be selected once. Subsequent calls to this function are ignored and cannot be switched without a system restart.

7.1.12.1 Write Parameter(s)

Set System Type	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	System type 2118808358 = Flash System
Byte 4	Enable Debug Messages Range = 0 to 1 with step size 1

7.1.13 Read System Type (03h)

This command returns the system type.

7.1.13.1 Return Parameter(s)

Get System Type	
Return Parameter(s)	
Byte	**Description**
Byte 0	System Type Information bits 0-2: System type 0 = Undefined 1 = Flash System bit3: Enable Debug Messages

7.1.14 Write Clear Error History (05h)

This command clears all entries in the error history. The 32-bit parameter is a signature to prevent accidental calls.

7.1.14.1 Write Parameter(s)

Set Clear Error History	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Clear Error History 0xDDCCBBAA = Clear Error History

7.1.15 Read Error History (06h)

This command reads the error history.

7.1.15.1 Return Parameter(s)

Get Error History	
Return Parameter(s)	
Byte	**Description**
Byte 0	Number of Errors

Get Error History	
Return Parameter(s)	
Byte	**Description**
Bytes 1-248	Data bits 0-12: Error Code bit13: Command Error bit14: Operational Error bit15: Initialization Error

8 Common Commands

The following commands can be executed in either boot hold or the main application.

8.1 Common

Common Commands Table	
Command	**Section**
Read Mode	Section 8.1.1
Read Version	Section 8.1.2
Write Switch Application	Section 8.1.3
Read Extended Software Version	Section 8.1.4

8.1.1 Read Mode (00h)

This command returns whether the controller software is in Boot ROM or in the main application.

8.1.1.1 Return Parameter(s)

Get Mode	
Return Parameter(s)	
Byte	**Description**
Byte 0	Mode Info bits 0-2: Application Mode 0 = BootROM 1 = Main Application 2 = Secondary boot Application bits 3-5: Controller Configuration 0 = Single 1 = Reserved 2 = Dual Primary 3 = Dual Secondary 4 = Quad Primary 5 = Quad Secondary 1 6 = Quad Secondary 2 7 = Quad Secondary 3

8.1.2 Read Version (01h)

This command returns the version of the currently active application. The currently active application can be queried using the Get Mode command.

8.1.2.1 Return Parameter(s)

Get Version	
Return Parameter(s)	
Byte	**Description**
Byte 0	Major
Byte 1	Minor
Byte 2	Patch

8.1.3 Write Switch Application (02h)

This command is used to switch between BootROM to Application and vice versa. When a hard reset to the Boot Application is issued and pin TSTPT_0 is held high, the controller will transfer to BootROM. TSTPT_0 will only be sampled for 1us after reset or power on. If TSTPT_0 is low when the command is issued, the controller will transfer to the secondary boot.

WARNING: If command is issued for jumping to Boot Application from Main Application, then returning back to Main Application is not possible without re-programming the Main Application.

8.1.3.1 Write Parameter(s)

Set Switch Application	
Write Parameter(s)	
Byte	**Description**
Byte 0	Application switch options 0 = Boot Application (via reset) 1 = Main Application (via Soft reset) 3 = Secondary boot (from BootROM)

8.1.4 Read Extended Software Version (04h)

This command returns the Pre-Release Info and Commit ID of software version residing in the Controller.

8.1.4.1 Return Parameter(s)

Get Extended Software Version	
Return Parameter(s)	
Byte	**Description**
Byte 0	0 - Production; A - Alpha; B - Beta 0 = Production 10 = Alpha 11 = Beta
Byte 1	0 - Production; 1-255 - Alpha/Beta
Byte 2	0 - Not a test build; 1-255 - Test build number
Bytes 3-9	First 7-chars of Commmit ID

9 Main Application Commands

The following commands can be executed while in the main application.

9.1 System

System Commands Table	
Command	**Section**
Read Controller ID	Section 9.1.1
Read DMD ID	Section 9.1.2
Read PMIC ID	Section 9.1.3
Read DMD Training Results	Section 9.1.4
Write DMD True Global Reset	Section 9.1.5
Read DMD True Global Reset	Section 9.1.6
Read System Errors	Section 9.1.7
Read System Status	Section 9.1.8
Read Flash Version	Section 9.1.9
Read System Temperature	Section 9.1.10
Read Last Command Result	Section 9.1.11
Read DLPA Main Status	Section 9.1.12

9.1.1 Read Controller ID (40h)

This command returns the device ID for the controller(s).

9.1.1.1 Return Parameter(s)

Get Controller ID	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	Controller Device ID
Bytes 4-7	Reserved
Bytes 8-11	Reserved
Bytes 12-15	Reserved

9.1.2 Read DMD ID (41h)

This command returns the device ID and fuse ID for the DMD.

9.1.2.1 Return Parameter(s)

Get DMD ID	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	DMD Device ID
Bytes 4-7	DMD Fuse ID
Bytes 8-11	Reserved 0
Bytes 12-15	Reserved 1

9.1.3 Read PMIC ID (42h)

This command returns the device ID for the PMIC.

9.1.3.1 Return Parameter(s)

Get PMIC ID	
Return Parameter(s)	
Byte	**Description**
Byte 0	PMIC Device ID

9.1.4 Read DMD Training Results (43h)

This command returns the DMD Training results.

9.1.4.1 Read Parameter(s)

Get DMD Training Results	
Read Parameter(s)	
Byte	**Description**
Byte 0	DMD Channel
Byte 1	DMD Pin

9.1.4.2 Return Parameter(s)

Get DMD Training Results	
Return Parameter(s)	
Byte	**Description**
Byte 0	Is Channel Active Val bit0: Is Channel Active
Byte 1	Is Pin Active Val bit0: Is Pin Active
Byte 2	Last Known Good Dll Val bit0: Last Known Good Dll
Bytes 3-6	Bit Result 63 32
Bytes 7-10	Bit Result 31 00
Byte 11	High Pass
Byte 12	Low Pass
Byte 13	Dll Delay
Bytes 14-15	Error Please refer to Section 11 for descriptions of error codes.

9.1.5 Write DMD True Global Reset (44h)

This command sets the display mode to true global. The True Global mode should be set to true only during factory/assembly operation and is primarily designed for DMD protection on systems being tested or assembled at the cost of improved image quality.

Note that true global mode will not take place until the system is rebooted. This means that the system must contain an EEPROM that can store the true global mode setting.

9.1.5.1 Write Parameter(s)

Set DMD True Global Reset	
Write Parameter(s)	
Byte	**Description**
Byte 0	True Global Mode bit0: 0 = True Global Reset Mode Disabled; 1 = True Global Reset Mode Enabled.

9.1.6 Read DMD True Global Reset (44h)

This command returns whether or not true global mode is enabled.

9.1.6.1 Return Parameter(s)

Get DMD True Global Reset
Data returned is in the same format as the Write Parameter(s).

9.1.7 Read System Errors (45h)

This command returns the overall system errors.

9.1.7.1 Return Parameter(s)

Get System Errors	
Return Parameter(s)	
Byte	**Description**
Byte 0	DMD Errors bit0: DMD device ID mismatch with composer project bit1: DMD initialization error bit2: DMD Low speed interface Error bit3: DMD High speed interface Error bit4: DMD Training Error bit5: DMD power down error
Byte 1	Controller Errors bit0: Product configuration failed bit1: DLPC Initialization Error bit2: Sequencer Error bit3: Sequence Selection failed bit4: Temperature Overshoot detected bit5: Sequence Stalled
Byte 2	Chipset Errors bit0: EEPROM initialization Error bit1: DLPA Communication error (If DLPA present) bit2: Illumination Fault Error bit3: Duty Cycle Update Error (Invalid DC) bit4: Reserved1 Error bit5: Reserved2 Error bit6: Reserved3 Error
Byte 3	Data Port Errors bit0: UART port 0 communication error (If Port Enabled) bit1: SSP port 0 communication error (If Port Enabled) bit2: SSP port 1 communication error (If Port Enabled) bit3: I2C port 0 communication error bit4: I2C port 1 communication error bit5: USB port communication error (If Port Enabled)

9.1.8 Read System Status (46h)

This command reads the overall system status from the controller.

9.1.8.1 Return Parameter(s)

Get System Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	System Status bit0: System Init Done bit indicates, system bring up is complete and HOST_IRQ pin is pulled high. bit1: The System Error bit indicates an error for the system (controller chipset or peripheral hardware controlled by the chipset such as LEDs).Details about system errors are available using the Get System Errors command. bit2: The Video Port Error bit indicates invalid input timing on the active video port.Details about video port errors are available using the Source Timing and Errors command.
Byte 1	System Mode bit0: The Actuator Calibration Mode bit indicates that system is currently in actuator calibration mode.Multiple system functionalities will be bypassed in this mode.
Byte 2	Sequencer and LED Status bit0: Sequencer phase lock bit indicates Sequencer Phase is locked with incoming video source Vsync.Phase lock will not happen in case of Splash input or asynchronous input source. bit1: Sequencer frequency lock bit indicates Sequencer Frequency is locked with incoming video source Vsync.Frequency lock will not happen in case of Splash input or asynchronous input source. bit2: Red Enabled bit3: Green Enabled bit4: Blue Enabled

9.1.9 Read Flash Version (48h)

This command reads the version number that uniquely identifies the flash image.

9.1.9.1 Return Parameter(s)

Get Flash Version	
Return Parameter(s)	
Byte	**Description**
Byte 0	Flash Version Major
Byte 1	Flash Version Minor
Byte 2	Flash Version Subminor

9.1.10 Read System Temperature (4Ah)

This command is used to read the system temperature using an external thermistor via the DLPA (if available).

9.1.10.1 Return Parameter(s)

Get System Temperature	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	This is 12-bit number with 0-10 bit giving value and 11th bit giving sign of number.1 means negative and 0 means positive.Remaining bits 15 to 12 will be zero. Value needs to be divided by 10 to get temperature in degree celsius.

9.1.11 Read Last Command Result (4Dh)

This command returns the execution result of the last command.

9.1.11.1 Return Parameter(s)

Get Last Command Result	
Return Parameter(s)	
Byte	**Description**
Byte 0	The most recent command destination
Bytes 1-2	The most recent command ID
Bytes 3-4	The most recent command error code Please refer to Section 11 for descriptions of error codes.

9.1.12 Read DLPA Main Status (4Eh)

This command gets main status of DLPA.

9.1.12.1 Return Parameter(s)

Get DLPA Main Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	PMIC Main Status bit0: Chip temperature more than 123 Celcius bit1: Chip temperature more than 156 Celcius, or violation in V5V0 bit2: VIN more than LOWBATT_SEL 4: 0 bit3: VIN more than UVLO_SEL 4: 0 bit4: DMD Fault bit5: Proj On Init bit6: Illum Fault bit7: PG failures for a LV Supplies

9.2 Color Processing

Color Processing Commands Table	
Command	**Section**
Write HDR Source Configuration	Section 9.2.1
Read HDR Source Configuration	Section 9.2.2
Write System Brightness Range Setting	Section 9.2.3
Read System Brightness Range Setting	Section 9.2.4
Write WPC Enable	Section 9.2.5
Read WPC Enable	Section 9.2.6
Read WPC Duty Cycles	Section 9.2.7
Read WPC Sensor Output	Section 9.2.8
Write Image CCA Coordinates	Section 9.2.9
Read Image CCA Coordinates	Section 9.2.10
Write Image HSG	Section 9.2.11
Read Image HSG	Section 9.2.12
Write Image CCA HSG Enable Mode	Section 9.2.13
Read Image CCA HSG Enable Mode	Section 9.2.14
Write WPC LED Calibration Matrix	Section 9.2.15
Read WPC LED Calibration Matrix	Section 9.2.16

Color Processing Commands Table	
Write WPC Sensor Calibration Matrix	Section 9.2.17
Read WPC Sensor Calibration Matrix	Section 9.2.18
Write WPC Target Manual Mode	Section 9.2.19
Read WPC Target Manual Mode	Section 9.2.20
Write WPC Target Manual Color Point	Section 9.2.21
Read WPC Target Manual Color Point	Section 9.2.22
Read WPC Target Color Point	Section 9.2.23
Read WPC System Color Point	Section 9.2.24

9.2.1 Write HDR Source Configuration (71h)

HDR maps wider brightness and color range of HDR sources to projector brightness and color range. The mapping requires multiple source groups and system groups to define the HDR source and projection device properties respectively. This command sets the source properties and based on this information nearest source group is selected for mapping.

9.2.1.1 Write Parameter(s)

Set HDR Source Configuration	
Write Parameter(s)	
Byte	**Description**
Byte 0	HDR Enable bit0: Enables HDR Processing
Byte 1	Transfer Function 0 = PQ 1 = HLG

Bytes 2-5	Master Display Black Level
	(nits)
	Range = 0.0000 to 10000.0
	Format = u16.16

Bytes 6-9	Master Display White Level
	(nits)
	Range = 0.0000 to 10000.0
	Format = u16.16

Bytes 10-11	Master Display Color Gamut Red x
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 12-13	Master Display Color Gamut Red y
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 14-15	Master Display Color Gamut Green x
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 16-17	Master Display Color Gamut Green y
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 18-19	Master Display Color Gamut Blue x
	Range = 0.0000 to 1.0000
	Format = u1.15

Set HDR Source Configuration	
Write Parameter(s)	
Byte	**Description**

Bytes 20-21	Master Display Color Gamut Blue y
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 22-23	Master Display Color Gamut White x
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 24-25	Master Display Color Gamut White y
	Range = 0.0000 to 1.0000
	Format = u1.15

9.2.2 Read HDR Source Configuration (71h)

This command returns the metadata information.

9.2.2.1 Return Parameter(s)

Get HDR Source Configuration
Data returned is in the same format as the Write Parameter(s).

9.2.3 Write System Brightness Range Setting (73h)

This command sets the system brightness range in nits. These are used in determining the appropriate transfer functions to be applied on the HDR source. This only needs to be set for HDR functionality.

9.2.3.1 Write Parameter(s)

Set System Brightness Range Setting	
Write Parameter(s)	
Byte	**Description**

Bytes 0-3	Minimum Brightness
	(nits)
	Range = 0.0000 to 10000.0
	Format = u16.16

Bytes 4-7	Maximum Brightness
	(nits)
	Range = 0.0000 to 10000.0
	Format = u16.16

9.2.4 Read System Brightness Range Setting (73h)

This command returns currently set HDR system brightness range.

9.2.4.1 Return Parameter(s)

Get System Brightness Range Setting
Data returned is in the same format as the Write Parameter(s).

9.2.5 Write WPC Enable (74h)

This command enables the WPC function.

White Point Correction (WPC) is a function that automatically adjusts the duty cycles for the red, green, and blue LEDs until the target white point is achieved. The target white point for each Look is set in the firmware. Use command Get Look to read the target white point for the active Look.

When enabled, WPC runs continuously. The controller regularly reads measured data from the light sensor and makes updates to the duty cycles. Continuous operation assures that any drift in the white point over time such as due to LED heating is removed and the target white point is always maintained.

For proper convergence and operation of the WPC algorithm, WPC Sensor Calibration Data must be loaded via Set WPC Calibration Data command or via EEPROM before enabling WPC.

Note that the Dynamic Black commands will take precedence over other color processing algorithms.

9.2.5.1 Write Parameter(s)

Set WPC Enable	
Write Parameter(s)	
Byte	**Description**
Byte 0	WPC Settings bit0: 0 - Disable WPC 1 - Enable WPC

9.2.6 Read WPC Enable (74h)

This command returns whether WPC is enable or disabled.

9.2.6.1 Return Parameter(s)

Get WPC Enable
Data returned is in the same format as the Write Parameter(s).

9.2.7 Read WPC Duty Cycles (76h)

This command reads the LED duty cycles after adjustments by the WPC algorithm have been made.

When WPC is enabled via command Set WPC Enable, WPC automatically adjusts the LED duty cycles until the target white point is achieved. A light sensor embedded in the system is used to monitor the illumination light applied to the DMD.

Before reading duty cycles with this command, make sure WPC is enabled.

The target white point, and the target duty cycles that nominally achieve this white point, can be read with command Get Look. To maintain the target white point the actual LED duty cycles used by WPC will vary from the target values as needed to compensate for system opto-mechanical tolerances and LED performance drift with temperature and aging.

9.2.7.1 Return Parameter(s)

Get WPC Duty Cycles	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	Actual Red Duty Cycle Format = u8.8
Bytes 2-3	Actual Green Duty Cycle Format = u8.8
Bytes 4-5	Actual Blue Duty Cycle Format = u8.8

9.2.8 Read WPC Sensor Output (77h)

This command reads the measured output data from the integrating light sensor for red, green, and blue light.

Values returned are those read by the controller directly from registers in the light sensor device. For byte data formats see the applicable sensor device data sheet.

9.2.8.1 Return Parameter(s)

Get WPC Sensor Output	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	Red
Bytes 4-7	Green
Bytes 8-11	Blue

9.2.9 Write Image CCA Coordinates (78h)

This command allows independent adjustment of the primary, secondary and white coordinates. This call will override any CCA settings performed by prior calls.

Note: CCA is not recommended with Color overlap systems due to performace issues, HSG is recommended for color overlap systems.

9.2.9.1 Write Parameter(s)

Set Image CCA Coordinates	
Write Parameter(s)	
Byte	**Description**

Bytes 0-1	Original Coordinate Red x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 2-3	Original Coordinate Red y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 4-5	Original Coordinate Red Lum
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 6-7	Original Coordinate Green x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 8-9	Original Coordinate Green y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 10-11	Original Coordinate Green Lum
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 12-13	Original Coordinate Blue x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 14-15	Original Coordinate Blue y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 16-17	Original Coordinate Blue Lum
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Set Image CCA Coordinates	
Write Parameter(s)	
Byte	**Description**

Bytes 18-19	Original Coordinate White x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 20-21	Original Coordinate White y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 22-23	Original Coordinate White Lum
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 24-25	Target Coordinate Red x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 26-27	Target Coordinate Red y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 28-29	Target Coordinate Red Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 30-31	Target Coordinate Green x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 32-33	Target Coordinate Green y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 34-35	Target Coordinate Green Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 36-37	Target Coordinate Blue x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 38-39	Target Coordinate Blue y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 40-41	Target Coordinate Blue Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 42-43	Target Coordinate Cyan x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 44-45	Target Coordinate Cyan y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 46-47	Target Coordinate Cyan Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Set Image CCA Coordinates	
Write Parameter(s)	
Byte	**Description**

Bytes 48-49	Target Coordinate Magenta x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 50-51	Target Coordinate Magenta y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 52-53	Target Coordinate Magenta Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 54-55	Target Coordinate Yellow x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 56-57	Target Coordinate Yellow y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 58-59	Target Coordinate Yellow Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 60-61	Target Coordinate White x
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 62-63	Target Coordinate White y
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

Bytes 64-65	Target Coordinate White Gain
	Range = 0.0 to 1.99996948242 with step size 0.00003051757
	Format = u1.15

9.2.10 Read Image CCA Coordinates (78h)

This command returns the current color coordinate configuration.

9.2.10.1 Return Parameter(s)

Get Image CCA Coordinates
Data returned is in the same format as the Write Parameter(s).

9.2.11 Write Image HSG (79h)

This command applies the given hue, saturation and gain values for all colors. It does not affect colors having a gain of zero.

Note: This call will override any CCA settings performed by prior calls.

9.2.11.1 Write Parameter(s)

Set Image HSG	
Write Parameter(s)	
Byte	**Description**

Bytes 0-1	HSG Red Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 2-3	HSG Red Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 4-5	HSG Red Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Bytes 6-7	HSG Green Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 8-9	HSG Green Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 10-11	HSG Green Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Bytes 12-13	HSG Blue Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 14-15	HSG Blue Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 16-17	HSG Blue Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Bytes 18-19	HSG Cyan Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 20-21	HSG Cyan Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 22-23	HSG Cyan Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Bytes 24-25	HSG Magenta Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 26-27	HSG Magenta Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 28-29	HSG Magenta Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Set Image HSG	
Write Parameter(s)	
Byte	**Description**

Bytes 30-31	HSG Yellow Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 32-33	HSG Yellow Saturation
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 34-35	HSG Yellow Hue
	Range = -1.0 to 1.0 with step size 0.00006103515
	Format = s2.14

Bytes 36-37	HSG White Red Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 38-39	HSG White Green Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

Bytes 40-41	HSG White Blue Gain
	Range = 0.0 to 1.99993896485 with step size 0.00006103515
	Format = s2.14

9.2.12 Read Image HSG (79h)

This command returns the currently applied hue, saturation and gain values for all the colors. If gain for a color is zero then the HSG is not applied on the color.

9.2.12.1 Return Parameter(s)

Get Image HSG
Data returned is in the same format as the Write Parameter(s).

9.2.13 Write Image CCA HSG Enable Mode (7Bh)

This command sets the PCC hardware Enable Mode which is needed by CCA/HSG.

9.2.13.1 Write Parameter(s)

Set Image CCA HSG Enable Mode	
Write Parameter(s)	
Byte	**Description**
Byte 0	PCC Enable bit0: CCA HSG Enable
Byte 1	Current PCC Algorithm 0 = No PCC 1 = CCA 2 = HSG

9.2.14 Read Image CCA HSG Enable Mode (7Bh)

This command sets the PCC hardware Enable Mode which is needed by CCA/HSG.

9.2.14.1 Return Parameter(s)

Get Image CCA HSG Enable Mode
Data returned is in the same format as the Write Parameter(s).

9.2.15 Write WPC LED Calibration Matrix (7Ch)

This command sets the pre-computed LED calibration matrix.

9.2.15.1 Write Parameter(s)

Set WPC LED Calibration Matrix	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Byte 0-3 - Matrix val at (0,0) (Format = s1.30) Format = s2.30
Bytes 4-7	Byte 4-7 - Matrix val at (0,1) (Format = s1.30) Format = s2.30
Bytes 8-11	Byte 8-11 - Matrix val at (0,2) (Format = s1.30) Format = s2.30
Bytes 12-15	Byte 12-15 - Matrix val at (1,0) (Format = s1.30) Format = s2.30
Bytes 16-19	Byte 16-19 - Matrix val at (1,1) (Format = s1.30) Format = s2.30
Bytes 20-23	Byte 20-33 - Matrix val at (1,2) (Format = s1.30) Format = s2.30
Bytes 24-27	Byte 24-27. - Matrix val at (2,0) (Format = s1.30) Format = s2.30
Bytes 28-31	Byte 28-31 - Matrix val at (2,1) (Format = s1.30) Format = s2.30
Bytes 32-35	Byte 32-35 - Matrix val at (2,2) (Format = s1.30) Format = s2.30

9.2.16 Read WPC LED Calibration Matrix (7Ch)

This command returns the pre-computed LED calibration matrix.

9.2.16.1 Return Parameter(s)

Get WPC LED Calibration Matrix
Data returned is in the same format as the Write Parameter(s).

9.2.17 Write WPC Sensor Calibration Matrix (7Dh)

This command sets the pre-computed sensor calibration matrix.

9.2.17.1 Write Parameter(s)

Set WPC Sensor Calibration Matrix	
Write Parameter(s)	
Byte	**Description**
Bytes 0-3	Byte 0-3 - Matrix val at (0,0) (Format = s8.23) Format = s9.23

Set WPC Sensor Calibration Matrix	
Write Parameter(s)	
Byte	**Description**
Bytes 4-7	Byte 4-7 - Matrix val at (0,1) (Format = s8.23) Format = s9.23
Bytes 8-11	Byte 8-11 - Matrix val at (0,2) (Format = s8.23) Format = s9.23
Bytes 12-15	Byte 12-15 - Matrix val at (1,0) (Format = s8.23) Format = s9.23
Bytes 16-19	Byte 16-19 - Matrix val at (1,1) (Format = s8.23) Format = s9.23
Bytes 20-23	Byte 20-33 - Matrix val at (1,2) (Format = s8.23) Format = s9.23
Bytes 24-27	Byte 24-27. - Matrix val at (2,0) (Format = s8.23) Format = s9.23
Bytes 28-31	Byte 28-31 - Matrix val at (2,1) (Format = s8.23) Format = s9.23
Bytes 32-35	Byte 32-35 - Matrix val at (2,2) (Format = s8.23) Format = s9.23

9.2.18 Read WPC Sensor Calibration Matrix (7Dh)

This command gets the sensor calibration matrix.

9.2.18.1 Return Parameter(s)

Get WPC Sensor Calibration Matrix
Data returned is in the same format as the Write Parameter(s).

9.2.19 Write WPC Target Manual Mode (87h)

This command enables WPC Target Manual mode to use WPC Target Manual Color Point at run-time.

When this mode is enabled, all target color points specified in the project will be ignored. Software will set only the user specified target manual color point until the manual mode is reset using this same command.

9.2.19.1 Write Parameter(s)

Set WPC Target Manual Mode	
Write Parameter(s)	
Byte	**Description**
Byte 0	WPC Manual Mode Enable bit0: 0 - Disabled 1 - Enabled

9.2.20 Read WPC Target Manual Mode (87h)

This command returns whether WPC Target Manual mode is enabled.

9.2.20.1 Return Parameter(s)

Get WPC Target Manual Mode
Data returned is in the same format as the Write Parameter(s).

9.2.21 Write WPC Target Manual Color Point (88h)

This command sets the target color point which will be used in WPC Target Manual Mode.

9.2.21.1 Write Parameter(s)

Set WPC Target Manual Color Point	
Write Parameter(s)	
Byte	**Description**

Bytes 0-1	Chromatic x
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 2-3	Chromatic y
	Range = 0.0000 to 1.0000
	Format = u1.15

9.2.22 Read WPC Target Manual Color Point (88h)

This command gets the target color point which will be used in WPC Target Manual Mode.

9.2.22.1 Return Parameter(s)

Get WPC Target Manual Color Point
Data returned is in the same format as the Write Parameter(s).

9.2.23 Read WPC Target Color Point (89h)

This command reads the target white point color coordinates.

Values returned give the current target white point by the algorithm. The target white point for each Look is set in the firmware. Use command Get Look to read the target white point for the active Look.

9.2.23.1 Return Parameter(s)

Get WPC Target Color Point	
Return Parameter(s)	
Byte	**Description**

Bytes 0-1	Actual Chromatic x coordinate
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 2-3	Actual Chromatic y coordinate
	Range = 0.0000 to 1.0000
	Format = u1.15

9.2.24 Read WPC System Color Point (8Ah)

This command reads the white point color coordinates as derived from embedded light sensor data.

Before reading the white point with this command, WPC Sensor Calibration Data must be available.

The target white point for each Look is set in the firmware. Use command Get WPC Target Color Point or Get Look to read the target white point for the active Look.

9.2.24.1 Return Parameter(s)

Get WPC System Color Point	
Return Parameter(s)	
Byte	**Description**

Bytes 0-1	Actual Chromatic x coordinate
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 2-3	Actual Chromatic y coordinate
	Range = 0.0000 to 1.0000
	Format = u1.15

Bytes 4-5	Actual Luminance Y coordinate

9.3 TPG

TPG Commands Table	
Command	**Section**
Write TPG Pre Defined Timings Queued	Section 9.3.1
Read TPG Pre Defined Timings Queued	Section 9.3.2
Write TPG Frame Rate	Section 9.3.3
Read TPG Frame Rate	Section 9.3.4
Write TPG Pre Defined Pattern	Section 9.3.5
Read TPG Pre Defined Pattern	Section 9.3.6
Write TPG Border	Section 9.3.7
Read TPG Border	Section 9.3.8
Write TPG Solid Field	Section 9.3.9
Read TPG Solid Field	Section 9.3.10
Write TPG Horizontal Ramp	Section 9.3.11
Read TPG Horizontal Ramp	Section 9.3.12
Write TPG Vertical Ramp	Section 9.3.13
Read TPG Vertical Ramp	Section 9.3.14
Write TPG Horizontal Lines	Section 9.3.15
Read TPG Horizontal Lines	Section 9.3.16
Write TPG Diagonal Lines	Section 9.3.17
Read TPG Diagonal Lines	Section 9.3.18
Write TPG Vertical Lines	Section 9.3.19
Read TPG Vertical Lines	Section 9.3.20
Write TPG Grid	Section 9.3.21
Read TPG Grid	Section 9.3.22
Write TPG Checkerboard	Section 9.3.23
Read TPG Checkerboard	Section 9.3.24
Write TPG Colorbars	Section 9.3.25
Write TPG Multi Color Horizontal Ramp	Section 9.3.26
Read TPG Multi Color Horizontal Ramp	Section 9.3.27
Write TPG Fixed Step Horizontal Ramp	Section 9.3.28
Read TPG Fixed Step Horizontal Ramp	Section 9.3.29
Write TPG Diamond Diagonal Lines	Section 9.3.30
Read TPG Diamond Diagonal Lines	Section 9.3.31

9.3.1 Write TPG Pre Defined Timings Queued (A0h)

This command selects a pre-defined TPG timing that is stored in flash. This command will validate and set the frame rate, active resolution, and blankings.

This command must be followed with a Write Execute Display command to be applied.

9.3.1.1 Write Parameter(s)

Set TPG Pre Defined Timings Queued	
Write Parameter(s)	
Byte	**Description**
Byte 0	Index of the pre-defined TPG timings stored in flash image.

9.3.2 Read TPG Pre Defined Timings Queued (A0h)

This command selects a pre-defined TPG timing that is stored in flash. This command will validate and set the frame rate, active resolution, and blankings.

9.3.2.1 Return Parameter(s)

Get TPG Pre Defined Timings Queued
Data returned is in the same format as the Write Parameter(s).

9.3.3 Write TPG Frame Rate (A1h)

This command specifies the frame rate to be used when a test pattern generator (TPG) image is displayed.

9.3.3.1 Write Parameter(s)

Set TPG Frame Rate	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Frame rate of test pattern (Hz) Range = 9 to 240 with step size 1

9.3.4 Read TPG Frame Rate (A1h)

This command returns frame rate in Hz for current test pattern.

9.3.4.1 Return Parameter(s)

Get TPG Frame Rate
Data returned is in the same format as the Write Parameter(s).

9.3.5 Write TPG Pre Defined Pattern (A2h)

This command will set one of the pre-defined test patterns stored in flash. The function selects a pattern to load from flash into the test pattern generator hardware. The information retrieved from the flash includes pattern definition, color definition, and the resolution. The Set Execute Display command must be called to switch the display mode from other modes to TPG prior to or after this command.

9.3.5.1 Write Parameter(s)

Set TPG Pre Defined Pattern	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Predefined test pattern number to be displayed

9.3.6 Read TPG Pre Defined Pattern (A2h)

This command returns the current selection for pre-defined test patterns.

9.3.6.1 Return Parameter(s)

Get TPG Pre Defined Pattern
Data returned is in the same format as the Write Parameter(s).

9.3.7 Write TPG Border (A3h)

This command enables a white border around the test pattern of given width. This is applicable only when TPG is selected as display source.

9.3.7.1 Write Parameter(s)

Set TPG Border	
Write Parameter(s)	
Byte	**Description**
Byte 0	Border Width in no. of pixels Range = 0 to 63 with step size 1

9.3.8 Read TPG Border (A3h)

The commands returns the TPG border width in number of pixels.

9.3.8.1 Return Parameter(s)

Get TPG Border
Data returned is in the same format as the Write Parameter(s).

9.3.9 Write TPG Solid Field (A4h)

This command sets color for solid field test pattern by selecting the intensity of each primary color.

9.3.9.1 Write Parameter(s)

Set TPG Solid Field	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Solid Color Red Value Range = 0 to 1023 with step size 1
Bytes 2-3	Solid Color Green Value Range = 0 to 1023 with step size 1
Bytes 4-5	Solid Color Blue Value Range = 0 to 1023 with step size 1

9.3.10 Read TPG Solid Field (A4h)

This command returns color for current solid field set by test patter generator. If Solid Field is not enabled via TPG, an error will be returned.

9.3.10.1 Return Parameter(s)

Get TPG Solid Field
Data returned is in the same format as the Write Parameter(s).

9.3.11 Write TPG Horizontal Ramp (A5h)

This command sets the Horizontal Ramp test pattern and its related parameters.

9.3.11.1 Write Parameter(s)

Set TPG Horizontal Ramp	
Write Parameter(s)	
Byte	**Description**
Byte 0	1 = TPG Horizontal Ramp
Byte 1	Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Not Applicable. Set 0
Bytes 3-4	Ramp Step 1 = Increment by 1 every 2 pixels 2 = Increment by 1 every pixel 3 = Increment by 2 every pixel 5 = Increment by 1 every 4 pixels
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.12 Read TPG Horizontal Ramp (A5h)

This command gets the Horizontal Ramp test pattern and its related parameters.

9.3.12.1 Return Parameter(s)

Get TPG Horizontal Ramp
Data returned is in the same format as the Write Parameter(s).

9.3.13 Write TPG Vertical Ramp (A5h)

This command sets the Vertical Ramp test pattern and its related parameters.

9.3.13.1 Write Parameter(s)

Set TPG Vertical Ramp	
Write Parameter(s)	
Byte	**Description**
Byte 0	2 = TPG Vertical Ramp

Set TPG Vertical Ramp	
Write Parameter(s)	
Byte	**Description**
Byte 1	Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Not Applicable. Set 0
Bytes 3-4	Ramp Step 1 = Increment by 1 every line 2 = Increment by 2 every line 3 = Increment by 4 every line
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.14 Read TPG Vertical Ramp (A5h)

This command gets the Vertical Ramp test pattern and its related parameters.

9.3.14.1 Return Parameter(s)

Get TPG Vertical Ramp
Data returned is in the same format as the Write Parameter(s).

9.3.15 Write TPG Horizontal Lines (A5h)

This command sets the Horizontal Lines test pattern and its related parameters.

9.3.15.1 Write Parameter(s)

Set TPG Horizontal Lines	
Write Parameter(s)	
Byte	**Description**
Byte 0	3 = TPG Horizontal Lines
Byte 1	Foreground Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White

Set TPG Horizontal Lines	
Write Parameter(s)	
Byte	**Description**
Byte 2	Background Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Bytes 3-4	Line Width
Bytes 5-6	Distance Between Lines
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.16 Read TPG Horizontal Lines (A5h)

This command gets the Horizontal Lines test pattern and its related parameters.

9.3.16.1 Return Parameter(s)

Get TPG Horizontal Lines
Data returned is in the same format as the Write Parameter(s).

9.3.17 Write TPG Diagonal Lines (A5h)

This command sets the Diagonal Lines test pattern and its related parameters.

9.3.17.1 Write Parameter(s)

Set TPG Diagonal Lines	
Write Parameter(s)	
Byte	**Description**
Byte 0	4 = TPG Diagonal Lines
Byte 1	Foreground Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Background Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Bytes 3-4	Distance Between Lines - (Should be 1 less than power of 2)

Set TPG Diagonal Lines	
Write Parameter(s)	
Byte	**Description**
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.18 Read TPG Diagonal Lines (A5h)

This command gets the Diagonal Lines test pattern and its related parameters.

9.3.18.1 Return Parameter(s)

Get TPG Diagonal Lines
Data returned is in the same format as the Write Parameter(s).

9.3.19 Write TPG Vertical Lines (A5h)

This command sets the Vertical Lines test pattern and its related parameters.

9.3.19.1 Write Parameter(s)

Set TPG Vertical Lines	
Write Parameter(s)	
Byte	**Description**
Byte 0	5 = TPG Vertical Lines
Byte 1	Foreground Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Background Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Bytes 3-4	Distance Between Lines
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.20 Read TPG Vertical Lines (A5h)

This command gets the Vertical Lines test pattern and its related parameters.

9.3.20.1 Return Parameter(s)

Get TPG Vertical Lines
Data returned is in the same format as the Write Parameter(s).

9.3.21 Write TPG Grid (A5h)

This command sets the Grid Lines test pattern and its related parameters.

9.3.21.1 Write Parameter(s)

Set TPG Grid	
Write Parameter(s)	
Byte	**Description**
Byte 0	6 = TPG Grid
Byte 1	Foreground Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Background Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Bytes 3-4	Horizontal Width of the lines
Bytes 5-6	Vertical Width of the lines
Bytes 7-8	Horizontal Distance Between Lines
Bytes 9-10	Vertical Distance Between Lines

9.3.22 Read TPG Grid (A5h)

This command gets the Grid Lines test pattern and its related parameters.

9.3.22.1 Return Parameter(s)

Get TPG Grid
Data returned is in the same format as the Write Parameter(s).

9.3.23 Write TPG Checkerboard (A5h)

This command sets the Checkerboard test pattern and its related parameters. In the case where the desired pattern does not evenly divide across the DMD, there may be apparent mis-alignments along the border.

9.3.23.1 Write Parameter(s)

Set TPG Checkerboard	
Write Parameter(s)	
Byte	**Description**
Byte 0	7 = TPG Checkerboard
Byte 1	Color of the top left checker 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Color of the next checker 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Bytes 3-4	No. of horizontal checkers
Bytes 5-6	No. of vertical checkers
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.24 Read TPG Checkerboard (A5h)

This command gets the Checkerboard test pattern and its related parameters.

9.3.24.1 Return Parameter(s)

Get TPG Checkerboard
Data returned is in the same format as the Write Parameter(s).

9.3.25 Write TPG Colorbars (A5h)

This command sets the Colorbars test pattern and its related parameters.

9.3.25.1 Write Parameter(s)

Set TPG Colorbars	
Write Parameter(s)	
Byte	**Description**
Byte 0	8 = TPG Colorbars
Byte 1	Not Applicable. Set 0
Byte 2	Not Applicable. Set 0
Bytes 3-4	Not Applicable. Set 0
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0

Set TPG Colorbars	
Write Parameter(s)	
Byte	**Description**
Bytes 9-10	Not Applicable. Set 0

9.3.26 Write TPG Multi Color Horizontal Ramp (A5h)

This command sets the Multi Color Horizontal Ramp test pattern and its related parameters.

9.3.26.1 Write Parameter(s)

Set TPG Multi Color Horizontal Ramp	
Write Parameter(s)	
Byte	**Description**
Byte 0	9 = TPG Multi Color Horizontal Ramp
Byte 1	Not Applicable. Set 0
Byte 2	Not Applicable. Set 0
Bytes 3-4	Ramp Step 1 = Increment by 1 every 2 pixels 2 = Increment by 1 every pixel 3 = Increment by 2 every pixel 5 = Increment by 1 every 4 pixels
Bytes 5-6	Not Applicable. Set 0
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.27 Read TPG Multi Color Horizontal Ramp (A5h)

This command gets the Multi Color Horizontal Ramp test pattern and its related parameters.

9.3.27.1 Return Parameter(s)

Get TPG Multi Color Horizontal Ramp
Data returned is in the same format as the Write Parameter(s).

9.3.28 Write TPG Fixed Step Horizontal Ramp (A5h)

This command sets the Fixed Step Horizontal Ramp test pattern and its related parameters.

9.3.28.1 Write Parameter(s)

Set TPG Fixed Step Horizontal Ramp	
Write Parameter(s)	
Byte	**Description**
Byte 0	10 = TPG Fixed Step Horizontal Ramp
Byte 1	Color for Test Pattern 0 = Black 1 = Red 2 = Green 3 = Yellow 4 = Blue 5 = Magenta 6 = Cyan 7 = White
Byte 2	Not Applicable. Set 0

Set TPG Fixed Step Horizontal Ramp	
Write Parameter(s)	
Byte	**Description**
Bytes 3-4	Initial Intensity Value Range = 0 to 1023 with step size 1
Bytes 5-6	No. of horizontal steps
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.29 Read TPG Fixed Step Horizontal Ramp (A5h)

This command gets the Fixed Step Horizontal Ramp test pattern and its related parameters.

9.3.29.1 Return Parameter(s)

Get TPG Fixed Step Horizontal Ramp
Data returned is in the same format as the Write Parameter(s).

9.3.30 Write TPG Diamond Diagonal Lines (A5h)

This command sets the Diamond Diagonal Lines test pattern and its related parameters.

9.3.30.1 Write Parameter(s)

Set TPG Diamond Diagonal Lines	
Write Parameter(s)	
Byte	**Description**
Byte 0	11 = TPG Diamond Diagonal Lines
Byte 1	Forward Diagonal Start Color for Test Pattern 1 = Red start, then Blue, then Green 2 = Green start, then Red, then Blue 4 = Blue start, then Green, then Red
Byte 2	Background Diagonal Fixed Color for Test Pattern 1 = Red 2 = Green 4 = Blue
Bytes 3-4	Double Line Mode enable: 0 = Normal Line Mode 1 = Double Line Mode
Bytes 5-6	Distance Between Lines - (Should be 1 less than power of 2)
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.3.31 Read TPG Diamond Diagonal Lines (A5h)

This command gets the Diamond Diagonal lines test pattern and its related parameters.

9.3.31.1 Return Parameter(s)

Get TPG Diamond Diagonal Lines	
Return Parameter(s)	
Byte	**Description**
Byte 0	11 = TPG Diamond Diagonal Lines

Get TPG Diamond Diagonal Lines	
Return Parameter(s)	
Byte	**Description**
Byte 1	Forward Diagonal Start Color for Test Pattern 1 = Red start, then Blue, then Green 2 = Green start, then Red, then Blue 4 = Blue start, then Green, then Red
Byte 2	Background Diagonal Fixed Color for Test Pattern 1 = Red 2 = Green 4 = Blue
Bytes 3-4	Double Line Mode enable: 0 = Normal Line Mode 1 = Double Line Mode
Bytes 5-6	Distance Between Lines - (Should be 1 less than power of 2)
Bytes 7-8	Not Applicable. Set 0
Bytes 9-10	Not Applicable. Set 0

9.4 Source

Source Commands Table	
Command	**Section**
Read Input Source Status	Section 9.4.1
Read Source Timings And Errors	Section 9.4.2
Write Enable Three D	Section 9.4.3
Read Enable Three D	Section 9.4.4
Write External Source Sync Polarity	Section 9.4.5
Read External Source Sync Polarity	Section 9.4.6
Write VBO Lane Configuration	Section 9.4.7
Read VBO Lane Configuration	Section 9.4.8
Write VBO Configuration	Section 9.4.9
Read VBO Configuration	Section 9.4.10
Read VBO Status	Section 9.4.11
Read Frame CRC	Section 9.4.12
Write VRR Enable Queued	Section 9.4.13
Read VRR Enable Queued	Section 9.4.14

9.4.1 Read Input Source Status (B1h)

This command returns the current video source and the status of the source being displayed.

The command indicates whether the system is displaying an internal source, if it is trying to detect an external source, if it has detected an external source, or if it is in standby mode. This command can be used by the host processor to query the status at any point in time and take the necessary action. It is expected that the host will use this command after setting up the required source and ensure that the source validity is established before setting it up for display or reading the detected timings.

9.4.1.1 Return Parameter(s)

Get Input Source Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	The source currently being displayed. 0 = External Source 1 = TPG 2 = Splash
Byte 1	The status of source. 0 = Error in Source Configuration 1 = Detecting source 2 = Source Locked

9.4.2 Read Source Timings And Errors (B2h)

This command reads the source timings and errors for the currently selected source as detected by the hardware. For internal sources (TPG, Splash), it reports the timings as generated by the hardware. For these sources there will no timing errors as the timing selection command validates the timings before allowing user to select it. For external source this command will return the parameters as detected. Some of these parameters are directly by the hardware whereas some of them are computed in the software based on the hardware detected parameters. These are the timings as seen by the system when a source is connected. If the system detects any invalid parameter or faces any measurement error, it will report the value detected and flag the error. This command is designed to never return fail so that we can get the system detected parameters at any point of time. If system is in standby mode or in the middle of source detection, the Get Source Status Command will report this status. Under normal scenarios this command should be used only when the Get Source Status Command shows a valid source status. This command can, however, be used to check for any anomalies or errors if the Get Source status command is stuck in external source detection state.

The horizontal and vertical back porches may be less than the front end configuration by a value of 1. This is due to Vsync width and Hsync width being extrapolated from these blankings.

9.4.2.1 Return Parameter(s)

Get Source Timings And Errors	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	Pixel Clock Rate (Hz)
Bytes 4-5	Active Pixels Per Line (Pixels)
Bytes 6-7	Active Lines Per Frame (Lines)
Bytes 8-9	Frame Rate (Hz)
Bytes 10-13	H Sync Rate (Hz)
Bytes 14-15	Vertical Front Porch (Lines)
Bytes 16-17	Vertical Back Porch (Lines)
Bytes 18-19	Vertical Sync Width (Lines)

Get Source Timings And Errors	
Return Parameter(s)	
Byte	**Description**
Bytes 20-21	Horizontal Front Porch (Pixels)
Bytes 22-23	Horizontal Back Porch (Pixels)
Bytes 24-25	Horizontal Sync Width (Pixels)
Bytes 26-27	Total Pixels Per Line (Pixels)
Bytes 28-29	Total Lines Per Frame (Lines)
Bytes 30-31	Timing Errors bit0: Invalid APPL bit1: Invalid ALPF bit2: Invalid Horizontal Blanking bit3: Invalid Vertical Blanking bit4: Invalid Hsync Width bit5: Invalid Vsync Width bit6: Invalid Clock bit7: Unstable TPPL bit8: Unstable Active Area bit9: System Measurement Error

9.4.3 Write Enable Three D (B3h)

This command enables the 3D image handling and sync functionality in the controller.

For 3D inputs to the active video port, left (L) and right (R) eye frames are input on alternate controller input frames. 3D frame inputs can be received on V-by-One. The controller receives a L/R input sync signal (on a GPIO) to identify L vs. R input frames. For 3D synchronization, there are no words to be decoded from data received over V-by-One, or FPD-Link.

The controller stores left and right images as needed by 3D stereoscopic glasses. In these glasses, one eye sees a dark shudder while the other sees the DMD's projected image. As images are loaded into the DMD, light from the left and right frames is sent sequentially to the screen. The controller outputs sync information to tell the glasses which image is left and which is right.

Frame rates accepted by the controller when using DLP Link are 100+/-2Hz and 120+/-2Hz.

When DLP Link is not used, the full range of controller input frame rates supported for non-3D applications is supported for 3D L/R frame sequential inputs. For example, the L/R frame sequential input can be 240Hz for a 1080p DMD when using a single controller.

9.4.3.1 Write Parameter(s)

Set Enable Three D	
Write Parameter(s)	
Byte	**Description**
Byte 0	Enable 3D Configuration bit0: TRUE = Enable 3D Processing FALSE = Disable 3D Processing bits 1-2: Dominant Frame 0 = Left Frame Dominance 1 = Right Frame Dominance 2 = Undefined Frame Dominance

9.4.4 Read Enable Three D (B3h)

This command returns the current state of 3D image handling in the controller.

9.4.4.1 Return Parameter(s)

Get Enable Three D
Data returned is in the same format as the Write Parameter(s).

9.4.5 Write External Source Sync Polarity (B4h)

This command sets the input polarity detection mode for VSync and HSync signals for all external sources. The controller requires Sync signals to have active high polarity. The software can auto-detect and correct the polarity. The user should choose Automatic Mode to use this feature. Automatic Mode is the default mode. Otherwise, user can choose manual mode and provide the input polarities. The provided input polarities take effect only in manual mode. It will be used by hardware to correct the polarities.

9.4.5.1 Write Parameter(s)

Set External Source Sync Polarity	
Write Parameter(s)	
Byte	**Description**
Byte 0	Source Sync Polarity Params bit0: Set as True if polarities are manually provided. If False, then polarities areauto-detected bit1: Current VSync Polarity bit2: Current HSync Polarity

9.4.6 Read External Source Sync Polarity (B4h)

This command returns the input polarity detection mode for VSync and HSync signals for all external sources. For Automatic Mode it will return the polarities detected by the hardware as current VSync and HSync Polarity. For Manual Mode it will return the polarities provided by the user.

9.4.6.1 Return Parameter(s)

Get External Source Sync Polarity	
Return Parameter(s)	
Byte	**Description**
Byte 0	Source Sync Polarity bit0: Set as True if polarities are manually provided. If False, then polarities areauto-detected bit1: Current VSync Polarity bit2: Current HSync Polarity

9.4.7 Write VBO Lane Configuration (BAh)

This command is used to set separate V-by-One swizzle setting based on lane configuration.

The Set VBO Configuration command must follow this command for the lane swizzle to take place. The Set Execute Display command must follow the VBO commands for these settings to be updated.

9.4.7.1 Write Parameter(s)

Set VBO Lane Configuration	
Write Parameter(s)	
Byte	**Description**
Byte 0	Num Lanes 1 = One Lane 2 = Two Lane 4 = Four Lane 8 = Eight Lane
Byte 1	Tx Lane 0 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 2	Tx Lane 1 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 3	Tx Lane 2 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 4	Tx Lane 3 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7

Set VBO Lane Configuration	
Write Parameter(s)	
Byte	**Description**
Byte 5	Tx Lane 4 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 6	Tx Lane 5 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 7	Tx Lane 6 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 8	Tx Lane 7 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7

9.4.8 Read VBO Lane Configuration (BAh)

This command is used to get seperate V-by-One swizzle setting based on lane configuration.

9.4.8.1 Read Parameter(s)

Get VBO Lane Configuration	
Read Parameter(s)	
Byte	**Description**
Byte 0	Num Lanes 1 = One Lane 2 = Two Lane 4 = Four Lane 8 = Eight Lane

9.4.8.2 Return Parameter(s)

Get VBO Lane Configuration	
Return Parameter(s)	
Byte	**Description**
Byte 0	Tx Lane 0 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 1	Tx Lane 1 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 2	Tx Lane 2 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 3	Tx Lane 3 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 4	Tx Lane 4 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7

Get VBO Lane Configuration	
Return Parameter(s)	
Byte	**Description**
Byte 5	Tx Lane 5 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 6	Tx Lane 6 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7
Byte 7	Tx Lane 7 0 = VBO Lane 0 1 = VBO Lane 1 2 = VBO Lane 2 3 = VBO Lane 3 4 = VBO Lane 4 5 = VBO Lane 5 6 = VBO Lane 6 7 = VBO Lane 7

9.4.9 Write VBO Configuration (BBh)

This command configures the characteristics of the V-by-One (VBO) external video source.

The byte mode is the color depth set by the front end.

The data map is the color format that is transmitted from the video front end.

The data rate frequency range is the bit rate window. If not defined by the front end, the bit rate window can be found by the following equation by multiplying the total pixel clock by the byte mode number multiplied by 10.

The number of lanes is typically set by the video source. Note the source clock frequency, link clock frequency per lane, and the link transfer rate (or bit rate) per lane limitations when selecting a number of lanes. The source clock frequency is calculated by multiplying the total pixel clock by the number of bytes per pixel (byte mode) and dividing by the number of lanes. The link frequency per lane is the total pixel clock divided by the number of lanes. The link transfer rate per lane is the data rate frequency divided by the number of lanes.

The minimum and maximum specifications for these parameters can be found in the V-by-One interface timing requirements section of the controller datasheet. The VBO configuration needs to be adjusted to meet the source frame timing requirements as well as the parameters mentioned above. The source frame timing requirements can also be found in the controller datasheet.

The Horizontal Total of the video feed must be a multiple of 8.

This command must be followed with a Write Execute Display command to be properly applied.

9.4.9.1 Write Parameter(s)

Set VBO Configuration	
Write Parameter(s)	
Byte	**Description**
Byte 0	Byte Mode 0 = 8bit mode (=3Byte mode) 1 = 10bit mode (=4Byte mode) 2 = 12bit mode (=5Byte mode)
Byte 1	Data Map 0 = 36bpp RGB 1 = 36bpp Y Cb Cr 444 2 = 30bpp RGB 3 = 30bpp Y Cb Cr 444 4 = 24bpp RGB 5 = 24bpp Y Cb Cr 444 6 = 32bpp Y Cb Cr 422 7 = 24bpp Y Cb Cr 422 8 = 20bpp Y Cb Cr 422 9 = 16bpp Y Cb Cr 422
Byte 2	Data Rate Frequency Range 0 = 4 Gbps To 2 Gbps 1 = 2 Gbps To 600 Mbps
Byte 3	The number of lanes to use

9.4.10 Read VBO Configuration (BBh)

This command returns the characteristics of the VBO source.

9.4.10.1 Return Parameter(s)

Get VBO Configuration	
Return Parameter(s)	
Byte	**Description**
Byte 0	Byte Mode 0 = 8bit mode (=3Byte mode) 1 = 10bit mode (=4Byte mode) 2 = 12bit mode (=5Byte mode)
Byte 1	Data Map 0 = 36bpp RGB 1 = 36bpp Y Cb Cr 444 2 = 30bpp RGB 3 = 30bpp Y Cb Cr 444 4 = 24bpp RGB 5 = 24bpp Y Cb Cr 444 6 = 32bpp Y Cb Cr 422 7 = 24bpp Y Cb Cr 422 8 = 20bpp Y Cb Cr 422 9 = 16bpp Y Cb Cr 422
Byte 2	Frange 0 = 4 Gbps To 2 Gbps 1 = 2 Gbps To 600 Mbps

Get VBO Configuration	
Return Parameter(s)	
Byte	**Description**
Byte 3	The number of lanes in use

9.4.11 Read VBO Status (BCh)

This command returns the status of the V-by-One source lock.

9.4.11.1 Return Parameter(s)

Get VBO Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	VBO Lock Status bit0: Whether VBO is CDR locked. bit1: Whether VBO is data locked.

9.4.12 Read Frame CRC (BDh)

This command returns the CRC of the displayed image.

9.4.12.1 Return Parameter(s)

Get Frame CRC	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	The CRC of the image. CRC seed = 0xA5A5A5A5

9.4.13 Write VRR Enable Queued (BEh)

This command is used to enable/disable VRR (Variable Refresh Rate).

It is a queued command, so must Set Execute Display after Enabling/Disabling.

9.4.13.1 Write Parameter(s)

Set VRR Enable Queued	
Write Parameter(s)	
Byte	**Description**
Byte 0	VRR Settings bit0: 0 - Disable VRR 1 - Enable VRR

9.4.14 Read VRR Enable Queued (BEh)

This command returns if VRR (Variable Refresh Rate) is enabled or disabled.

9.4.14.1 Return Parameter(s)

Get VRR Enable Queued
Data returned is in the same format as the Write Parameter(s).

9.5 Splash

Splash Commands Table	
Command	**Section**

Splash Commands Table	
Write Splash Screen Select	Section 9.5.1
Read Splash Screen Select	Section 9.5.2
Read Splash Screen Header	Section 9.5.3

9.5.1 Write Splash Screen Select (C1h)

This command specifies which splash screen is to be displayed among the splash screen images stored in the firmware image.

The splash screen is read over the flash interface and sent down the controller image processing path once and then stored in the frame buffer. As such, all image processing settings (e.g. image crop, image orientation, display size, splash screen reference number) should be set by the user.

The availability of splash screens is limited by the available space in flash memory, and all splash screens must use landscape orientation.

The minimum splash image size allowed for flash storage is 640 in horizontal resolution and 360 in vertical resolution. The maximum size supported matches the full display resolution available on the DMD, and splash screens are typically set to the full resolution. Full resolution is useful for supporting optical test splash screens.

The user must specify how the splash image is displayed on the screen. Key commands for this are Set Image Crop and Set Display Size.

When issued to the controller this command makes splash screen as the active source being displayed. The controller then stores the specified splash screen reference number, and the controller software also reads the header information from flash for this splash screen and stores this in internal memory.

This command must be followed with a Write Execute Display command.

9.5.1.1 Write Parameter(s)

Set Splash Screen Select	
Write Parameter(s)	
Byte	**Description**
Byte 0	Splash Image Index number

9.5.2 Read Splash Screen Select (C1h)

This command returns the splash image index.

9.5.2.1 Return Parameter(s)

Get Splash Screen Select
Data returned is in the same format as the Write Parameter(s).

9.5.3 Read Splash Screen Header (C2h)

This command returns splash image header info from the index. If there is no splash screen stored in the frame buffer for the selection given by command Set Splash Screen, this is considered an error. The command Get Communication Errors will return invalid command parameter error.

9.5.3.1 Read Parameter(s)

Get Splash Screen Header	
Read Parameter(s)	
Byte	**Description**
Byte 0	Splash Image Index number

9.5.3.2 Return Parameter(s)

Get Splash Screen Header	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	Horizontal Resolution of the Splash Image
Bytes 2-3	Vertical Resolution of the Splash Image
Byte 4	Pixel Format of the Splash Image 0 = Splash image pixel is in 16-bit RGB 1 = Splash image pixel is in 16-bit YCrCb 3 = Splash image pixel is in 24-bit RGB
Byte 5	Compression Type of the Splash Image 0 = No compression 1 = Rle 2 0

9.6 Illumination

Illumination Commands Table	
Command	**Section**
Write LED Enable	Section 9.6.1
Read LED Enable	Section 9.6.2
Write LED Currents	Section 9.6.3
Read LED Currents	Section 9.6.4
Write LED Max Currents	Section 9.6.5
Read LED Max Currents	Section 9.6.6
Read LED Min Currents	Section 9.6.7

9.6.1 Write LED Enable (D0h)

This command enables or disables the individual LEDs.

9.6.1.1 Write Parameter(s)

Set LED Enable	
Write Parameter(s)	
Byte	**Description**
Byte 0	LED Channel Enable bit0: Red bit1: Green bit2: Blue bit3: Reserved 0 bit4: Reserved 1

9.6.2 Read LED Enable (D0h)

This command returns if the LEDs are enabled or disabled.

9.6.2.1 Return Parameter(s)

Get LED Enable
Data returned is in the same format as the Write Parameter(s).

9.6.3 Write LED Currents (D1h)

This command sets the drive levels for each LED. The drive levels are set only if the LED Channel is valid.

The currents for each individual LED can be controlled independently. The limits for the current level set is decided by the minimum and maximum values specified for the system in the flash image. The command will not return an error if the current is outside the expected range. Instead, it will automatically limits to minimum limit if the value is less then the minimum limit and limits to maximum limit if the value is greater then the maximum limit. Reading back the values will guarantee whether it is applied directly or limited against the minimum and maximum. The applicable LED must be enabled for the effect to be seen on the display.

When the color processing algorithms (WPC and Dynamic Black) are disabled, this command directly sets the LED currents (the R, G, and B 10-bit values provided are sent directly to the DLPA PMIC by the controller via SPI). If color processing algorithm is enabled, then this command will not take effect immediately.

When an all-white image is displayed and color processing algorithms are disabled, this command allows the system white point to be adjusted while at the same time establishing the total LED power for the display module.

9.6.3.1 Write Parameter(s)

Set LED Currents	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Red Level
Bytes 2-3	Green Level
Bytes 4-5	Blue Level
Bytes 6-7	Reserved 0
Bytes 8-9	Reserved 1
Bytes 10-11	Reserved 2

9.6.4 Read LED Currents (D1h)

This command gets the drive levels for each LED.

9.6.4.1 Return Parameter(s)

Get LED Currents
Data returned is in the same format as the Write Parameter(s).

9.6.5 Write LED Max Currents (D5h)

This command specifies the maximum 10-bit current allowed for each LED in the display module.

This command sets the maximum LED currents that can be used when DB is enabled or disabled. When DB is enabled, the maximum LED currents may be further limited by the DB intensity-to-current LUTs stored in flash.

This command protects the LEDs from a user accidently setting LED currents higher than the system can handle when using command Set RGB LED Currents.

9.6.5.1 Write Parameter(s)

Set LED Max Currents	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Red
Bytes 2-3	Green
Bytes 4-5	Blue

9.6.6 Read LED Max Currents (D5h)

Data returned is in the same format as Write Parameters for command Set RGB LED Max Currents.

9.6.6.1 Return Parameter(s)

Get LED Max Currents
Data returned is in the same format as the Write Parameter(s).

9.6.7 Read LED Min Currents (D6h)

This command returns the minimum current allowed for each LED in the display module.

9.6.7.1 Return Parameter(s)

Get LED Min Currents	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	Red
Bytes 2-3	Green
Bytes 4-5	Blue

9.7 Display

Display Commands Table	
Command	**Section**
Write Execute Display	Section 9.7.1
Read Execute Display Status	Section 9.7.2
Write Input Image Size Queued	Section 9.7.3
Read Input Image Size Queued	Section 9.7.4
Write Image Crop Queued	Section 9.7.5
Read Image Crop Queued	Section 9.7.6
Write Display Size Queued	Section 9.7.7
Read Display Size Queued	Section 9.7.8
Write Display Image Orientation Queued	Section 9.7.9
Read Display Image Orientation Queued	Section 9.7.10
Write Display Curtain	Section 9.7.11
Read Display Curtain	Section 9.7.12
Write Image Freeze	Section 9.7.13
Read Image Freeze	Section 9.7.14
Write Border Color	Section 9.7.15
Read Border Color	Section 9.7.16

9.7.1 Write Execute Display (E1h)

This command initiates the execution of queued display commands.

Due to an extended setup and reconfiguration time, some display commands have been made into queued commands. This means that the commands will not be executed until the Execute Display Command is called. Grouping multiple queued commands allows for a reduced configuration time.

The queued commands include but are not limited to:

1. Set Input Image Size Queued
2. Set Image Crop Queued
3. Set Display Size Queued
4. Set Pre-Defined Timings Queued
5. Any XPR control commands
6. Set VBO Configuration
7. Write Splash Screen Select

It is recommended to follow this command with Read Execute Display Status to confirm that this has been executed without error.

9.7.1.1 Write Parameter(s)

Set Execute Display
Write Parameter(s)
This command has no write parameters.

9.7.2 Read Execute Display Status (E2h)

This command returns the status of the previous Execute Display command.

9.7.2.1 Return Parameter(s)

Get Execute Display Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	Execute Command State 0 = Not Started 1 = Execution pending after source change 2 = In Queuing Mode 3 = Execution In Progress 4 = Execution Successful 5 = Execution Failed. See ErrorCode
Bytes 1-2	Error Code Please refer to Section 11 for descriptions of error codes.

9.7.3 Write Input Image Size Queued (E3h)

This command specifies the active data size of the internal/external input image to the controller.

The parameter values are 1-based (for example, a value of 1280 pixels specifies 1280 pixels per line).

This command must be followed with a Write Execute Display command to be applied.

9.7.3.1 Write Parameter(s)

Set Input Image Size Queued	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Pixels per line of the active area of the internal/external input Image
Bytes 2-3	Lines per frame of the active area of the internal/external input Image

9.7.4 Read Input Image Size Queued (E3h)

This command returns currently set internal/external input image size.

9.7.4.1 Return Parameter(s)

Get Input Image Size Queued
Data returned is in the same format as the Write Parameter(s).

9.7.5 Write Image Crop Queued (E4h)

This command specifies the cropping applied to input images.

This command applies to all sources including test patterns, splash screens, and external sources.

Cropping is done prior to the scaling function in the controller. As such, the size difference between the cropped image size and displayed image size determines the amount of scaling needed in both dimensions if scaling using WRP is enabled.

This command must be followed with a Write Execute Display command to be applied.

The cropping rules are demonstrated in the following figure.

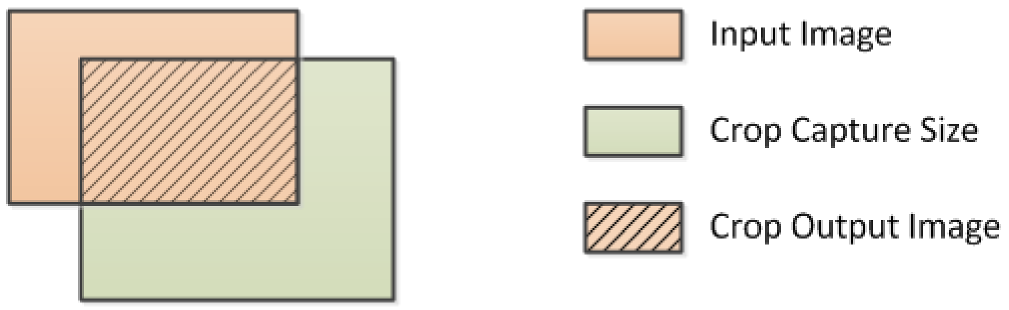


Figure 9-1. Crop Rules when Crop exceeds input size

9.7.5.1 Write Parameter(s)

Set Image Crop Queued	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Capture Start Pixel (0-based, indicates first active pixel of a line)
Bytes 2-3	Capture Start Line (0-based, indicates first active line of a frame)
Bytes 4-5	Pixels per Line (1-based, such that specifying a pixel per line value of 854 indicates 854 pixels to be crop)
Bytes 6-7	Lines per Frame (1-based, such that specifying a lines per frame value of 480 indicates 480 lines to be crop)

9.7.6 Read Image Crop Queued (E4h)

This command returns the cropping applied to input images prior to controller image processing.

9.7.6.1 Return Parameter(s)

Get Image Crop Queued
Data returned is in the same format as the Write Parameter(s).

9.7.7 Write Display Size Queued (E5h)

This command specifies the active image resolution to be output on the display module, specifies the size of the non-keystone corrected, non-warped image to be output from the scalar function. This is the resolution of the rectangular or square displayed image. The controller will determine the amount of scaling needed in x and in y based on the size of the input image. For a source image from the external video port, the controller will measure the size of the source image based on the DATEN signal. However, if the image is cropped by command Set Image Crop, the controller will use this smaller size to be scaled to fill the DMD.

The parameter values are to be 1-based such that a value of 1280 pixels displays 1280 pixels per line.

If keystone correction or warping is enabled, the resulting non-rectangular images on the DMD will all fit within the active display region specified by command Set Display Size.

If the display size exceeds the resolution available on the DMD, this is considered an error and the command does not execute. The display size parameters are checked against the DMD available resolution in both rotation image orientations (non-rotated and rotated), and if the DMD resolution is exceeded in either of these orientations, it is considered an error. The system does not check for proper image orientation setup.

If the source, crop, and display parameter combinations exceed the capabilities of the scalar, the system implements the user request as best it can, and the displayed image may be broken. The user must provide updated parameters to fix the image.

This command must be followed with a Write Execute Display command to be applied.

9.7.7.1 Write Parameter(s)

Set Display Size Queued	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Display Area Pixels Per Line
Bytes 2-3	Display Area Lines Per Frame

9.7.8 Read Display Size Queued (E5h)

This command returns the size of the active image display size.

9.7.8.1 Return Parameter(s)

Get Display Size Queued
Data returned is in the same format as the Write Parameter(s).

9.7.9 Write Display Image Orientation Queued (E6h)

This command specifies the orientation of the image displayed on the DMD.

The command rotates and/or flips the image displayed on the DMD. Rotation can be useful for a portrait source image such as inside a mobile phone. Flips are provided to support ceiling mount and rear projection use cases.

Landscape images typically should not be rotated, but the controller allows this as it may be appropriate for some situations or configurations.

Image rotation is allowed while keystone correction or warping is enabled, though it may not be appropriate for all situations or configurations.

The user is responsible for determining if the resulting image from these commands is acceptable.

User can use the Display Size command to centre the image.

This command must be followed with a Write Execute Display command to be applied.

The following figure shows the long axis flip.

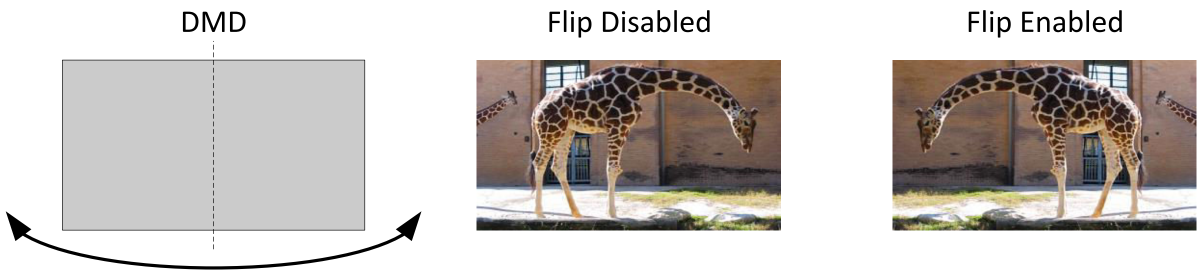


Figure 9-2. Long Axis Flip

The following figure shows the short axis flip.

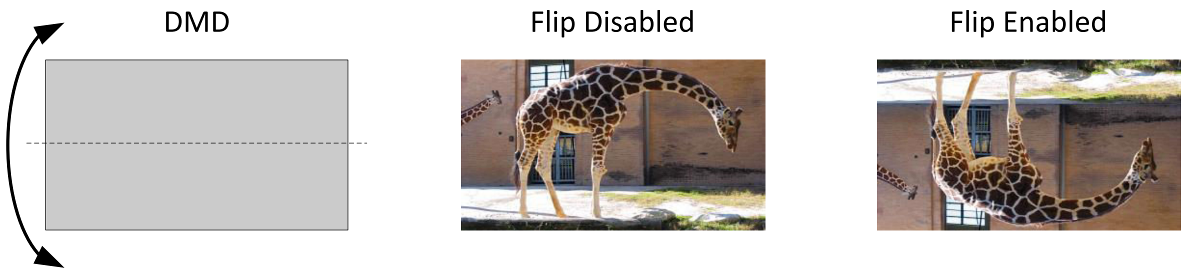


Figure 9-3. Short Axis Flip

9.7.9.1 Write Parameter(s)

Set Display Image Orientation Queued	
Write Parameter(s)	
Byte	**Description**
Byte 0	Orientation bit0: Rotate 0 - No rotation 1 - Minus 90 deg rotation bit1: "Long Axis Image Flip 0 - Image not flipped 1 - Image flipped bit2: "Short Axis Image Flip 0 - Image not flipped 1 - Image flipped

9.7.10 Read Display Image Orientation Queued (E6h)

Data returned is in the same format as Write Parameters for command Set Display Image Orientation.

9.7.10.1 Return Parameter(s)

Get Display Image Orientation Queued
Data returned is in the same format as the Write Parameter(s).

9.7.11 Write Display Curtain (E7h)

This command controls the image curtain displayed on the DMD.

The image curtain fills the entire DMD with a user-specified color. The color specified for the curtain by this command is separate from the border color defined by command Set Border Color.

Please note that scaling or cropped images may affect the border color.

9.7.11.1 Write Parameter(s)

Set Display Curtain	
Write Parameter(s)	
Byte	**Description**
Byte 0	Curtain Settings bit0: Curtain Enable 0 - Curtain disabled 1 - Curtain enabled bits 1-3: "Curtain Color 0 = Black 3 = Blue 4 = Cyan 2 = Green 5 = Magenta 1 = Red 7 = White 6 = Yellow

9.7.12 Read Display Curtain (E7h)

Data returned is in the same format as Write Parameters for command Display Curtain.

9.7.12.1 Return Parameter(s)

Get Display Curtain
Data returned is in the same format as the Write Parameter(s).

9.7.13 Write Image Freeze (E8h)

This command freezes the image whereby the last image received is displayed in every subsequent frame.

The image freeze capability has two main uses. The first use is to simply freeze the current image on the screen. The second use is to assist the user in reducing display artifacts during limited system configuration changes. In this second case, the image is frozen, system changes are made, and the image is unfrozen after completion. When the image is unfrozen, the display shows the most recent input image frames. Input data between the freeze point and the unfreeze point is lost. However, due to the possibility of commands that affect the DMD immediately, it is recommended to disabled LED's during transitions.

The controller software does not freeze or unfreeze the image except when explicitly commanded to by command Set Image Freeze. This is true whether controller software is making updates to the system on its own volition or during any operation commanded via the I2C interface.

An overview of using the Set Image Freeze command is included.

Use of Image Freeze to Reduce On-Screen Artifacts:

Commands that take a long time to process, require a lot a data to be loaded from flash, or change the frame timing of the system may create on-screen artifacts. The Set Image Freeze command can minimize or eliminate these artifacts. The process is:

1. Send a Set Image Freeze command to enable
2. Send commands with the potential to create image
3. Send a Set Image Freeze command to disable

Because commands to the controller process serially, no special timing or delay is required between these commands. The number of commands placed between the freeze and unfreeze should be small, as it is not desirable for the image to be frozen for a long period of time. Caution: Command Set Display Curtain or any operation that requires curtain will override Freeze and the frozen displayed image will be lost. The following operations require a curtain and will override Freeze:

1. Source Type Switch (Standard - XPR - 3D)
2. Switch to Splash display
3. Source Re-lock
4. Switch to Standby Mode

9.7.13.1 Write Parameter(s)

Set Image Freeze	
Write Parameter(s)	
Byte	**Description**
Byte 0	Freeze bit0: 0 - Image freeze disabled 1 - Image freeze enabled

9.7.14 Read Image Freeze (E8h)

Data returned is in the same format as Write Parameters for command Set Image Freeze.

9.7.14.1 Return Parameter(s)

Get Image Freeze
Data returned is in the same format as the Write Parameter(s).

9.7.15 Write Border Color (E9h)

This command specifies the border color displayed on the DMD.

Whenever a displayed image is smaller than the active mirror array on the DMD, the border color is used for all non-image pixels. Some examples using a border include a window box, pillar box, and letterbox image.

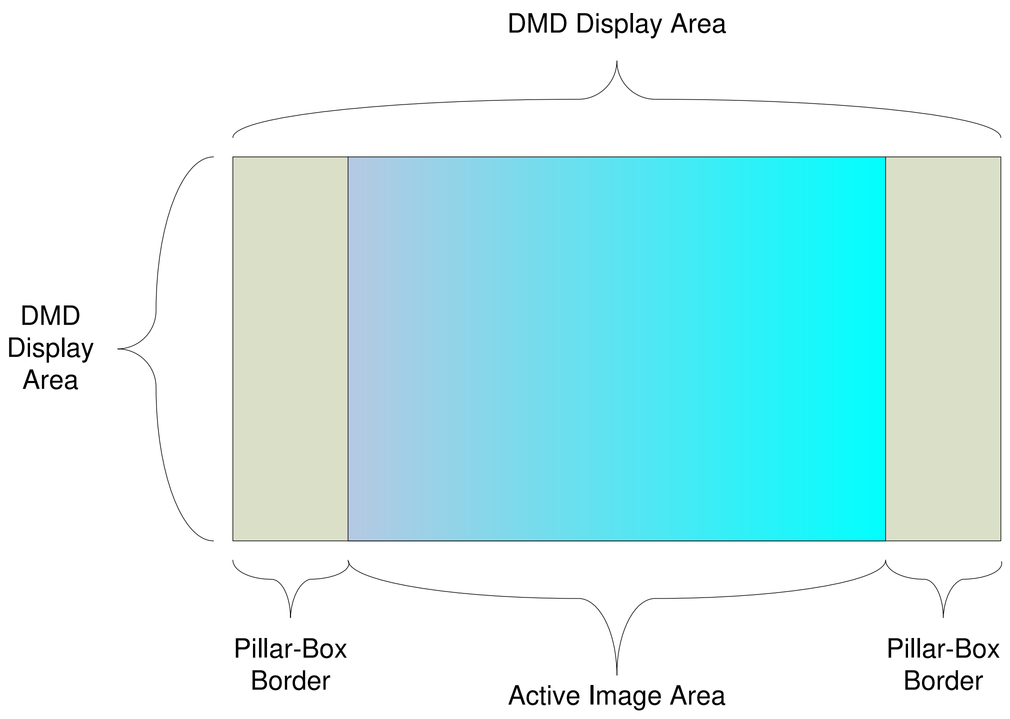


Figure 9-4. Pillar-Box Border Example

9.7.15.1 Write Parameter(s)

Set Border Color	
Write Parameter(s)	
Byte	**Description**
Byte 0	Border Color bits 0-2: Display Border Color 0 = Black 3 = Blue 4 = Cyan 2 = Green 5 = Magenta 1 = Red 7 = White 6 = Yellow

9.7.16 Read Border Color (E9h)

Data returned is in the same format as Write Parameters for command Set Border Color.

9.7.16.1 Return Parameter(s)

Get Border Color
Data returned is in the same format as the Write Parameter(s).

9.8 Sequence

Sequence Commands Table	
Command	**Section**
Read Is Discrete Duty Cycle Supported	Section 9.8.1
Write System Look Index	Section 9.8.2
Read System Look Index	Section 9.8.3

Sequence Commands Table	
Write Color Duty Cycles	Section 9.8.4
Read Color Duty Cycles	Section 9.8.5
Write Discrete Duty Cycle Index	Section 9.8.6
Read Discrete Duty Cycle Index	Section 9.8.7
Read Min Max Duty Cycle Supported	Section 9.8.8
Read LED Illumination Delay	Section 9.8.9

9.8.1 Read Is Discrete Duty Cycle Supported (57h)

This command returns if the current source has discrete Duty cycle supported

9.8.1.1 Return Parameter(s)

Get Is Discrete Duty Cycle Supported	
Return Parameter(s)	
Byte	**Description**
Byte 0	Discrete Duty Cycle Supported bit0: Enable or disable

9.8.2 Write System Look Index (F0h)

This command changes the Look of the displayed image by attempting a Write Look Select value by the user which will trigger to sequence selection. The value should be within the looks specified in the firmware image.

It is recommended to add a delay of approximately 50ms after this command if it is to be followed by a Color Duty Cycle command.

9.8.2.1 Write Parameter(s)

Set System Look Index	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Look Index value for sequence

9.8.3 Read System Look Index (F0h)

This command returns current Look Index value for the sequence selection.

9.8.3.1 Return Parameter(s)

Get System Look Index
Data returned is in the same format as the Write Parameter(s).

9.8.4 Write Color Duty Cycles (F1h)

This command defines duty cycles for illumination light colors. The input values in the u9.23 fixed point format. The lower and upper limits for the duty cycles are built into the firmware image.

9.8.4.1 Write Parameter(s)

Set Color Duty Cycles	
Write Parameter(s)	
Byte	**Description**

Bytes 0-3	Red Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Set Color Duty Cycles	
Write Parameter(s)	
Byte	**Description**

Bytes 4-7	Green Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Bytes 8-11	Blue Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Bytes 12-15	Cyan Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Bytes 16-19	Magenta Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Bytes 20-23	Yellow Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

9.8.5 Read Color Duty Cycles (F1h)

This command returns the duty cycles for illumination light colors.

9.8.5.1 Return Parameter(s)

Get Color Duty Cycles
Data returned is in the same format as the Write Parameter(s).

9.8.6 Write Discrete Duty Cycle Index (F2h)

This command changes the Duty Cycle which will trigger to sequence selection. The value should be within the number of discrete value specified in the firmware image. It will only be valid when Discrete DC suport is enabled with the selected source.

9.8.6.1 Write Parameter(s)

Set Discrete Duty Cycle Index	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Look Index value for sequence

9.8.7 Read Discrete Duty Cycle Index (F2h)

This command returns current Duty Cycle Index value for the sequence selection. The command will only be functional for discrete duty cycle entries and will not be functional for duty cycle range entries.

9.8.7.1 Return Parameter(s)

Get Discrete Duty Cycle Index
Data returned is in the same format as the Write Parameter(s).

9.8.8 Read Min Max Duty Cycle Supported (F4h)

This command returns the minimum and maximum duty cycle that is supported by the selected illuminator.

9.8.8.1 Read Parameter(s)

Get Min Max Duty Cycle Supported	
Read Parameter(s)	
Byte	**Description**
Byte 0	The Illuminator type 0 = Red Color 1 = Green Color 2 = Blue Color 3 = Cyan Color 4 = Magenta Color 5 = Yellow Color

9.8.8.2 Return Parameter(s)

Get Min Max Duty Cycle Supported	
Return Parameter(s)	
Byte	**Description**

Bytes 0-3	Min Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

Bytes 4-7	Max Duty Cycle
	Range = 0.0 to 100 with step size 0.00000008
	Format = u9.23

9.8.9 Read LED Illumination Delay (F5h)

This command returns the delay values programmed in the sequences for all the illuminators in the project.

9.8.9.1 Read Parameter(s)

Get LED Illumination Delay	
Read Parameter(s)	
Byte	**Description**
Byte 0	The Illuminator type 0 = Red Color 1 = Green Color 2 = Blue Color 3 = Cyan Color 4 = Magenta Color 5 = Yellow Color

9.8.9.2 Return Parameter(s)

Get LED Illumination Delay	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	Enable Delay (us)
Bytes 2-3	Rise Time (us)
Bytes 4-5	Disable Delay (us)

Get LED Illumination Delay	
Return Parameter(s)	
Byte	**Description**
Bytes 6-7	Fall Time (us)

9.9 Image Processing

Image Processing Commands Table	
Command	**Section**
Write Dynamic Black Enable	Section 9.9.1
Read Dynamic Black Enable	Section 9.9.2
Write Image Pixel Brightness	Section 9.9.3
Read Image Pixel Brightness	Section 9.9.4
Write Image Pixel Contrast	Section 9.9.5
Read Image Pixel Contrast	Section 9.9.6
Write Degamma Table	Section 9.9.7
Read Degamma Table	Section 9.9.8
Write Image Sharpness	Section 9.9.9
Read Image Sharpness	Section 9.9.10
Write Image CSC Index Value	Section 9.9.11
Read Image CSC Index Value	Section 9.9.12
Write XPR Filter Strength Command	Section 9.9.13
Read XPR Filter Strength Command	Section 9.9.14

9.9.1 Write Dynamic Black Enable (D2h)

This command specifies the Dynamic Black enable or disable.

When Dynamic Black is enabled, LED currents are controlled automatically and Set LED Currents command connot be used to set currents. Commands supported in each mode are shown below.

Dynamic Black disabled - Set RGB LED Enable, Get RGB LED Enable, Set RGB LED Currents, Get RGB LED Currents, Set RGB LED Max Currents, Get RGB LED Max Currents, Get RGB LED Min Currents

Dynamic Black enabled - Set RGB LED Enable, Get RGB LED Enable, Get RGB LED Currents, Set RGB LED Max Currents, Get RGB LED Max Currents, Get RGB LED Min Currents.

Note that the Dynamic Black commands will take precedence over other color processing algorithms and that dynamic content is needed for this algorithm to work. DB will not have any affect on static images such as splash screen.

9.9.1.1 Write Parameter(s)

Set Dynamic Black Enable	
Write Parameter(s)	
Byte	**Description**
Byte 0	Dynamic Black Enable bit0: 0 - Dynamic Black Disable 1 - Dynamic Black Enable

9.9.2 Read Dynamic Black Enable (D2h)

Data returned is in the same format as Write Parameters for command Set Dynamic Black Enable.

9.9.2.1 Return Parameter(s)

Get Dynamic Black Enable
Data returned is in the same format as the Write Parameter(s).

9.9.3 Write Image Pixel Brightness (F6h)

This command controls the pixel brightness by providing an additive gain to each pixel. The brightness value specified affects the red, green and blue components uniformly. For YCbCr images, brightness setting is applied after it is converted to RGB.

9.9.3.1 Write Parameter(s)

Set Image Pixel Brightness	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Brightness

9.9.4 Read Image Pixel Brightness (F6h)

This command returns the currently applied Image Brightness value (offset applied to pixel values). The offset added to red, green, and blue is same. Brightness is applied on YCbCr images after converting it to RGB.

9.9.4.1 Return Parameter(s)

Get Image Pixel Brightness
Data returned is in the same format as the Write Parameter(s).

9.9.5 Write Image Pixel Contrast (F7h)

This command controls the pixel contrast by providing a multiplicative gain to each pixel. The contrast value specified affects the red, green and blue components uniformly. For YCbCr images, contrast setting is applied after it is converted to RGB.

9.9.5.1 Write Parameter(s)

Set Image Pixel Contrast	
Write Parameter(s)	
Byte	**Description**
Byte 0	Contrast (%) Range = 0 to 200 with step size 1

9.9.6 Read Image Pixel Contrast (F7h)

This command returns the currently applied Image Contrast value (gain multiplied to pixel values). The gain is same for red, green, and blue. Contrast is applied on YCbCr images after converting it to RGB.

9.9.6.1 Return Parameter(s)

Get Image Pixel Contrast
Data returned is in the same format as the Write Parameter(s).

9.9.7 Write Degamma Table (F8h)

This command specifies the degamma look-up table (LUT) to be used when displaying images. This command causes the degamma table for given index to be read from flash memory and loaded into the controller's internal degamma LUT.

The table indexes are 0-based (starting from 0). The maximum value can be 254. A value of 255 would disable de-gamma operations.

9.9.7.1 Write Parameter(s)

Set Degamma Table	
Write Parameter(s)	
Byte	**Description**
Byte 0	De-gamma Table Index

9.9.8 Read Degamma Table (F8h)

This command returns currently applied degamma table.

9.9.8.1 Return Parameter(s)

Get Degamma Table
Data returned is in the same format as the Write Parameter(s).

9.9.9 Write Image Sharpness (F9h)

This command configures the sharpness filter. A value of 0 is the least sharp (smoothest), while a value of 31 is the sharpest. This filter is in the back end of the data path, so both video and graphics are affected. TI recommends that the sharpness filters be disabled (sharpness=16) for graphics sources.

9.9.9.1 Write Parameter(s)

Set Image Sharpness	
Write Parameter(s)	
Byte	**Description**
Byte 0	Sharpness value to apply. Range = 0 to 31 with step size 1

9.9.10 Read Image Sharpness (F9h)

The command returns the current sharpness value.

9.9.10.1 Return Parameter(s)

Get Image Sharpness
Data returned is in the same format as the Write Parameter(s).

9.9.11 Write Image CSC Index Value (FAh)

This command controls the applied CSC matrix by setting the currently applied CSC matrix index. All indices can be populated as defined by the customer in the flash.

9.9.11.1 Write Parameter(s)

Set Image CSC Index Value	
Write Parameter(s)	
Byte	**Description**
Byte 0	CSC Index Value
Byte 1	CSC Enable bit0: User CSC Enable

9.9.12 Read Image CSC Index Value (FAh)

This command returns the currently applied CSC matrix index.

9.9.12.1 Return Parameter(s)

Get Image CSC Index Value
Data returned is in the same format as the Write Parameter(s).

9.9.13 Write XPR Filter Strength Command (FBh)

This commands sets the XPR filter segment length.

For more information regarding filter segment length, please refer to the actuator user's guide or contact your actuator manufacturer.

9.9.13.1 Write Parameter(s)

Set XPR Filter Strength Command	
Write Parameter(s)	
Byte	**Description**
Byte 0	Filter Strength setting determines how much of high frequency content is filtered out. Valid range 0-7Setting of 0 means least filtering of high frequency content (sharpest image; more flicker) Setting of 7 means most filtering of high frequency content (smoothest image; least flicker)

9.9.14 Read XPR Filter Strength Command (FBh)

This commands returns the XPR filter segment length.

9.9.14.1 Return Parameter(s)

Get XPR Filter Strength Command
Data returned is in the same format as the Write Parameter(s).

9.10 Blending

Blending Commands Table	
Command	**Section**
Write Blending Function Control	Section 9.10.1
Read Blending Function Control	Section 9.10.2
Write Blendmap Control Points	Section 9.10.3
Read Blendmap Control Points	Section 9.10.4
Write Blendmap Gain Values	Section 9.10.5
Read Blendmap Gain Values	Section 9.10.6
Write Blendmap Offset Values	Section 9.10.7
Read Blendmap Offset Values	Section 9.10.8

9.10.1 Write Blending Function Control (58h)

This command disables or enables EBF processing.

9.10.1.1 Write Parameter(s)

Set Blending Function Control	
Write Parameter(s)	
Byte	**Description**
Byte 0	Blending Function Enable bit0: Enable or disable

9.10.2 Read Blending Function Control (58h)

This command returns the EBF processing enable state.

9.10.2.1 Return Parameter(s)

Get Blending Function Control
Data returned is in the same format as the Write Parameter(s).

9.10.3 Write Blendmap Control Points (59h)

This command takes input of the user defined control points location in horizontal and vertical direction as part of the Blend Map

9.10.3.1 Write Parameter(s)

Set Blendmap Control Points	
Write Parameter(s)	
Byte	**Description**
Bytes 0 - *	Blend Map Horizontal control points position array Number of points in this array equal to 32.These control points are 0 based.
Bytes 0 - *	Blend Map Vertical control points position array Number of points in this array equal to 32.These control points are 0 based.

9.10.4 Read Blendmap Control Points (59h)

This command gets the user defined blend map control points location stored in EEPROM.

9.10.4.1 Return Parameter(s)

Get Blendmap Control Points	
Return Parameter(s)	
Byte	**Description**
Bytes 0-63	Blend map Horizontal control points position array. Number of points in this array equal to 32
Bytes 64-127	Blend map Vertical control points position array. Number of points in this array equal to 32

9.10.5 Write Blendmap Gain Values (5Ah)

This command takes input from the user of Gain values of control points as part of the Blend Map

9.10.5.1 Write Parameter(s)

Set Blendmap Gain Values	
Write Parameter(s)	
Byte	**Description**
Byte 0	Color Channel Select 0 = Green color 1 = Red color 2 = Blue color
Bytes 1-2	Start index in the table for the data to be written
Bytes 3 - *	Gain of control points .The format of input should be such that required gain which is a value between 0 to 1.99 be multiplied by 4096 before passing it in command.

9.10.6 Read Blendmap Gain Values (5Ah)

This command reads from the blend map table already loaded using CMD_WriteBlendMapGainValues command. N Blend map gain values (that does not exceed the command packet size) can be read at a time from anywhere within the table.

9.10.6.1 Read Parameter(s)

Get Blendmap Gain Values	
Read Parameter(s)	
Byte	**Description**
Byte 0	Color Channel Select 0 = Green color 1 = Red color 2 = Blue color
Bytes 1-2	Start index in the Blend map channel gain values from which the data is to be read
Bytes 3-4	Number of entries to be read

9.10.6.2 Return Parameter(s)

Get Blendmap Gain Values	
Return Parameter(s)	
Byte	**Description**
Bytes 0 - *	Selected Color Channel gain values

9.10.7 Write Blendmap Offset Values (5Bh)

This command takes input from the user of Offset values of control points as part of the Blend Map

9.10.7.1 Write Parameter(s)

Set Blendmap Offset Values	
Write Parameter(s)	
Byte	**Description**
Byte 0	Color Channel Select 0 = Green color 1 = Red color 2 = Blue color
Bytes 1-2	Start index in the table for the data to be written

Set Blendmap Offset Values	
Write Parameter(s)	
Byte	**Description**
Bytes 3 - *	Offset of control points.The format of input should be such that the offset values are in the internal floating point format of s1m7e4.

9.10.8 Read Blendmap Offset Values (5Bh)

This command reads from the blend map table already loaded using CMD_WriteBlendMapOffsetValues command. N Blend map Offset values (that does not exceed the command packet size) can be read at a time from anywhere within the table.

9.10.8.1 Read Parameter(s)

Get Blendmap Offset Values	
Read Parameter(s)	
Byte	**Description**
Byte 0	Color Channel Select 0 = Green color 1 = Red color 2 = Blue color
Bytes 1-2	Start index in the Blend map channel Offset values from which the data is to be read
Bytes 3-4	Number of entries to be read

9.10.8.2 Return Parameter(s)

Get Blendmap Offset Values	
Return Parameter(s)	
Byte	**Description**
Bytes 0 - *	Selected Color Channel Offset values

9.11 Peripherals

Peripherals Commands Table	
Command	**Section**
Write User Settings Commit Mode	Section 9.11.1
Read User Settings Commit Mode	Section 9.11.2
Write Use Factory Defaults On Next Power Up	Section 9.11.3
Write Update Lock State	Section 9.11.4
Read Update Lock State	Section 9.11.5
Write Data Invalidate	Section 9.11.6
Write Commit Data	Section 9.11.7
Read Data Operations Status	Section 9.11.8
Read DMD Temperature	Section 9.11.9
Write Calibration Data	Section 9.11.10
Read Calibration Data	Section 9.11.11

9.11.1 Write User Settings Commit Mode (92h)

This command is used to switch between the commit modes - immediate and update via command, during run-time. This command is only applicable if data storage mode is EEPROM. In Immediate mode data will be stored in EEPROM as it is updated. In Command mode updated data will be only stored once the Commit Data command is given.

9.11.1.1 Write Parameter(s)

Set User Settings Commit Mode	
Write Parameter(s)	
Byte	**Description**
Byte 0	Commit Mode Immediate/Command 0 = Immediate 1 = Command

9.11.2 Read User Settings Commit Mode (92h)

This command returns the current user settings commit mode.

9.11.2.1 Return Parameter(s)

Get User Settings Commit Mode
Data returned is in the same format as the Write Parameter(s).

9.11.3 Write Use Factory Defaults On Next Power Up (93h)

When this flag is set, the default factory settings are applied on next power-up without invalidating (erasing) the settings stored in the EEPROM/Flash. Upon power up, change in any setting will not be updated to the EEPROM/Flash irrespective of commit mode. If Data Storage mode is EEPROM and commit mode is immediate, setting this flag will restart the system immediately. If data storage is Flash, the Commit Data command needs to be given and system will not start restart immediately.

9.11.3.1 Write Parameter(s)

Set Use Factory Defaults On Next Power Up	
Write Parameter(s)	
Byte	**Description**
Byte 0	Default Mode bit0: 0 - Normal data loading 1 - On Next power cycle factory defaults will be used.

9.11.4 Write Update Lock State (94h)

This command sets the current data update lock state for EEPROM/FLASH.

When lock is set, all writes to EEPROM/Flash settings and/or calibration data from application software will not be actually written to the EEPROM/Flash. The locked mode is to be used only in factory where user wants to play around with various settings without actually recording them in the EEPROM/Flash. In Normal Use mode, the lock is not supposed to be set.

9.11.4.1 Write Parameter(s)

Set Update Lock State	
Write Parameter(s)	
Byte	**Description**
Byte 0	Lock State bit0: 0 - Update is unlocked 1 - Update is locked

9.11.5 Read Update Lock State (94h)

This command returns the current data update lock state for EEPROM/FLASH.

9.11.5.1 Return Parameter(s)

Get Update Lock State
Data returned is in the same format as the Write Parameter(s).

9.11.6 Write Data Invalidate (99h)

This command invalidates the user settings portion of EEPROM data or calibration portion of EEPROM data or both as per input arguments and restarts the system. If none of the settings or calibration data is selected, then the command does nothing.

9.11.6.1 Write Parameter(s)

Set Data Invalidate	
Write Parameter(s)	
Byte	**Description**
Byte 0	Invalidate User Settings Data bit0: Invalidate User Settings
Byte 1	Invalidate Calibration Data bit0: Invalidate SSI Calibration Data bit1: Invalidate WPC Calibration Data bit2: Invalidate WPC Calibration Matrix Data bit3: Invalidate XPR Calibration Data bit4: Invalidate XPR Waveform Calibration Data bit5: Invalidate Surface Correction Data

9.11.7 Write Commit Data (9Ah)

If data storage mode is EEPROM, then calibration data is written immediately irrespective of Commit Mode. But if commit mode is Command, then user settings will be only stored in eeprom on passing this command. If data storage mode is Flash, then there is no Immediate Commit mode. User setting and calibration data will be only stored when a command is given.

9.11.7.1 Write Parameter(s)

Set Commit Data
Write Parameter(s)
This command has no write parameters.

9.11.8 Read Data Operations Status (9Bh)

The command returns the current working status of EEPROM/flash.

9.11.8.1 Return Parameter(s)

Get Data Operations Status	
Return Parameter(s)	
Byte	**Description**
Byte 0	Write Operation Status bit0: 0 - update can be done 1 - update is disabled bit1: 0 - use factory default disabled 1 - use factory default enabled

Get Data Operations Status	
Return Parameter(s)	
Byte	**Description**
Bytes 1-2	Scratchpad Area Offset
Bytes 3-6	(SizeInBytes)
Byte 7	CommunicationStatus bit0: 0 - unsuccessful 1 - successful
Byte 8	Data Storage Mode 0 = EEPROM 1 = Flash
Byte 9	Update bit0: 0 - There is no pending commit 1 - Data is updated, but commit is pending.

9.11.9 Read DMD Temperature (9Ch)

This command is applicable only if TMP411A temperature sensor is installed in the system. If properly configured, the thermistor should not return a negative value.

A local reading means that the reading is directly at the TMP411. A remote reading would mean that the reading is taking place at the test point pins of the DMD (indirectly at the DMD).

9.11.9.1 Read Parameter(s)

Get DMD Temperature	
Read Parameter(s)	
Byte	**Description**
Byte 0	Defines the temperature sensor from which the TMP411 reads 0 = Local Temp Sensor 1 = Remote Temp Sensor

9.11.9.2 Return Parameter(s)

Get DMD Temperature	
Return Parameter(s)	
Byte	**Description**
Bytes 0-1	value in degree Celcius Range = -256 to 255 with step size 1

9.11.10 Write Calibration Data (9Dh)

This command sets the Calibration data for the selected block.

9.11.10.1 Write Parameter(s)

Set Calibration Data	
Write Parameter(s)	
Byte	**Description**
Byte 0	Calibration Data Block 0 = SSI Data 1 = WPC Calib Data 2 = WPC Calib Matrix Data 3 = XPR Data 4 = XPR Waveform Data 5 = Surface Correction Data
Bytes 1-2	Number of bytes to be written
Bytes 3-4	Offset From Start Of Calib Block
Bytes 5- Number of bytes passed	Data

9.11.11 Read Calibration Data (9Dh)

This command sets the Calibration data for the selected block.

9.11.11.1 Read Parameter(s)

Get Calibration Data	
Read Parameter(s)	
Byte	**Description**
Byte 0	Calibration Data Block 0 = SSI Data 1 = WPC Calib Data 2 = WPC Calib Matrix Data 3 = XPR Data 4 = XPR Waveform Data 5 = Surface Correction Data
Bytes 1-2	Number of bytes to be written
Bytes 3-4	Offset From Start Of Calib Block

9.11.11.2 Return Parameter(s)

Get Calibration Data	
Return Parameter(s)	
Byte	**Description**
Bytes 0- Number of bytes passed	Data

9.12 Warp

Warp Commands Table	
Command	**Section**
Write Warp Feature Control Queued	Section 9.12.1
Read Warp Feature Control Queued	Section 9.12.2
Write Optical Parameters Queued	Section 9.12.3
Read Optical Parameters Queued	Section 9.12.4
Write Keystone Angles Queued	Section 9.12.5
Read Keystone Angles Queued	Section 9.12.6

Warp Commands Table	
Write Keystone Corners Queued	Section 9.12.7
Read Keystone Corners Queued	Section 9.12.8

9.12.1 Write Warp Feature Control Queued (61h)

This command selects warping or keystone correction image processing in the controller.

9.12.1.1 Write Parameter(s)

Set Warp Feature Control Queued	
Write Parameter(s)	
Byte	**Description**
Byte 0	Feature 1 = Keystone
Byte 1	Warp Feature Enable bit0: Enable or disable

9.12.2 Read Warp Feature Control Queued (61h)

Data returned is in the same format as Write Parameters for same command.

9.12.2.1 Read Parameter(s)

Get Warp Feature Control Queued	
Read Parameter(s)	
Byte	**Description**
Byte 0	Feature 1 = Keystone

9.12.2.2 Return Parameter(s)

Get Warp Feature Control Queued	
Return Parameter(s)	
Byte	**Description**
Byte 0	Warp Feature Enable bit0: Enable or disable

9.12.3 Write Optical Parameters Queued (62h)

This command configures the optical parameters for the keystone correction when the throw ratio & the vertical offset for the projector are known.

To adjust images to compensate for pitch and yaw movements of the projector, 1D/2D keystone correction and warping must know the projector's optical throw ratio and optical vertical offset.

9.12.3.1 Write Parameter(s)

Set Optical Parameters Queued	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	Throw Ratio Format = u8.8
Bytes 2-3	Vertical Offset Format = s8.8

9.12.4 Read Optical Parameters Queued (62h)

This command returns the optical parameters currently set.

9.12.4.1 Return Parameter(s)

Get Optical Parameters Queued
Data returned is in the same format as the Write Parameter(s).

9.12.5 Write Keystone Angles Queued (63h)

This command configures the Keystone correction when the pitch, yaw, and roll for the corrected image are known. Keystone correction is used to remove the distortion caused when the projector is not orthogonal to the projection surface (screen). Roll correction is used to remove the distortion caused when the projector is rotated while projecting an image. 3D keystone correction is used to remove the distortion when the projector is both rotated and not orthogonal to the projection surface.

This command provides pitch angle and yaw angle inputs as needed for the controller to perform 1D or 2D keystone correction. This command provides a roll angle input as needed for the controller to perform roll correction. If pitch angle and yaw angle are also input, then 3D keystone correction is performed.

For the projector’s pitch, yaw and roll angles to be properly comprehended by the controller software, the projector’s optical throw ratio and optical vertical offset must be entered using command Set Optical Parameters.

The keystone correction feature is enabled when command Set Warp Feature Control settings are selected and enabled.

9.12.5.1 Write Parameter(s)

Set Keystone Angles Queued	
Write Parameter(s)	
Byte	**Description**

Bytes 0-1	Pitch angle in degrees
	Range = -128 to 127.9960375 with step size 0.00390625
	Format = s8.8

Bytes 2-3	Yaw angle in degrees Set to 0 for 1Dcorrection
	Range = -128 to 127.9960375 with step size 0.00390625
	Format = s8.8

Bytes 4-5	Roll angle in degrees Set to 0 for1D/2D correction
	Range = -128 to 127.9960375 with step size 0.00390625
	Format = s8.8

9.12.6 Read Keystone Angles Queued (63h)

This command returns the keystone configuration parameters currently set.

9.12.6.1 Return Parameter(s)

Get Keystone Angles Queued
Data returned is in the same format as the Write Parameter(s).

9.12.7 Write Keystone Corners Queued (64h)

This command configures the 2D keystone correction when the corners of the corrected image are known. Keystone correction is used to remove the distortion caused when the projector is not orthogonal to the projection surface (screen). For the effects to take place, the Keystone feature has to be enabled.

9.12.7.1 Write Parameter(s)

Set Keystone Corners Queued	
Write Parameter(s)	
Byte	**Description**
Bytes 0-1	X position of the top left corner
Bytes 2-3	Y position of the top left corner
Bytes 4-5	X position of the top right corner
Bytes 6-7	Y position of the top right corner
Bytes 8-9	X position of the bottom left corner
Bytes 10-11	Y position of the bottom left corner
Bytes 12-13	X position of the bottom right corner
Bytes 14-15	Y position of the bottom right corner

9.12.8 Read Keystone Corners Queued (64h)

This command returns the keystone configuration parameters currently set.

9.12.8.1 Return Parameter(s)

Get Keystone Corners Queued
Data returned is in the same format as the Write Parameter(s).

9.13 XPR

XPR Commands Table	
Command	**Section**
Write XPR Enable Mode Queued	Section 9.13.1
Read XPR Enable Mode Queued	Section 9.13.2
Write XPR Calibration Mode	Section 9.13.3
Write XPR Actuator Position	Section 9.13.4
Read XPR Actuator Position	Section 9.13.5
Write XPR Actuator Dac Gain	Section 9.13.6
Read XPR Actuator Dac Gain	Section 9.13.7
Write XPR Actuator Subframe Delay	Section 9.13.8
Read XPR Actuator Subframe Delay	Section 9.13.9
Write XPR Actuator Dac Offset	Section 9.13.10
Read XPR Actuator Dac Offset	Section 9.13.11
Write XPR Actuator Fixed Output Level	Section 9.13.12
Read XPR Actuator Fixed Output Level	Section 9.13.13

9.13.1 Write XPR Enable Mode Queued (80h)

This command sets the XPR mode.

In XPR On the actuator drivers are always engaged. In XPR Off the actuator drivers are always disabled. In XPR Auto the system determines if the actuator drives should be driven.

An XPR waveform will not be seen when the controller is configured for a 1080p using frame rates exceeding 62Hz.

Note that a fixed output of 1.65V is expected in the case that XPR is enabled and an unstable FSync or invalid configuration is detected.

This command must be followed with a Write Execute Display command to be applied.

9.13.1.1 Write Parameter(s)

Set XPR Enable Mode Queued	
Write Parameter(s)	
Byte	**Description**
Byte 0	XPR enable mode 0 = XPR Auto 1 = XPR Always ON 2 = XPR Always Off

9.13.2 Read XPR Enable Mode Queued (80h)

This command returns the current XPR Enable Mode.

9.13.2.1 Return Parameter(s)

Get XPR Enable Mode Queued
Data returned is in the same format as the Write Parameter(s).

9.13.3 Write XPR Calibration Mode (81h)

This command enables an image processing bypass mode for use during XPR calibration.

This command causes the controller to bypass all image processing including XPR image processing and establishes a one-to-one correspondence between source image pixels and the mirrors on the DMD. This mode is useful for seeing clear splits between XPR subframes when performing XPR calibration. XPR calibration is needed to assure the mechanical actuator has optimal alignment when the system displays each spatially-shifted subframe image.

This command must be followed with a Write Execute Display command to be applied, along with speacial calibration Splash or external image set.

There is no readback command for this setting because there is no exit from this bypass mode. To exit bypass a system restart is required.

9.13.3.1 Write Parameter(s)

Set XPR Calibration Mode
Write Parameter(s)
This command has no write parameters.

9.13.4 Write XPR Actuator Position (82h)

This command sets the position of the XPR mechanical actuator for calibration purposes. There are 24 possible mechanical positions for 4-way actuator. Use this command while performing XPR calibration using a TI-provided XPR calibration splash image.

For this command to take effect, command Set XPR Calibration Mode must be enabled.

9.13.4.1 Write Parameter(s)

Set XPR Actuator Position	
Write Parameter(s)	
Byte	**Description**
Byte 0	Mechanical position number. Range 0 - 23 for 4-way actuator and 0 - 1 for 2-way actuator.

9.13.5 Read XPR Actuator Position (82h)

This command sets the position of the XPR mechanical actuator for calibration purposes.

9.13.5.1 Return Parameter(s)

Get XPR Actuator Position
Data returned is in the same format as the Write Parameter(s).

9.13.6 Write XPR Actuator Dac Gain (83h)

This command configures the DAC gain for actuator waveforms 0 and 1.

This command must be followed with a Write Execute Display command to be applied.

9.13.6.1 Write Parameter(s)

Set XPR Actuator Dac Gain	
Write Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2

Byte 1	DAC Gain. Range 0 - 255 format u1.7 (0 to 1.9921875)
	Range = 0.0000 to 1.9921875
	Format = u1.7

9.13.7 Read XPR Actuator Dac Gain (83h)

This command configures the DAC gain for actuator waveforms 0 and 1.

9.13.7.1 Read Parameter(s)

Get XPR Actuator Dac Gain	
Read Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2

9.13.7.2 Return Parameter(s)

Get XPR Actuator Dac Gain	
Return Parameter(s)	
Byte	**Description**

Byte 0	DAC Gain. Range 0 - 255 format u1.7 (0 to 1.9921875)
	Range = 0.0000 to 1.9921875
	Format = u1.7

9.13.8 Write XPR Actuator Subframe Delay (84h)

This command configures the subframe delay for actuator waveforms 0 and 1.

This command must be followed with a Write Execute Display command to be applied.

9.13.8.1 Write Parameter(s)

Set XPR Actuator Subframe Delay	
Write Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2
Bytes 1-4	Subframe delay. Range 0 - 4000 in micro-second (us) (us) Range = 0 to 4000

9.13.9 Read XPR Actuator Subframe Delay (84h)

This command configures the subframe delay for actuator waveforms 0 and 1.

9.13.9.1 Read Parameter(s)

Get XPR Actuator Subframe Delay	
Read Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2

9.13.9.2 Return Parameter(s)

Get XPR Actuator Subframe Delay	
Return Parameter(s)	
Byte	**Description**
Bytes 0-3	Subframe delay. Range 0 - 4000 in micro-second (us) (us) Range = 0 to 4000

9.13.10 Write XPR Actuator Dac Offset (85h)

This command configures the subframe delay for actuator waveforms 0 and 1.

This command must be followed with a Write Execute Display command to be applied.

9.13.10.1 Write Parameter(s)

Set XPR Actuator Dac Offset	
Write Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2
Byte 1	DAC Offset. Range 0 to 255 format s7.0 (-128 to 127) Range = -128 to 127

9.13.11 Read XPR Actuator Dac Offset (85h)

This command configures the subframe delay for actuator waveforms 0 and 1.

9.13.11.1 Read Parameter(s)

Get XPR Actuator Dac Offset	
Read Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2

9.13.11.2 Return Parameter(s)

Get XPR Actuator Dac Offset	
Return Parameter(s)	
Byte	**Description**
Byte 0	DAC Offset. Range 0 to 255 format s7.0 (-128 to 127) Range = -128 to 127

9.13.12 Write XPR Actuator Fixed Output Level (86h)

This command configures the fixed output level for actuator waveforms 0 and 1.

Note that a fixed output of 1.65V is expected in the case that XPR is enabled and an unstable FSync or invalid configuration is detected. If XPR is disabled, or a 1080p resolution is configured for boot up, the output is expected to be at 0V.

This command must be followed with a Write Execute Display command to be applied.

9.13.12.1 Write Parameter(s)

Set XPR Actuator Fixed Output Level	
Write Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2
Byte 1	Fixed Output Level. Range 0 to 255 format s7.0 (-128 to 127) Range = -128 to 127

9.13.13 Read XPR Actuator Fixed Output Level (86h)

This command configures the fixed output level for actuator waveforms 0 and 1.

9.13.13.1 Read Parameter(s)

Get XPR Actuator Fixed Output Level	
Read Parameter(s)	
Byte	**Description**
Byte 0	Actuator waveform control channel for which the command parameter has to be applied 0 = 1 1 = 2

9.13.13.2 Return Parameter(s)

Get XPR Actuator Fixed Output Level	
Return Parameter(s)	
Byte	**Description**
Byte 0	Fixed Output Level. Range 0 to 255 format s7.0 (-128 to 127) Range = -128 to 127

10 Notes On Use Cases

The following sections provide direction on specific uses cases of the controller.

10.1 Program Flash Image Procedure

To properly update the DLP® flash image, the following steps should be taken while the system is powered on:

1. Transition to Boot Application (see Section 8.1.3).
    a. **Note**
    Once the system enters boot mode, the flash image must be re-programmed as a sector of the flash image will always be corrupted once this transition takes place.
2. Build a checksum list or dictionary for all sectors to be programmed from the flash image data.
    a. See Section 7.1.10 for details on calculating checksums.
3. Unlock flash update operation (see Section 7.1.4).
4. (Optional) Back up flash image on DLP® system if desired by the following steps:
    a. Initalizing the Flash Read Write Settings (see Section 7.1.7) using 0 for the start address and 4096 for the number of bytes for the Flash Write command given that the entire image is being stored.
    b. Read and store the entire flash image using the Read Flash Write command (see Section 7.1.9) in a loop.
5. Erase the flash image entirely or by sectors by the following steps:
    a. Initializing the Flash Read Write Settings (see Section 7.1.7).
    b. Erasing the sectors (see Section 7.1.6) in a loop or the entire image at once (see Section 7.1.11).
6. Program flash image sectors data by the following steps:
    a. Initialize the Flash Read Write Settings (see Section 7.1.7).
    b. Writing new data to the flash using the Write Flash Write commands (see Section 7.1.8).
7. Lock flash update operation (see Section 7.1.4).
8. Verify that checksums match by looping through flash image sectors using the Read Checksum command (see Section 7.1.10) and compairing with the checksums recorded in step 2.
9. Transition to Main Application (see Section 8.1.3).

10.2 Source Setup Procedure

Note

The controller should not receive V-by-One data until the initialization process is complete. Issuing V-by-One data before the completion of initialization may jeopardize hardware.

For optimal performance, the following procedure is recommended for configuring sources:

1. Disable all LED's to hide transition artifacts with Write LED Enable (Section 9.6.1)
2. Set source configuration
   a. Write VBO Configuration (Section 9.4.9) for V-by-One
   b. Write Splash Screen Select (Section 9.5.1) for Splash
   c. Write TPG Pre Defined Pattern (Section 9.3.5) or specific TPG command for test patterns
3. Send Write Execute Display (Section 9.7.1) command
   a. Issuing a Write Execute Display command at this point may not be fully necessary, but can improve locking performance in the following step.
4. Check for source lock with Read Input Source Status (Section 9.4.1)
5. Set display and buffer commands which include:
   a. Write Display Size Queued (Section 9.7.7)
   b. Write Image Crop Queued (Section 9.7.5)
   c. Write XPR Mode Queued (Section 9.13.1)
6. Send Write Execute Display (Section 9.7.1) command
   a. It is recommended to check Read Execute Display Status (Section 9.7.2) for a successful execution before proceeding
7. Enable LED's with Write LED Enable (Section 9.6.1)

Note

When using streaming XPR resolutions, it may be necessary to adjust the image sharpness (Section 9.9.9) to correct shadowing or flickering artifacts. By default this value will be set to 0, which should be optimal for clearing these artifacts. If any blurriness is seen on XPR images, the Image Sharpness may need to be tuned for improved performance.

While the controller can support the timings noted in the controller's datasheet, the following standard timings have been explicitly validated:

- VESA 1280x720 @ 60Hz
- VESA 1920x1080 @ 60Hz
- EIA 3840x2160 @ 30Hz
- EIA 3840x2160 @ 50Hz
- EIA 3840x2160 @ 60Hz

10.2.1 Loss Of Video Source

In the case that video data is suddenly disconnected or halted, the controller will continue transmitting the last valid data that is has received. This may result in image artifacts. It is recommended to monitor the video source via a host processor controlling the video front-end or the VBO status (see Section 9.4.11) in the case of a V-by-One front end to ensure the state of the video source. If this video source is found to be lost it is recommended to switch the source to an image curtain, test pattern, or splash screen.

10.3 Variable Refresh Rate (VRR)

DLP®, being a color sequential system, requires relatively longer time compared to sample & hold display in order to “completely” display frame content since all 3 color components need to be displayed one after the other. This increases the mininimum resolution that can be supported for VRR since the step size for change will be the minimum time to display all three colors. To overcome this limitation, DLP® has leveraged bitplane dithering (that can display complete frame content for a color field for a short time) and rolling buffer operation instead. Running the system in Low Latency VRR Mode results in a higher refresh rate and smaller color integration period. This will equal or surpass the Nyquist sampling frequency. This approach ensures that there is no minimum step size limitation and the DLP® system can still display the complete image for every frame. Such behavior is due to the color orders being cyclic. In other words, the system can project RGB, GBR, or BRG for a frame and could change per frame with no impact to image content being displayed on the screen.

In non-VRR cases when running in Default Mode, the frame rate isn't changing and so the DLPC will lock the sequence to the frame period so long as this frequency is within the defined frequency bins.

10.4 3D

In order for three dimensional images to be perceived, the controller projects differing left and right frames sequentially. A complete 3D frame will have a left and a right subframe which will be shown or blocked via 3D glasses. As a result the controller will be transmitting frames as 120Hz for a 60Hz frame to be perceived. For proper use, an external trigger signal must drive into the GPIO that acts as the left/right frame sync trigger, known as the 3DLR trigger. In the case of the DLPC8445, this input is GPIO_21.

A proper setup for 3D video transmission would be the following:

1. Connect oscillating signal into the controller left/right frame sync trigger GPIO.
2. Boot DLP® circuitry.
3. Set up source and configure V-by-One.
    a. For 3D, acceptable sources would be 1080p 100Hz or 120Hz frame sequential alternating streams. The sequential alternating stream must be configured by the video front end.
    b. See Section 10.2 for information on V-by-One source setup.
4. Send Enable Three D command (Section 9.4.3) according to the front end and sync signal.

Note

When running a 3DLR trigger, 1 milli-second minimum seperation is needed between the trigger pulse and VSYNC for stable locking.

Note

Tests have shown that 3D is non-functional when blankings equal or exceed 144.

10.5 High Dynamic Range (HDR)

In order to properly set up HDR, the following steps should be performed:

1. Ensure that HDR is enabled for the Look being used.
    a. HDR is typically enabled by Looks by default.
2. Set the System Brightness Range Setting (Section 9.2.3). Typical values are between 0.1 and 400 nits.
3. Set HDR Source Configuration (Section 9.2.1)
4. Set HDR Strength (*Write HDR Strength Setting (72h)*)

11 Error Codes

The following table contains the error codes returned from commands. Any codes missing from this table are internal error codes. Please contact a DLP engineer if any non-disclosed error is reported.

Table 11-1. Error Code Table

Error Code Table	
Error Code	**Description**
2	RTP Invalid Peripheral
3	Flash Operation Locked
4	System in Busy State
5	Flash Communication failed
6	Flash R/W not initialized
7	Invalid Parameter
8	IRAM R/W not Initialized
9	IRAM Communication Failed
10	IQC Abort Exception
11	IQC Prefetch Exception
12	IQC Undefined Exception
13	BootRom CRC Error
14	Flash Out of Range
15	Flash Read Time-out
16	Flash SFC busy
17	Flash SFC Status Timeout
18	Flash Status Timeout
19	Invalid Address
20	Invalid Byte Count
21	Invalid Command Type
22	Invalid Command Format
23	Invalid Flash Clock Rate
24	Invalid Flash Write Sequence
26	Invalid OP Code
27	Invalid Signature
28	Command Mismatch
30	Pay Load Size Mismatch
31	No Command Available
32	No Valid Command Found
33	Software FIFO Full
34	TXFIFO Full
36	RTP GPIO Invalid Pin Access
37	RTP GPIO Invalid Dual use Pin Access
38	SSF Invalid Peripheral
39	SSF Invalid Boot Config Type
40	DDP Invalid Flag Id
41	Input Parameter Error Type is Invalid
42	RTP GPT Error
43	Flashtable ID Error
44	Flashtable Locked
617	I2C Slave Suspended
801	UART Port Invalid

Table 11-1. Error Code Table (continued)

Error Code Table	
802	UART Invalid Baud Rate
803	UART Invalid Parity
804	UART Invalid DataBits
805	UART Invalid StopBits
806	UART Invalid Flowcontrol
807	UART Invalid RX Trigger Level
808	UART Invalid TX Trigger Level
809	UART Port Disabled
810	UART Framing Error
811	UART Parity Error
812	UART Break Error
813	UART Overrun Error
814	UART Timeout
815	UART unexpected Error
817	UART Group Invalid Timeout
818	UART Invalid RX Data Polarity
819	UART Function not Valid in this Mode
820	UART Port in Use
821	UART Flowcontrol Not Valid
822	UART Invalid RX Data Source
1001	SSP Invalid Port
1002	SSP Port Disabled
1003	SSP Invalid Port Mode
1004	SSP Invalid Duplex
1005	SSP Invalid Transfer Mode
1006	SSP Invalid Chip Select
1007	SSP Invalid Data Bits
1008	SSP Invalid Clock Rate
1009	Invalid SPI Clock Polarity
1010	Invalid SPI Clock Phase
1011	SSP Timeout
1012	SSP Receive Overrun Error
1013	SSP Group Invalid Timeout
1014	SSP Group Event Fail
1015	SSP Unexpected Error
1016	SSP Invalid Auto Transfer Size
1017	SSP DS Not Populated
1018	SSP Invalid Protocol
1019	SSP Invalid Number Of Bytes
1201	SPI Not Enabled
1202	SPI Invalid Secondary Port
1203	SPI Invalid Port Chip Select
1204	SPI Invalid Listen Command
1401	USB Invalid Endpoint number
1402	USB Invalid Packet size
1403	USB Invalid Endpoint type
1404	USB One or more Parameters are Invalid

Table 11-1. Error Code Table (continued)

Error Code Table	
1405	USB Device timeout
1406	No free HW endpoins to allocate
1407	USB Device not ready
1901	Valid flash table is not found
1902	Flash initialization failed
1903	The provided Index BIGGER_THAN the number of block instances present in flash
1904	The provided BlockType is invalid
1905	Mismatch between revisions of app binary and flash table
2201	No DLPA device configured
2202	DLPA SSP chip select conflict
2203	Cannot write to read only DLPA register
2204	DLPA invalid register Access
2205	DLPA register out of Range Value
2206	DLPA device index out of range
2207	DLPA illumination drive level out of range
2208	DLPA illumination drive channel out of range
2209	DLPA invalid device ID
2210	DLPA incorrect device type
2211	Number of DLPA device out of range
2212	DLPA functionality not supported
2214	DLPA fault condition detected
2215	DLPA illumination fault detected
2619	SRC VBO Config Block does not exist in Flash
2620	Invalid Number of VBO Lanes selected. Valid Lane Mode Configurations are 1 2 4 and 8
2621	Invalid VBO Byte Mode selected. Valid Byte Mode Configurations are 3 Byte 4 Byte and 5 Byte
2622	Invalid VBO Data Map selected
2623	Invalid VBO FRange selected
2624	Invalid Lane Mapping. VBO Lane Configuration should follow 1-1 Mapping
3402	Invalid Display Source
3403	Display Width Bigger than Valid Max Width
3404	Display First Pixel Plus Width Bigger than Valid Maximum Width
3405	Display First Line Plus Height Bigger than Valid Maximum Height
3406	Display First Line Plus Hieght Bigger Than Valid Maximum Height
3407	Display Width Smaller than Valid Minimum Width
3408	Display Height Smaller than Valid Minimum Height
3409	Crop Width Bigger than Image Width
3410	Crop First Pixel Plus Crop Width Bigger than Image Width
3411	Crop Height Bigger than Image Height
3412	Crop First Line Plus Crop Line Bigger than Image Height
3413	Crop Width Is Smaller than Valid Min Crop Width
3414	Crop Height Is Smaller than Valid Min Crop Height
3415	Input Image Width Bigger than Valid Image Width
3416	Input Image Height Bigger than Valid Image Height
3417	Input Image Width Smaller than Valid Image Width
3418	Input Image Height Smaller than Valid Image Height
3430	Source Display Resolution Mismatch
3431	Invalid Display Color

Table 11-1. Error Code Table (continued)

Error Code Table	
3432	Warp Calculation Error
3433	Unsupported XPR Mode for Display
3434	Invalid or no Frame Syncs
3435	Unsupported Frame Timings for Scaling Operation
3501	Input Horizontal resolution is less than Minumum allowed
3502	Input Horizontal resolution is greater than Maximum allowed
3503	Input Vertical resolution is less than Minumum allowed
3504	Input Vertical resolution is greater than Minumum allowed
3505	For Warping Only Keystone feature is supported
3506	Image rotation is not supported when warp is enabled
3507	Warp is Disabled
3508	For provided keystone corners Corner coordinates cannot be negative or more than display resolution
3509	For provided keystone corners Two corners coordinates are same which is not supported
3510	Unable to Read Angles from Corners Only corners can be read from Angles and Optical parameter Input
3511	Unable to Read Optical parameters from Corners Only corners can be read from Angles and Optical parameter Input
3512	Provided Keystone Angles are out of Solution Space
3513	Provided Throw Ratio Input is not supported
3514	Provided Vertical Offset Input is not supported
3515	Provided Keystone Angles Input are out of scope of DMD
3516	Provided Keystone Corners Input creates twisted image or out of scope of DMD
3517	Provided Top-Left Corner Input is not supported
3518	Provided Top-Right Corner Input is not supported
3519	Provided Bottom-Left Corner Input is not supported
3520	Provided Bottom-Right Corner Input is not supported
3521	For provided keystone corners Corner coordinates cannot be negative or more than display resolution
3522	Provided Input or Display Resolution for keystone operation is not Supported
3523	Warp Algorithm Calculation Error
3524	Warp Algorithm Calculation Error
3525	Warp Algorithm Calculation Error
3818	DMD is not parked before enabling DMD power
3819	Sequence is enabled before enabling DMD power
3820	Failed enabling DMD low speed interface
3821	Communication error with DMD Low speed interface
3822	DMD fuse read error
3823	DMD is not valid for the configuration detected in the system
3824	Failed enabling DMD voltages
3825	Failed sending global VBIAS command to DMD
3826	Failed powering up DMD high speed interface
3827	Communication error with DMD high speed interface
3828	Failed unparking the DMD mirrors
3829	Failed parking the DMD mirrors
3830	Failed powering down DMD high speed interface
3831	Failed disabling DMD low speed interface
3832	Failed disabling DMD voltages
3833	True global not supported in DMD
3834	DMD is powered up before changing true global mode

Table 11-1. Error Code Table (continued)

Error Code Table	
3835	DMD Clock setup failure
3836	Controller is invalid or has not been validated before powering up DMD
3837	Failed in Mirror Protection Cicuit initialization
3838	DMD Device ID mismatch
4001	Selected Look index BIGGER_THAN number of looks in firmware
4002	Selected Source type not present in current look
4003	Selected Illuminator not present in current Look/Source
4004	Desired color time not valid in the current Look/Source/FrameRateBin
4018	Sequence start is not internally triggered
4019	Input Vsync not providing well defined Vsync
4020	Input frame rate is not supported in any of the present bins
4021	Selected Source type does not support discrete DC
4022	Dark Time % specified in flash is less than the TI specified min value
4023	Set System Brightness bins is not found
4024	Set Duty Cycle is out of Range
4901	The source seleced is not being supported among TPG Splash External Source
4902	Configuration Block in Composer Project not Found
4903	Invalid Image Processing YCbCr argument
4904	Image Provessing input argument out of range
4905	Hue Satuation Gain input argument out of range
4906	CCA Values resulated in Unsuccessful Measured Luminance not equal to 1 or calculations error
4907	HDR Configuration Not valid or out of range
4908	HDR is not enabled in the composer project
4909	System Brightness is not set via command
4910	Issue with the configuraiton. HDR is disabled
5201	Value of pattern number is not valid
5202	Specified border width is too large
5203	Ramp configuration is not valid
5204	Function does not match current test pattern
5205	Horizontal Resolution is not valid
5206	Vertical Resolution is not valid
5207	Horizontal Porch is not valid
5208	Vertical Porch is not valid
5209	Value of Frame Rate is not valid
5210	Color Value improper
5211	AOM Data not received successfully
5212	Pixel Clock crossed greater than maximum limit
5213	Parameters for the current test pattern are invalid
5301	The splash width is out of VALID range
5302	The splash height is out of VALID range
5304	Invalid Pixel format. Only RGB 888 RGB 565 and YUV 422 pixel formats are VALID
5305	The splash image could not be transferred properly from Flash to Mailbox using FDMA
5306	Unable to get the Buffer Mode from AOM
5307	Source other than Splash is setup
5308	Trying to access an invalid Splash Screen Index
5311	Horizontal Front Porch for Splash Image is out of VALID range
5312	The Splash Index has not been set

Table 11-1. Error Code Table (continued)

Error Code Table	
5313	The frame rate set for internal FGen is out of VALID range
5401	Block does not exist in the system
5402	SSI Channel value exceeds 6
5403	SSI minimum drive level for the color exceeds the maximum drive level value
5404	PMIC type is invalid(>=3)
5405	LED connection error
8000	Product configuration error
12000	WPC configuraton is invalid or not available
12001	WPC calibration data for LED or Sensor or both is/are invalid or not available
12002	WPC function is enabled to perform this operation first WPC function needs to be disabled
12003	WPC function is disabled to perform this operation first WPC function needs to be enabled
12004	WPC sensor initialization failed. It could be because of missing sensor or invalid sensor configuration
12005	WPC sensor sync signal initialization failed
12006	WPC sensor output couldn't be read
12008	WPC calculated DC is not ready. WPC function needs to be enabled to read computed duty cycle
12009	WPC calibration matrix computation failed
12010	WPC system color point is not ready. WPC function needs to be enabled to read system color point
12011	WPC calculated DC is not ready. WPC function needs to be enabled to read computed duty cycle
12012	WPC timer configuration failed
12013	WPC target white point is invalid or not set
12014	WPC feature is disabled in the system
12015	WPC target color point gain should not be greater then 0
12016	Invalid LED type specified
12201	DB initialization is failed
12202	DB configuraton is invalid or not available
12203	DB feature is disabled in the system
12204	DB configuraton is invalid or not available
12205	DB feature is disabled in the system
12206	DB function is disabled to perform this operation first DB function needs to be enabled
12207	DB output is not processed completely
12208	DB configuraton is invalid
12401	Illumination configuration is invalid or not available
12402	Invalid duty cycle or not within the range supported
12403	Invalid duty cycle or not within the range supported
12404	Automatic current adjustment is enabled manual update is not allowed
12405	Duty cycle is controlled by WPC function manual duty cycle adjustment is not allowed
12406	System is running at discrete DC mode duty cycle can be selected only through supported discrete DC sets
12407	System is running at DC range mode discrete DC index selection is not allowed
12408	Invalid current levels set
12409	Invalid duty cycle set
12410	Current adjustment is not processed yet
12411	Current adjustment failed
12412	Duty cycle adjustment couldn't be processed
12601	Unsupported sensor type
12602	Sensor initialization failed
12603	Incorrect Chip ID read from sensor
12604	Unable to communicate with sensor

Table 11-1. Error Code Table (continued)

Error Code Table	
12605	Provided I2C Port for sensor is not valid
12606	Provided sensor configuration is not valid
12607	Data from sensor is not ready yet
12701	Thermistor is not availible
12720	EEPROM I2C Port not valid
12721	EEPROM driver failure
12722	EEPROM address not valid
12723	EEPROM I2C semaphore acquire failed
12724	EEPROM I2C API call error
12725	EEPROM invalid device type
12726	EEPROM Scratchpad read size should be between 1-2048 bytes
13000	EEPROM initialization failed
13001	EEPROM communication failed
13002	Secondary EEPROM initialization timeout
13003	Provided offset is not valid
13004	Runtime Data writes are disabled
13005	EEPROM write operation failed
13006	EEPROM read operation failed
13007	Request not valid in Use Default Mode
13008	Requested Calibration data block not present in EEPROM
13009	Data Manager cannot Update Commit Mode with Data Storage Mode set to Flash
13010	Communication with secondary EEPROM failed
13011	Flash Sector read failed
13012	Flash Sector erase failed
13013	Flash Sector write failed
13014	Flash setting overwrite limit exceeded
32767	Invalid/out of range error type

12 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from August 24, 2024 to November 30, 2024 (from Revision * (August 2024) to Revision A (November 2024)) **Page**