MCM301 APT Command Reference

Issue 1

Thorlabs Imaging System

October 23, 2023

Contents

1 Introduction

1.1 Purpose and Scope

This document describes the low-level communications protocol and commands used between the host device and the MCM301. Note that this communication protocol follows the standards laid out by the Thorlabs Motion Control systems communication protocol (APT), allowing for interoperability. The information contained in this document is intended to help third party system developers to write their own application to interface to the MCM301 without the constraints of using a particular operating system or hardware platform. As a result, all commands listed represent an action that the end-user could perform on the device if the software developer intends to expose it.

1.2 Electrical Interface

The the MCM301 uses one USB interface to communicate with the host device. Be aware of the USB enumeration schemes used in the system.

1.2.1 USB Interface

The electrical interface within the the MCM301 uses one USB interface. The interface uses a USB driver in the embedded microcontroller. Note that the enumeration scheme is dependent on the transport layer (serial, Ethernet, etc.), but the commands themselves are not.

The embedded USB driver is a USB 2.0 device. Moreover, the serial connection uses a default configuration of 8N1 at 512000 baud.

1.3 USB Device Enumeration

Thorlabs Imaging Systems devices can be identified over USB by the USB Product ID. All Thorlabs Imaging Systems devices have the USB Vendor ID 0x1313.

Table 1: Thorlabs Imaging Systems PIDs

Product ID	Product
0x2F03	Thorlabs MCM Bootloader
0x2016	Thorlabs MCM301

1.4 Overview of the Communications Protocol

The communications protocol used in Thorlabs Imaging Systems devices always starts with a fixed length, 6-byte *message header*. In some cases, the message header is followed by a variable-length *data packet*. For simple commands, the 6-byte header is sufficient to convey the entire command. For more complex commands (such as parameter passing), the header needs the data packet to convey the additional data.

The header part of the message always contains information that indicates whether or not a data packet follows the header. If a data packet follows the header, the header will also contain the number of bytes in the following data packet. This allows for the ends of the communication system to segment the start and end of packets.

Note that in the sections following, the C-type notation will be used for hexadecimal values (e.g. 0x55 means 55 hexadecimal). In addition, logical operators will use C-type notation as well (| for bit-wise OR, & for bit-wise AND, etc.). Values that are longer than a byte follow the Intel little-endian format *unless* specified otherwise.

1.5 Description of the Message Header

The diagram below depicts the message header.

Table 2: APT Header Diagram

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
No Data Packet	Message ID		Param 1	Param 2	Destination	Source
Data Packet	Message ID		Data Packet Length		Destination	Source

Table 3: APT Header Details

Field	Description
Message ID	Enumeration for the action the command represents.
Param 1	The first parameter for the command (if applicable). 0 when unused.
Param 2	The second parameter for the command (if applicable). 0 when unused.
Data Packet Length	The length of the proceeding data packet in bytes.
Destination	The destination module for the message.
Source	The source module for the message.

Bytes 2 and 3 change meaning if a data packet follows the header. If no data packet follows, the two bytes represent two, 8-bit parameters for the command.

Table 4: APT Addresses

Name	ID	Description
HOST_ID	0x01	Host device (i.e. control PC)
MOTHERBOARD_ID	0x11	Rack controller, motherboard, or comms router board
SLOT_1	0x21	Slot/Bay 0
SLOT_2	0x22	Slot/Bay 1
SLOT_3	0x23	Slot/Bay 2
SLOT_4	0x24	Slot/Bay 3
SLOT_5	0x25	Slot/Bay 4
SLOT_6	0x26	Slot/Bay 5
SLOT_7	0x27	Slot/Bay 6
SLOT_8	0x28	Slot/Bay 7
SLOT_9	0x29	Slot/Bay 8
SLOT_10	0x2A	Slot/Bay 9
MOTHERBOARD_ID_ STANDALONE	0x50	Generic USB Unit

If a data packet follows, the two bytes represent one, 16-bit field describing the length of the data packet. To determine if a data packet follows a header, check the most-significant bit (MSB) of byte 4; when set, a data packet will follow.

Host devices will always use the address 0x01. However, the destination address depends on the type of controller. "Generic USB units," any device that acts as one system, have the address 0x50. Slot, bay, and routed units–systems that provide an interface for other hardware systems–have the address 0x11. When a system uses address 0x11, then they can also receive messages with different destinations; the device will route the message to the appropriate system. Currently, addresses 0x21 through 0x2A can be used with slot/bay units to communicate to slot 0 through slot 9, respectively.

1.6 General Message Exchange Rules

The type of messages used in the communications exchange between the host and sub-modules can be divided into 4 general categories:

Commands are typically bundled in the SET → REQUEST → GET format. The SET command is a one-way exchange to configure a parameter for the controller. The REQUEST (REQ) command begins a two-way exchange to read a parameter from the controller. The GET command finishes the exchange by sending the requested data to the host.

Two-way commands that do not request data typically follow the COMMAND → RESPONSE format. In this format, an action is attempted by the COM-

Table 5: APT Exchange Types

One-Way	Host issues a command. The sub-module carries out the command without acknowledgement.
Two-Way	Host issues a command. The sub-modules carries out the command and responds with data.
Periodic	The sub-module sends data to the host periodically.
Reverse	The sub-module sends a message to the host aperiodically.

MAND, and the RESPONSE indicates how the commands was handled. This format works best when the data replied depends on the proceeding command, not the internal state of the controller.

1.7 Format Specifiers

See Table 6 for the format specifiers.

Streams allow for more complex command sequences without bloating the limited APT message ID set. They commonly represent bulk-transfer of data could exceed the APT command size.

Table 6: APT Format Specifiers

Format	Encoding
byte	8-bit integer.
char	8-bit ASCII character.
word	16-bit unsigned integer in the little-endian format.
short	16-bit signed integer in the little-endian format.
dword	32-bit unsigned integer in the little-endian format.
long	32-bit signed integer in the little-endian format.
byte[N]	A contiguous array of N bytes.
char[N]	A contiguous ASCII-character array (string) taking up N bytes.
float	IEEE-754 single-precision floating point (32 bits) in the big-endian format.
double	IEEE-754 double-precision floating point (64 bits) in the big-endian format.
stream	Multi-command, byte-aligned binary stream.

2 Commands

The rest of this page has been left intentionally blank.

MGMSG_MOT_SET_CHANENABLESTATE 0x0210

Function: Sent to set enable or disable a stepper.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x10	
Message ID (upper)	1	0x02	
Slot Card	2	byte	
Enabled	3	byte	• 0 Disable the stepper. • 1 Enable the stepper.
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

MGMSG_MOT_REQ_CHANENABLESTATE 0x0211
MGMSG_MOT_GET_CHANENABLESTATE 0x0212

Function: Sent to query the enabled status of stepper.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x11	
Message ID (upper)	1	0x02	
Slot Card	3	byte	
Parameter 2	2	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x12	
Message ID (upper)	1	0x02	
Slot Card	2	byte	
Enabled	3	byte	• 0 Stepper is disabled. • 1 Stepper is enabled.
Destination	4	0x01	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$

MGMSG_MOD_IDENTIFY 0x0223

Function: Sent to initiate the LED identification sequence on the controller (power LED) as well as any HID LEDs for controls mapped to the given slot.

COMMAND:

Name	Offset	Format	Notes
Message ID (lower)	0	0x23	
Message ID (upper)	1	0x02	
Slot	3	byte	The ID of the slot whose mapped LEDs should identify. Set to "0xFF" to apply to identify the controller itself.
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

MGMSG_MOT_SET_ENCCOUNTER 0x0409

Function: Sets the encoder count of the stepper to the provided value, updates the encoder (if one is equipped), then updates the position to align with the encoder count.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.
This command should be avoided, with homing being used instead.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x09	
Message ID (upper)	1	0x04	
Length (lower)	2	0x06	
Length (upper)	3	0x00	
Destination	4	0x21 + $\|SLOT\|$ 0x80	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.
Slot Card	6	word	
Encoder Count	8	long	

MGMSG_MOT_SET_JOGPARAMS 0x0416

Function: Sets the jogging parameters for the slot card.

NOTICE: The response of MGMSG_MCM_GET_JOGPARAMS must be used as a template for MGMSG_MCM_SET_JOGPARAMS. Modify the response of the command with the fields provided.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x16	
Message ID (upper)	1	0x04	
Length (lower)	2	0x16	
Length (upper)	3	0x00	
Destination	4	0x21 + $SLOT$| 0x80	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.
Slot Card	6	word	
Reserved	8	byte[2]	For internal use only. Use previous values.
Jog Step Size	10	dword	The size of a jog step in encoder counts.
Reserved	14	byte[12]	For internal use only. Use previous values.

MGMSG_MOT_REQ_JOGPARAMS 0x0417
MGMSG_MOT_GET_JOGPARAMS 0x0418

Function: Sent to query the jogging parameters for the slot card.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x17	
Message ID (upper)	1	0x04	
Slot Card	3	byte	
Parameter 2	2	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x18	
Message ID (upper)	1	0x04	
Length (lower)	2	0x16	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$
Slot Card	6	word	
Reserved	8	byte[2]	For internal use only.
Jog Step Size	10	dword	The size of a jog step in encoder counts.
Reserved	14	byte[12]	For internal use only.

MGMSG_MOT_MOVE_HOME 0x0443

Function: Sent to begin a homing movement sequence.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.
Homing is disabled when the stepper has any soft limits.
Any movement command will interrupt the homing routine and mark the stepper as not homed.
The encoder count becomes unreliable until homing ends.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x43	
Message ID (upper)	1	0x04	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

The controller does not send a MGMSG_MOT_MOVE_HOMED command.

MGMSG_MOT_MOVE_ABSOLUTE 0x0453

Function: Sent to move the stepper to a specified encoder position.

NOTICE: The slot card must be able to drive the connected device for this command to be processed. The 6 byte "short" version of this command is *not* supported.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x53	
Message ID (upper)	1	0x04	
Length (lower)	2	0x06	
Length (upper)	3	0x00	
Destination	4	0x21 + $SLOT\|$ 0x80	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.
Slot Card	6	word	
Target Encoder Position	8	long	

The controller does NOT send a MGMSG_MOT_MOVE_COMPLETED after this command is finished.

MGMSG_MOT_MOVE_STOP 0x0465

Function: Sent to stop any motion on this stepper.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x65	
Message ID (upper)	1	0x04	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

This controller does not send a MGMSG_MOT_MOVE_STOPPED command.

MGMSG_MOT_MOVE_JOG 0x046A

Function: Sent to start a jog movement in the specified direction.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x6A	
Message ID (upper)	1	0x04	
Slot Card	2	byte	
Direction	3	byte	• 0 Negative Direction • 1 Positive Direction
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

The controller does not send a MGMSG_MOT_MOVE_COMPLETED command.

MGMSG_MOT_REQ_STATUSUPDATE 0x0480
MGMSG_MOT_GET_STATUSUPDATE 0x0481

Function: Sent to query general information (i.e. status) of the stepper driver.

NOTICE: The slot card must be able to drive the connected device for this command to be processed.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x80	
Message ID (upper)	1	0x04	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x81	
Message ID (upper)	1	0x04	
Length (lower)	2	0x14	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$
Slot Card	6	word	
Position	8	long	The stepper's step counts from its internal encoder.

Name	Offset	Format	Notes
Encoder Count	12	long	For stages with an encoder, this represents the value from the encoder. For stages without an encoder, this equals to the position divided by this stepper's counts per unit.
Status Bits	16	dword	

Bit Field for Status Bits

Mask	Bits	Name	Description
0x00000001	0	On Positive Direction Hardware Limit Switch	
0x00000002	1	On Negative Direction Hardware Limit Switch	
0x00000004	2	On Positive Direction Software Limit Switch	
0x00000008	3	On Negative Direction Software Limit Switch	
0x00000010	4	Moving in Positive Direction	Stage is in motion.
0x00000020	5	Moving in Negative Direction	Stage is in motion.
0x00000040	6	Jogging in Positive Direction	Stage is in motion.
0x00000080	7	Jogging in Negative Direction	Stage is in motion.
0x00000100	8	Motor Connected	The motor has been recognized by the controller.
0x00000200	9	Homing	Stage is in motion.
0x00000400	10	Homed	

Mask	Bits	Name	Description
0x00000800	11	Reserved	For internal use only.
0x00001000	12	Encoder Warning	Only applicable to steppers with BISS encoders. Reset when the encoder scale (and/or reading window) should be cleaned.
0x00002000	13	Encoder Error	Only applicable to steppers with BISS encoders. Reset when the internal safety checking algorithm fails or when the temperature exceeds the product's maximum.
0x00004000	14	Encoder High	Only appliable to steppers with magnetic encoders. Set when the magnet is too far or is out of alignment.
0x00008000	15	Encoder Low	Only appliable to steppers with magnetic encoders. Set when the magnet is too close.
0x00010000	16	Encoder Ready	Only applicable to steppers with mangetic encoders. Set while the encoder is ready to be used.
0x7FFE0000	17 - 30	Reserved	For internal use only.
0x80000000	31	Channel Enabled	• 0 Stepper motor is disabled. • 1 Stepper motor is enabled.

MGMSG_MOT_SET_EEPROMPARAMS 0x04B9

Function: Sent to save the parameters set by the passed command into EEPROM.

The following commands are supported:

Table 7: Supported Save Commands

Destination	Command	Parameter Usage	Additional Information
Motherboard	MGMSG_MOD_SET_JOYSTICK_MAP_IN	port number (16-bit int)	Saves the HID IN mappings of the selected port
Motherboard	MGMSG_MOD_SET_JOYSTICK_MAP_OUT	port number (16-bit int)	Saves the HID OUT mappings of the selected port
Stepper Card	MGMSG_MCM_SET_SOFT_LIMITS	unused	Saves the soft limits of the device
Stepper Card	MGMSG_MCM_SET_HOMEPARAMS	unused	Saves the homing parameters of the device
Stepper Card	MGMSG_MCM_SET_JOGPARAMS	unused	Saves the jog parameters of the device

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xB9	
Message ID (upper)	1	0x04	
Length (lower)	2	0x04	
Length (upper)	3	0x00	
Destination	4	byte| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Parameters	6	byte[2]	Command-dependent parameters identifying what to save.
Command to Save	8	word	APT command to save into EEPROM.

MGMSG_MCM_HW_REQ_INFO 0x4000
MGMSG_MCM_HW_GET_INFO 0x4001

Function: Sent to query hardware information from the MCM6000.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x00	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

Example: Query the hardware info on controller #1.
TX 05 00 00 00 11 01

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x01	
Message ID (upper)	1	0x40	
Length (lower)	2	0x5A	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Reserved	6	byte[4]	For internal use only.
Model Number	10	byte[8]	"MCM301"
Type	18	word	0
Firmware Version	20	byte[3]	Stored in interim, minor, major order.
CPLD Version	23	byte[2]	Stored in major, minor order.
Serial Number	25	char[17]	USB serial number of the device. Null-terminated.

Name	Offset	Format	Notes
APT Extended Data Limit	42	word	The maximum length of APT extended data sent to the controller.
Reserved	44	byte[24]	Unused.
Slot 0 Card Type	68	word	
Slot 1 Card Type	70	word	
Slot 2 Card Type	72	word	
Slot 3 Card Type	74	word	
Slot 4 Card Type	76	word	
Slot 5 Card Type	78	word	
Slot 6 Card Type	80	word	
Slot 7 Card Type	82	word	
Board Type	84	word	
Reversed	86	byte[2]	For internal use only.
Board Slot Count	88	word	

MGMSG_REQ_DEVICE 0x4006
MGMSG_GET_DEVICE 0x4007

Function: Sent to query identifying information on a device.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x06	
Message ID (upper)	1	0x40	
Slot Number	3	byte	
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x07	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0D	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Device ID	6	word	
Serial Number	8	uint64_t	
Default Slot Type	16	word	
Part Number	18	char[16]	Null-terminated part number for the connected device.
Device is Connected	34	byte	• 0 A device is not connected. The preceeding data is meaningless. • 1 A device is connected.

MGMSG_SET_DEVICE_BOARD 0x4008

Function: Sent to set the serial number of the device allowed to run on the passed slot.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x08	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0A	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Slot Number	6	word	
Serial Number	8	uint64_t	The 48-bit serial number from a plug-and-play device. If 0xFFFF000000000000 is sent, then the serial number check will be ignored.

MGMSG_REQ_DEVICE_BOARD 0x4009
MGMSG_GET_DEVICE_BOARD 0x400A

Function: Sent to query the serial number of the device allowed to run on the passed slot.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x09	
Message ID (upper)	1	0x40	
Slot Number	3	byte	
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x0A	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0A	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Slot Number	6	word	
Serial Number	8	uint64_t	The 48-bit serial number from a plug-and-play device. If the serial number is 0xFFFF000000000000, then serial number checking is not currently in use.

MGMSG_RESTART_PROCESSOR 0x400B

Function: Sent to manually restart the board.

COMMAND:

Name	Offset	Format	Notes
Message ID (lower)	0	0x0B	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

MGMSG_BOARD_REQ_STATUSUPDATE 0x4010
MGMSG_BOARD_GET_STATUSUPDATE 0x4011

Function: Sent to query the temperature sensors, high-voltage input, and slot card error bits.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x10	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x11	
Message ID (upper)	1	0x40	
Length (lower)	2	0x07	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Board Temperature Sensor, T	6	word	Refer to equation 1 for conversion.
High-Voltage Monitor, V_{in}	8	word	Refer to equation 2 for conversion.
CPU Temperature Sensor, T_C	10	word	Refer to equation 3 for conversion.
Error Code	12	byte[1]	See table below.

Temperature Sensor, T ($\circ$ C):

$$\begin{aligned} T_0 &:= 298 \\ V_{cc} &\approx 3.3 \\ B &:= 3930 \\ C &:= 273 \\ I &\in \mathbb{N} \\ I &\in [0, 4095] \\ V_T &= V_{cc}\frac{I}{4095} \\ T &= \frac{T_0 B}{T_0 \log(\frac{V_{cc}}{V_T} - 1) + B} - C \end{aligned} \tag{1}$$

Voltage Sensor, V_{in} (V):

$$\begin{aligned} \text{Board Type} &\in \mathbb{N} \\ \text{Rev 4} &:= 32774 \\ G_1 &:= 0.083170 \\ G_2 &:= 0.0661 \\ V_{cc} &\approx 3.3 \\ I &\in \mathbb{N} \\ I &\in [0, 4095] \\ V_V &= V_{cc}\frac{I}{4095} \\ V_{in} &= \begin{cases} \frac{V_V}{G_1} & \text{Board Type} < \text{Rev 4} \\ \frac{V_V}{G_2} & \text{Board Type} \geq \text{Rev 4} \end{cases} \end{aligned} \tag{2}$$

The board type can be acquired from MGMSG_MCM_HW_GET_INFO.

CPU Temperature Sensor, T_C ($\circ$ C):

$$\begin{aligned} m &:= \frac{1}{0.00233} \\ V_0 &:= 0.72 \\ T_0 &:= 27 \\ V_{cc} &\approx 3.3 \\ I &\in \mathbb{N} \\ I &\in [0, 4095] \\ V_T &= V_{cc}\frac{I}{4095} \\ T_C &= m(V_T - V_0) + T_0 \end{aligned} \tag{3}$$

Bit Field for Error Code

Mask	Bits	Name	Description
0x01	0	Slot 0 Error	Set when slot card 0 has an error.
0x02	1	Slot 1 Error	Set when slot card 1 has an error.
0x04	2	Slot 2 Error	Set when slot card 2 has an error.
0x08	3	Slot 3 Error	Set when slot card 3 has an error.
0x10	4	Slot 4 Error	Set when slot card 4 has an error.
0x20	5	Slot 5 Error	Set when slot card 5 has an error.
0x40	6	Slot 6 Error	Set when slot card 6 has an error.
0x80	7	Slot 7 Error	Set when slot card 7 has an error.

MGMSG_MOD_REQ_JOYSTICK_INFO 0x4012
MGMSG_MOD_GET_JOYSTICK_INFO 0x4013

Function: Sent to query information about a USB device connected to the controller. The return structure for MGMSG_MOD_GET_JOYSTICK_INFO varies between USB hubs and USB HID devices. To differentiate between the two, use the "Is a Hub" bit in the "Type Flags" field of the command. Use the MGMSG_MOD_REQ_JOYSTICK_CONTROL to get information about the joystick's controls.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x12	
Message ID (upper)	1	0x40	
Port Number	3	byte	Index (zero-based) of the port to query.
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

(device is a hub)

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x13	
Message ID (upper)	1	0x40	
Length (lower)	2	0x07	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Port Number	6	byte	
Vendor ID	7	word	
Product ID	9	word	
Type Flags	11	byte[1]	See table below. The "Is a Hub" bit will be set.

Name	Offset	Format	Notes
Port Count	12	byte	

(device is a joystick)

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x13	
Message ID (upper)	1	0x40	
Length (lower)	2	0x08	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Port Number	6	byte	
Vendor ID	7	word	
Product ID	9	word	
Type Flags	11	byte[1]	See table below. The "Is a Hub" bit will be reset.
Input Control Count	12	byte	
Output Control Count	13	byte	

Bit Field for Type Flags

Mask	Bits	Name	Description
0x01	0	Is a Hub	
0x02	1	Is Low Speed	
0xFC	2 - 7	Reserved	For internal use only.

MGMSG_MOD_SET_JOYSTICK_MAP_IN 0x4014

Function: Sent to configure the control of a HID input control, save it to storage, and enable the control.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x14	
Message ID (upper)	1	0x40	
Length (lower)	2	0x15	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Port Number	6	byte	
Control Number	7	byte	Control index for devices with multiple interfaces (buttons, knobs, etc.)
Vendor ID	8	word	
Product ID	10	word	
Target Port Number	12	byte	For controls that affect other HID controls, this is that target's port.
Target Control Number	13	word	For controls that affect other HID controls, this is that target's control number.
Destination Slots	15	byte	Bitfield indicating which slot will get USB messages from this control. For example, 0x05 (0b00000101) indicates that slot 1 and slot 3 will receive this HID control's messages.
Reversed	16	byte[3]	For internal use only. Set to 0.
Speed Modifier	19	byte	Scaling factor applicable to axis-type controls to reduce the rate of change. Strength = (IN Report Strength) × $(\frac{1+\text{Speed Modifier}}{256})$

Name	Offset	Format	Notes
Reverse Direction	20	byte	Inverts the control data when set.
Dead Band	21	byte	Control values within the deadband of 0 will have their control values set to 0.
Control Mode	22	dword	The type of MCM control that this HID control represents. See Table 8 for the control modes.
Control Disabled	26	byte	• 0 Control is enabled. • 1 Control is disabled.

MGMSG_MOD_REQ_JOYSTICK_MAP_IN 0x4015
MGMSG_MOD_GET_JOYSTICK_MAP_IN 0x4016

Function: Sent to query the configuration for a given HID control.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x15	
Message ID (upper)	1	0x40	
Port Number	2	byte	
Control Number	3	byte	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x16	
Message ID (upper)	1	0x40	
Length (lower)	2	0x15	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Port Number	6	byte	
Control Number	7	byte	Control index for devices with multiple interfaces (buttons, knobs, etc.)
Vendor ID	8	word	
Product ID	10	word	
Target Port Number	12	byte	For controls that affect other HID controls, this is that target's port.
Target Control Number	13	word	For controls that affect other HID controls, this is that target's control number.

Name	Offset	Format	Notes
Destination Slots	15	byte	Bitfield indicating which slot will get USB messages from this control. For example, 0x05 (0b00000101) indicates that slot 1 and slot 3 will receive this HID control's messages.
Reserved	16	byte[3]	For internal use only.
Speed Modifier	19	byte	Scaling factor applicable to axis-type controls to reduce the rate of change. Strength = (IN Report Strength) × $(\frac{1+\text{Speed Modifier}}{256})$
Reverse Direction	20	byte	Inverts the control data when set.
Dead Band	21	byte	Control values within the deadband of 0 will have their control values set to 0.
Control Mode	22	dword	The type of MCM control that this HID control represents. See Table 8 for the control modes.
Control Disabled	26	byte	• 0 Control is enabled. • 1 Control is disabled.

Table 8: HID Control Modes

Name	ID	Description
CTL_AXIS	0	Drives a stepper with a velocity proportional to the control data.
CTL_BTN_SELECT	1	When pressed, the target HID control's destination mapping (slot, bit, port, and virtual), direction, and deadband will be overwritten with the current HID's values.

Name	ID	Description
CTL_BTN_SELECT_TOGGLE	2	When pressed, the target HID control and the current control's destination mapping, direction, deadband, and speed will be swapped.
CTL_BTN_TOGGLE_DISABLE_MOD	3	When pressed, the target HID control will have its enable state toggled.
CTL_BTN_DISABLE_MOD	4	When pressed, the target HID control will be disabled.
CTL_BTN_TOGGLE_DISABLE_DEST	5	When pressed, the target slot card will have its driver's enable toggled.
CTL_BTN_DISABLE_DEST	6	When pressed, the target slot card will have its driver disabled.
CTL_BTN_POS_1	11	When pressed, the target slot card (stepper) will got its stored position #1.
CTL_BTN_POS_2	12	When pressed, the target slot card (stepper) will got its stored position #2.
CTL_BTN_POS_3	13	When pressed, the target slot card (stepper) will got its stored position #3.
CTL_BTN_POS_4	14	When pressed, the target slot card (stepper) will got its stored position #4.
CTL_BTN_POS_5	15	When pressed, the target slot card (stepper) will got its stored position #5.
CTL_BTN_POS_6	16	When pressed, the target slot card (stepper) will got its stored position #6.
CTL_BTN_POS_7	17	When pressed, the target slot card (stepper) will got its stored position #7.
CTL_BTN_POS_8	18	When pressed, the target slot card (stepper) will got its stored position #8.
CTL_BTN_POS_9	19	When pressed, the target slot card (stepper) will got its stored position #9.

Name	ID	Description
CTL_BTN_POS_10	20	When pressed, the target slot card (stepper) will got its stored position #10.
CTL_BTN_SPEED	21	When pressed, the target HID control's speed will be set to the current control's speed. When released, the value will reset to its default.
CTL_BTN_SPEED_TOGGLE	22	When pressed, the target HID control's speed will either be set to the current control's speed (not toggled), or resetting the control speed to its default (toggled).

MGMSG_MOD_SET_JOYSTICK_MAP_OUT 0x4017

Function: Sent to configure a HID's driver and save it to storage.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x17	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0F	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Port Number	6	byte	
Control Number	7	byte	Control index for devices with multiple interfaces (buttons, knobs, etc.)
Usage Type	8	byte	See HID usage tables from usb.org.
Vendor ID	9	word	
Product ID	11	word	
LED Mode	13	byte	See Table 9 for LED modes.
Color 1 ID	14	byte	See Table 10 for LED color IDs.
Color 2 ID	15	byte	See Table 10 for LED color IDs.
Color 3 ID	16	byte	See Table 10 for LED color IDs.
Source Slots	17	byte	Bitfield indicating which slot will send USB messages to this control. For example, 0x05 (0b00000101) indicates that slot 1 and slot 3 will send control messages to this HID.
Source Bit	18	byte	Unused. Set to 0x00.
Source Port	19	byte	Unused. Set to 0x00.
Source Virtual	20	byte	Unused. Set to 0x00.

MGMSG_MOD_REQ_JOYSTICK_MAP_OUT 0x4018
MGMSG_MOD_GET_JOYSTICK_MAP_OUT 0x4019

Function: Sent to query the configuration of a HID's driver.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x18	
Message ID (upper)	1	0x40	
Port Number	2	byte	
Control Number	3	byte	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x19	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0F	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Port Number	6	byte	
Control Number	7	byte	Control index for devices with multiple interfaces (buttons, knobs, etc.)
Usage Type	8	byte	See HID usage tables from usb.org.
Vendor ID	9	word	
Product ID	11	word	
LED Mode	13	byte	See Table 9 for LED modes.
Color 1 ID	14	byte	See Table 10 for LED color IDs.
Color 2 ID	15	byte	See Table 10 for LED color IDs.

Name	Offset	Format	Notes
Color 3 ID	16	byte	See Table 10 for LED color IDs.
Source Slots	17	byte	Bitfield indicating which slot will send USB messages to this control. For example, 0x05 (0b00000101) indicates that slot 1 and slot 3 will send control messages to this HID.
Source Bit	18	byte	Unused.
Source Port	19	byte	Unused.
Source Virtual	20	byte	Unused.

Table 9: LED Modes

Name	ID	Brief Description
LED_STAGE	0	Color 2 when not moving, color 3 when moving, and color 1 for any invalid or "not ready" state.
LED_DISABLE_ALL	1	Always color 1.
LED_TOGGLE	2	Color 1 if the HID controls the active toggle slot. Otherwise, color 2.
LED_POSITION_1	3	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_2	4	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_3	5	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_4	6	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_5	7	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_6	8	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_7	9	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_8	10	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_9	11	Color 1 if state is on the saved position. Otherwise, color 2.
LED_POSITION_10	12	Color 1 if state is on the saved position. Otherwise, color 2.
LED_SPEED_SWITCH	13	Color 1 if the controlled HID is in the default speed state. Otherwise, color 2.

Table 10: LED Color IDs

Color Name	ID	Red	Green	Blue
Off	0	0	0	0
White	1	1	1	1
Red	2	1	0	0
Green	3	0	1	0
Blue	4	0	0	1
Yellow	5	1	1	0
Aqua	6	0	1	1
Magenta	7	1	0	1
Dim White	8	0.125	0.125	0.125
Dim Red	9	0.125	0	0
Dim Green	10	0	0.125	0
Dim Blue	11	0	0	0.125
Dim Yellow	12	0.125	0.125	0
Dim Aqua	13	0	0.125	0.125
Dim Magenta	14	0.125	0	0.125

MGMSG_MOD_SET_SYSTEM_DIM 0x401A

Function: Sets the maximum brightness of the LEDs connected to the controller.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x1A	
Message ID (upper)	1	0x40	
System Dim (%), D	3	byte	$D \in [0, 100]$
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

MGMSG_MOD_REQ_SYSTEM_DIM 0x401B
MGMSG_MOD_GET_SYSTEM_DIM 0x401C

Function: Sent to request the maximum brightness of the LEDs connected to the controller.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x1B	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x1C	
Message ID (upper)	1	0x40	
System Dim (%), D	3	byte	$D \in [0, 100]$
Parameter 2	2	0x00	
Destination	4	0x01	The destination of the message.
Source	5	0x11	The source of the message.

MGMSG_MCM_REQ_JOYSTICK_DATA 0x402A
MGMSG_MCM_GET_JOYSTICK_DATA 0x402B

Function: Sent to query the slot number controlled by USB devices.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2A	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

(device 1 is a hub)

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2B	
Message ID (upper)	1	0x40	
Length (lower)	2	0x04	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Hub Slot Select	6	byte	
Reserved	7	byte[3]	For internal use only.

(device 1 is not a hub)

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2B	

Name	Offset	Format	Notes
Message ID (upper)	1	0x40	
Length (lower)	2	0x04	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Device 2 Slot Select	6	byte	If no device is connected, then 0xFF.
Device 3 Slot Select	7	byte	If no device is connected, then 0xFF.
Device 4 Slot Select	8	byte	If no device is connected, then 0xFF.
Device 5 Slot Select	9	byte	If no device is connected, then 0xFF.

MGMSG_MCM_SET_SLOT_TITLE 0x402C

Function: Sent to set the slot title for the slot card specified and saves it to internal storage.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2C	
Message ID (upper)	1	0x40	
Length (lower)	2	0x12	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Slot Number	6	word	
Title	8	char[16]	User-defined title for the slot.

MGMSG_MCM_REQ_SLOT_TITLE 0x402D
MGMSG_MCM_GET_SLOT_TITLE 0x402E

Function: Sent to query the slot title for the slot card specified.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2D	
Message ID (upper)	1	0x40	
Slot Number	3	byte	
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2E	
Message ID (upper)	1	0x40	
Length (lower)	2	0x12	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Slot Number	6	word	
Title	8	char[16]	User-defined title for the slot.

MGMSG_MOD_REQ_JOYSTICK_CONTROL 0x402F
MGMSG_MOD_GET_JOYSTICK_CONTROL 0x4030

Function: Sent to query the control information from the USB HID device's report descriptor.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x2F	
Message ID (upper)	1	0x40	
Length (lower)	2	0x03	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Port Number	6	byte	
Control Type	7	byte	• 0 IN control • 1 OUT control
Control Number	8	byte	The index of the control to query. Zero-indexed

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x30	
Message ID (upper)	1	0x40	
Length (lower)	2	0x07	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Port Number	6	byte	
Control Type	7	byte	• 0 IN control • 1 OUT control

Name	Offset	Format	Notes
Control Number	8	byte	The index of the control queried.
HID Usage Page	9	word	See HID usage tables from usb.org.
HID Usage ID	11	word	See HID usage tables from usb.org.

MGMSG_MCM_SET_SOFT_LIMITS 0x403D

Function: Changes the rotational soft limits based on the current encoder position.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x3D	
Message ID (upper)	1	0x40	
Mode	3	byte	• 1 Sets the counter-clockwise soft limit to the current encoder position. • 2 Sets the clockwise high soft limit to the current encoder position. • 3 Removes both the high and low soft limits.
Parameter 2	2	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

MGMSG_MCM_SET_HOMEPARAMS 0x403E

Function: Sent to configure the homing behavior for the stepper.

NOTICE: The response of MGMSG_MCM_GET_HOMEPARAMS must be used as a template for MGMSG_MCM_SET_HOMEPARAMS. Modify the response of the command with the fields provided.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x3E	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0E	
Length (upper)	3	0x00	
Destination	4	0x21 + $SLOT$| 0x80	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.
Slot Card	6	word	
Reserved	8	byte[1]	For internal use only. Use previous values.
Homing Direction	9	byte	• 0 Home in the clockwise direction. • 1 Home in the counter-clockwise direction.
Reserved	10	byte[10]	For internal use only. Use previous values.

MGMSG_MCM_REQ_HOMEPARAMS 0x403F
MGMSG_MCM_GET_HOMEPARAMS 0x4040

Function: Sent to query the homing configuration for the slot card.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x3F	
Message ID (upper)	1	0x40	
Slot Card	3	byte	
Parameter 2	2	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x40	
Message ID (upper)	1	0x40	
Length (lower)	2	0x0E	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$
Slot Card	6	word	
Reserved	8	byte[1]	For internal use only.
Homing Direction	9	byte	• 0 Home in the clockwise direction. • 1 Home in the counter-clockwise direction.
Reserved	10	byte[10]	For internal use only.

MGMSG_MCM_REQ_STAGEPARAMS 0x4042
MGMSG_MCM_GET_STAGEPARAMS 0x4043

Function: Sent to query the set stage parameters of this slot card.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x42	
Message ID (upper)	1	0x40	
Slot Card	3	byte	
Parameter 2	2	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x43	
Message ID (upper)	1	0x40	
Length (lower)	2	0x60	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$
Slot Card	6	word	
Reserved	8	byte[24]	For internal use only.
Counts Per Unit (128 microstep), c	32	dword	The number of 1/128 microsteps per encoder count. This value is not affected by the step mode parameter. $c_{\text{floating}} = \frac{c}{100000}$

Name	Offset	Format	Notes
Minimum Position	36	dword	The smallest encoder value of the stage when homed.
Maximum Position	40	dword	The largest encoder value of the stage when homed.
Reserved	44	byte[30]	For internal use only.
Nanometers per Count	74	float	The number of nanometers per encoder count.
Reserved	78	byte[18]	For internal use only.

Acceleration and Deceleration Conversion

$$\begin{aligned} f(x) &= \frac{x \cdot 2^{-40}}{6.25 \times 10^{-14}} \\ g(x) &= c_{enc} \cdot \frac{128}{c_{\text{floating}}} \cdot x \\ \text{Acceleration } (\text{nm/s}^2) &= g(f(A)) \\ \text{Deceleration } (\text{nm/s}^2) &= g(f(D)) \end{aligned} \tag{4}$$

Velocity Conversion

$$\begin{aligned} f_{max}(x) &= \frac{x \cdot 2^{-18}}{2.50 \times 10^{-7}} \\ f_{min}(x) &= \frac{x \cdot 2^{-24}}{2.50 \times 10^{-7}} \\ g(x) &= c_{enc} \cdot \frac{128}{c_{\text{floating}}} \cdot x \\ \text{Max Velocity } (\text{nm/sec}) &= g(f_{max}(S_{max})) \\ \text{Min Velocity } (\text{nm/sec}) &= g(f_{min}(S_{min})) \end{aligned} \tag{5}$$

MGMSG_MCM_REQ_STATUSUPDATE 0x4044
MGMSG_MCM_GET_STATUSUPDATE 0x4045

Function: Send to query the status of the stepper, with additional information specific to MCM stepper cards.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x44	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x21 + $SLOT$	The destination of the message. $SLOT \in [0, 7]$
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x45	
Message ID (upper)	1	0x40	
Length (lower)	2	0x12	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x21 + $SLOT$	The source of the message. $SLOT \in [0, 7]$
Slot	6	word	
Position	8	long	
Encoder Count	12	long	
Status Flags	16	dword	
Stored Position	20	byte	The stored position the stepper is currently on. 0xFF if the stepper is not on a stored position.

Name	Offset	Format	Notes
Raw Encoder Count	21	long	Unprocessed encoder counter. Applicable to linear BISS encoders.

The position, encoder count, and status flag fields are identical in value as those in MGMSG_MOT_GET_STATUSUPDATE;

MGMSG_MCM_SET_ALLOWED_DEVICES 0x40F2

Function: Set the allowed devices on a slot card. In order for a device to be driven on a slot card, its device signature must be in the allowed devices list. Using 0xFFFF for the device ID acts as a wildcard for the passed slot type. If too many devices are allowed, the controller will reject the command.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xF2	
Message ID (upper)	1	0x40	
Length	2	word	Little endian. size = 2+n
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
Slot Card	6	word	
Device Signatures	8	byte[n][4]	Array of device signatures.

NOTICE: Due to internal limitations, no more than 23 device signatures may be allowed on each slot.

Table 11: Device Signature

Field Name	Offset	Size (bytes)
Default Slot Type	0	2
Device ID	2	2

MGMSG_MCM_REQ_ALLOWED_DEVICES 0x40F3
MGMSG_MCM_GET_ALLOWED_DEVICES 0x40F4

Function: Sent to query the allowed devices on a slot card.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0xF3	
Message ID (upper)	1	0x40	
Slot Card	3	byte	
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xF4	
Message ID (upper)	1	0x40	
Length	2	word	Little endian. size = 2+4 × n
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Slot Card	6	word	
Device Signatures	8	byte[n][4]	Array of device signatures.

MGMSG_MCM_EFS_REQ_HWINFO 0x40F8
MGMSG_MCM_EFS_GET_HWINFO 0x40F9

Function: Sent to query the hardware info for the MCM-device's embedded file system.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0xF8	
Message ID (upper)	1	0x40	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

Controller supports the Embedded File System version 1

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xF9	
Message ID (upper)	1	0x40	
Length (lower)	2	0x16	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
File System Identifier	6	char[3]	"EFS"
EFS Version Identifier	9	byte	0
Page Size	10	word	The size of a page in bytes.
Pages Supported	12	word	The number of pages in the file system.
Reserved	14	byte[8]	For internal use only.
Maximum Files	22	word	The maximum number of files supported by the system.

Name	Offset	Format	Notes
Files Remaining	24	word	The number of files that can be allocated.
Pages Remaining	26	word	The number of pages remaining. They may not be contiguous.

MGMSG_MCM_EFS_SET_FILEINFO 0x40FA

Function: Sent to create or delete files on the file system.

NOTICE: Existing files *cannot* modify their attributes or file size. The file *must* be deleted first.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFA	
Message ID (upper)	1	0x40	
Length (lower)	2	0x04	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
File Identifier	6	byte	Unique file identifier.
File Attributes	7	byte	See the bit field diagram.
File Length	8	word	Length of the file in pages. When zero, this will delete an existing file. When non-zero, this will create a new file.

Bit Field for File Attributes

Mask	Bits	Name	Description
0x01	0	APT Read Allowed	Set when APT commands can read from the file.
0x02	1	APT Write Allowed	Set when APT commands can write to the file.
0x04	2	APT Delete Allowed	Set when APT commands can delete the file.
0x08	3	Firmware Read Allowed	Set when firmware can read from the file.
0x10	4	Firmware Write Allowed	Set when firmware can write to the file.
0x20	5	Firmware Delete Allowed	Set when firmware can delete the file.
0xC0	6 - 7	Reserved	For internal use only.

MGMSG_MCM_EFS_REQ_FILEINFO 0x40FB
MGMSG_MCM_EFS_GET_FILEINFO 0x40FC

Function: Sent to query file information from the file system.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFB	
Message ID (upper)	1	0x40	
Length (lower)	2	0x01	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
File Identifier	6	byte	Unique file identifier.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFC	
Message ID (upper)	1	0x40	
Length (lower)	2	0x06	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
File Identifier	6	byte	Unique file identifier.
File Exists	7	byte	0 when the file does *not* exist.
File is Owned	8	byte	Indicates that the file is owned by the firmware.
File Attributes	9	byte	See the bit field diagram.
File Size	10	word	Length of the file in pages.

MGMSG_MCM_EFS_SET_FILEDATA 0x40FD

Function: Sets the data of a given file.

NOTICE: In order to write a file, it must be allocated and must be writable by external sources. If it is not writable, the command is discarded. If the file is not writable by APT, the command is discarded.
Any data written outside the address space of the file will be lost.
The size of the extended data must be under the maximum size or data will be lost.

SET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFD	
Message ID (upper)	1	0x40	
Length	2	word	Little endian. size = 5+D
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
File Identifier	6	byte	Identifier for the file to write.
File Address	7	dword	The file address to begin writing data.
Binary Data	11	byte[D]	The data to write to the file.

MGMSG_MCM_EFS_REQ_FILEDATA 0x40FE
MGMSG_MCM_EFS_GET_FILEDATA 0x40FF

Function: Send to query the data on a file.

NOTICE: The number of bytes returned may not equal the number of bytes requested.
No data will be returned for addresses outside of the valid file address space.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFE	
Message ID (upper)	1	0x40	
Length (lower)	2	0x07	
Length (upper)	3	0x00	
Destination	4	0x11| 0x80	The destination of the message.
Source	5	0x01	The source of the message.
File Identifier	6	byte	Identifier for the file to read.
Read Start Address	7	dword	The file address to begin reading from.
Bytes to Read	11	word	The maximum number of bytes to read.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0xFF	
Message ID (upper)	1	0x40	
Length	2	word	Little endian. size = 5+D
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
File Identifier	6	byte	Identifier for the file to read.
Read Start Address	7	dword	The file address that the data originated from.

Name	Offset	Format	Notes
Data	11	byte[*D*]	The data read from the file.

MGMSG_MCM_LUT_REQ_LOCK 0x4101
MGMSG_MCM_LUT_GET_LOCK 0x4102

Function: Sent to query the status of the LUT file lock.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x01	
Message ID (upper)	1	0x41	
Parameter 1	2	0x00	
Parameter 2	3	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x02	
Message ID (upper)	1	0x41	
Look-Up Tables Lock	3	byte	• 0 LUTs are unlocked. • 1 LUTs are locked.
Parameter 2	2	0x00	
Destination	4	0x01	The destination of the message.
Source	5	0x11	The source of the message.

MGMSG_MCM_REQ_PNPSTATUS 0x4108
MGMSG_MCM_GET_PNPSTATUS 0x4109

Function: Sent to query the plug-and-play status of a given slot card.

REQUEST:

Name	Offset	Format	Notes
Message ID (lower)	0	0x08	
Message ID (upper)	1	0x41	
Slot Number	3	byte	
Parameter 2	2	0x00	
Destination	4	0x11	The destination of the message.
Source	5	0x01	The source of the message.

GET:

Name	Offset	Format	Notes
Message ID (lower)	0	0x09	
Message ID (upper)	1	0x41	
Length (lower)	2	0x06	
Length (upper)	3	0x00	
Destination	4	0x01 | 0x80	The destination of the message.
Source	5	0x11	The source of the message.
Slot Number	6	word	
Plug-and-Play Error Flags	8	dword	If all bits are 0, then a device is connected and runable.

Bit Field for Plug-and-Play Error Flags

Mask	Bits	Name	Description
0x00000001	0	No Device Connected	Set when no device was detected on the slot card.

Mask	Bits	Name	Description
0x00000002	1	General Device Error	Set whenever a device error is raised.
0x00000004	2	Unknown Device File Version	Set when the connected device's file version presented is not known.
0x00000008	3	Device File Corruption	Set when the connected device's file checksum mismatches with the calculated checksum.
0x00000010	4	Serial Number Mismatch	Set when serial number checking is enabled and the connected device's serial number does not match what was saved.
0x00000020	5	Device Signature Not Allowed	Set when the connected device's signature is not allowed on the slot.
0x00000040	6	General Configuration Error	Set when any error occurred during the configuration step (after device detection).
0x00000080	7	Device Configuration Set Miss	Set when a configuration for the device could not be found for the attached slot card.
0x00000100	8	Configuration Struct Miss	Set when a device configuration is missing data necessary to build the device.
0xFFFFFE00	9 - 31	Reserved	For internal use only.