GLK240128-25/GLT240128

Including GLK240128-25-USB, GLK240128-25-422, GLT240128-USB, and GLT240128-422

Technical Manual

Revision 2.9

PCB Revision: 4.0 or Higher

Firmware Revision: 8.1 or Higher

Revision History

Revision	Date	Description	Author
2.9	May 21, 2014	Revision to Commands for Firmware Revision 8.5	Martino
2.8	March 12, 2014	Revision and correction to Colour in Ordering Options	Martino
2.7	September 9, 2013	Corrected Scripted Button/Key and Keypad Brightness Commands	Clark
2.6	July 10, 2013	Updated Data Packet Size Definitions	Clark
2.5	December 13, 2012	Added Firmware Revision 8.4 Commands	Clark
2.4	October 23, 2012	Added Alternate Power requirement for YG units	Clark
2.3	October 13, 2011	Major Command Additions	Clark
2.2	March 8, 2011	Updated Electrical Specifications	Clark
2.1	January 27, 2011	Filesystem Command Updates for Firmware Revision 8.1	Clark
2.0	November 3, 2010	Initial Release	Clark

Contents

1 Introduction 1

2 Quick Connect Guide 2

2.1 Available Headers 2

2.2 Standard Module 3

Recommended Parts 3

Serial Connections 3

I^2C Connections 4

2.3 USB Module 5

Recommended Parts 5

USB Connections 5

2.4 RS422 Module 6

RS422 Connections 6

3 Software 7

3.1 MOGD# 7

3.2 Firmware Upgrade 8

3.3 Application Notes 8

4 Hardware 9

4.1 Standard Model 9

Extended Communication/Power Header 9

Serial DB9 Connector 9

Power Through DB9 Jumper 10

Protocol Select Jumpers 10

Hardware Lock 10

4.2 USB Model 11

Mini USB Connector 11

Alternate USB Header 11

Alternate Power Connector 11

4.3 RS422 Model 12

RS422 Header 12

Alternate Power Connector 12

4.4 GLK Model 13
Keypad Header 13
4.5 GLT Model 14
Touch Screen 14
Coordinate Mode 14
Region Mode 14
4.6 Common Features 15
General Purpose Outputs 15
Dallas One-Wire Connector 15
5 Troubleshooting 16
5.1 Power 16
5.2 Display 16
5.3 Communication 17
5.4 Manual Override 17
6 Commands 18
6.1 Communication 18
6.2 Text 20
6.3 Drawing 22
6.4 Fonts 27
Font File Creation 28
6.5 Bitmaps 30
Bitmap File Creation 31
Bitmap Masking 32
6.6 9-Slices 33
9-Slice File Creation 34
6.7 Animations 35
Animation File Creation 36
6.8 General Purpose Output 37
6.9 Piezo Buzzer 38
6.10 Keypad 39
6.11 Touchpad 41
6.12 Display Functions 44

6.13 Scripting 45
6.14 Filesystem 46
File Transfer Protocol 48
XModem Transfer Protocol 49
6.15 Data Security 51
6.16 Miscellaneous 52
7 Appendix 54
7.1 Command Summary 54
7.1 Block Diagram 59
7.2 Environmental Specifications 59
7.3 Electrical Tolerances 59
7.4 Dimensional Drawings 60
7.1 Optical Characteristics 62
8 Ordering 62
8.1 Part Numbering Scheme 62
8.2 Options 62
8.3 Accessories 63
9 Definitions 65
10 Contact 65

1 Introduction

Figure 1: GLK240128-25/GLT240128 Display

The GLK240128-25/GLT240128 is an intelligent graphic liquid crystal display engineered to quickly and easily add an elegant creativity to any application. In addition to the RS232, TTL and I2C protocols available in the standard model, USB and RS422 communication models allow the GLK240128-25/GLT240128 to be connected to a wide variety of host controllers. Communication speeds of up to 115.2kbps for serial protocols and 100kbps for I^2C ensure lightning fast text and graphic display.

The simple command structure permits easy software control of many settings including backlight brightness, screen contrast, and baud rate. On board memory provides a whopping 256KB of customizable fonts and bitmaps to enhance the graphical user experience.

User input on the GLK240128-25 is available through a five by five matrix style keypad or a resistive touch overlay on the GLT240128. Six general purpose outputs provide simple switchable five volt sources on each model. In addition, a versatile Dallas One-Wire header provides a communication interface for up to thirty-two devices.

The versatile GLK240128-25/GLT240128, with all the features mentioned above, is available in a variety of colour, voltage, and temperature options to suit almost any application.

2 Quick Connect Guide

2.1 Available Headers

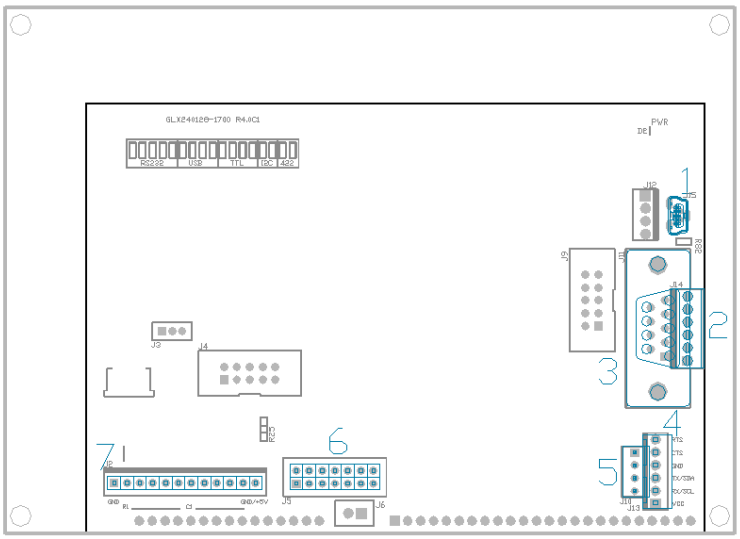


Figure 2: GLK240128-25/GLT240128 Header Locations

Table 1: List of Available Headers

#	Header	Mate	Population
1	Mini USB Connector	EXTMUSB3FT/INTMUSB3FT	USB Model Only
2	RS422 Terminal Block	16-30 AWG Wire	422 Model Only
3	DB9 Serial Header	CSS1FT/CSS4FT	Standard Model Only
4	Extended Communication/Power Connector	ESCCPC5V/BBC	Standard Model Only
5	Alternate Power Connector	PCS	422 and USB Models Only
6	GPO Header	None Offered	All Models
7	Keypad	KPP4x4	GLK Model Only

2.2 Standard Module

The standard version of the GLK240128-25/GLT240128 allows for user configuration of two common communication protocols. First, the unit can communicate using serial protocol at either RS323 or TTL voltage levels. Second, it can communicate using the Inter-Integrated Circuit connect, or I^2C protocol. Connections for each protocol can be accessed through the four pin Communication/Power Header as outlined in the Serial Connections and I^2C Connections sections below.

Recommended Parts

Figure 3: Extended Communication/Power Cable (ESCCPC5V)

The most common cable choice for any standard Matrix Orbital display, the Extended Communication/ Power Cable offers a simple connection to the unit with familiar interfaces. DB9 and floppy power headers provide all necessary input to drive your display.

Figure 4: Breadboard Cable (BBC)

For a more flexible interface to the GLK240128-25/GLT240128, a Breadboard Cable may be used. This provides a simple four wire connection that is popular among developers for its ease of use in a breadboard environment.

Serial Connections

Serial protocol provides a classic connection to the GLK240128-25/GLT240128. The Extended Communication/Power Cable is most commonly used for this set up as it provides connections for DB9 serial and floppy power cables. To place your board in Serial mode, adhere to the steps laid out below.

1. Set the Protocol Select jumpers.
   - RS232: Connect the five jumpers* in the 232 protocol box with the zero ohm jumper resistors provided or an alternate wire or solder solution.
   - TTL: Connect the four jumpers* in the TTL protocol box.

***Note:** Jumpers must be removed from all protocol boxes save for the one in use.

2. Make the connections.
    a. Connect the six pin female header of the Communication/Power Cable to the Extended Communication/Power Header of your GLK240128-25/GLT240128.
    b. Insert the male end of your serial cable to the corresponding DB9 header of the Extended Communication/Power Cable and the mate the female connector with the desired communication port of your computer.
    c. Select an unmodified floppy cable from a PC power supply and connect it to the power header of the Communication/Power Cable.
3. Create.
    - MOGD# or a terminal program will serve to get you started, and then you can move on with your own development. Instructions for the former can be found below and a variety of application notes are available for the latter at www.matrixorbital.ca/appnotes.

I^2C Connections

A more advanced connection to the GLK240128-25/GLT240128 is provided by the I^2C protocol setting. This is best accomplished using a breadboard and the Breadboard Cable. Power must be supplied from your breadboard or another external source. To dive right into your application and use the GLK240128-25/GLT240128 in I^2C mode, get started with the guidelines below.

1. Set the Protocol Select switches.
    - I^2C: Ensure that the two I^2C jumpers in the corresponding protocol box are connected while all others are open.
2. Make the connections.
    a. Connect the Breadboard Cable to the Extended Communication/Power Header on your GLK240128-25/GLT240128 and plug the four leads into your breadboard. The red lead will require power, while the black should be connected to ground, and the green and yellow should be connected to your controller clock and data lines respectively.
    b. Pull up the clock and data lines to five volts using a resistance between one and ten kilohms on your breadboard.
3. Create.
    - This time you're on your own. While there are many examples within the Matrix Orbital AppNote section, www.matrixorbital.ca/appnotes, too many controllers and languages exist to cover them all. If you get stuck in development, it is possible to switch over to another protocol on the standard board, and fellow developers are always on our forums for additional support.

2.3 USB Module

The GLK240128-25-USB/GLT240128-USB offers a single USB protocol for easy connection to a host computer. The simple and widely available protocol can be accessed using the on board mini B style USB connector as outlined in the USB Connections section.

Recommended Parts

Figure 5: Mini USB Cable (EXTMUSB3FT)

The External Mini USB cable is recommended for the GLK240128-25-USB/GLT240128-USB display. It will connect to the miniB style header on the unit and provide a connection to a regular A style USB connector, commonly found on a PC.

USB Connections

The USB connection is the quickest, easiest solution for PC development. After driver installation, the GLK240128-25-USB/GLT240128-USB will be accessible through a virtual serial port, providing the same result as a serial setup without the cable hassle. To connect to your GLK240128-25-USB/GLT240128-USB please follow the steps below.

1. Set the Protocol Select jumpers.
   - USB: The GLK240128-25-USB/GLT240128-USB offers USB protocol only. Model specific hardware prevents this unit from operating in any other protocol, and does not allow other models to operate in USB. Protocol Select jumpers on the USB model cannot be moved.
2. Make the connections.
   - Plug the mini-B header of your External Mini USB cable into your GLK240128-25-USB/GLT240128-USB and the regular USB header into your computer USB jack*.
3. Install the drivers.
   a. Download the latest drivers at www.matrixorbital.ca/drivers, and save them to a known location.
   b. When prompted, install the USB bus controller driver automatically
   c. If asked, continue anyway, even though the driver is not signed
   d. When the driver install is complete, your display will turn on, but communication will not yet be possible.
   e. At the second driver prompt, install the serial port driver automatically
   f. Again, if asked, continue anyway
4. Create.
   - Use MOGD# or a terminal program to get started, and then move on with your own development. Instructions for the former can be found below and a number of application notes are available for the latter at www.matrixorbital.ca/appnotes.

***Note:** The YG version of the GLK240128-25-USB/GLT240128-USB will require the alternate power connection.

2.4 RS422 Module

The GLK240128-25-422/GLT240128-422 provides an industrial alternative to the standard RS232 communication protocol. Rather than single receive and transmit lines, the RS422 model uses a differential pair for receive and transmit signals to reduce degradation and increase transmission lengths. Power can be transmitted at distance to a -VPT module or supplied from the immediate vicinity to a regular or -V unit. RS422 signals are available in a six pin connector as described in the RS422 Connections section.

RS422 Connections

The GLK240128-25-422/GLT240128-422 provides a robust RS422 interface to the display line. For this interface, a series of six wires are usually screwed into the RS422 terminal block provided. An alternate header is also available to provide local power to a regular or -V unit. To connect to your GLK240128-25-422/GLT240128-422, adhere to the steps laid out below.

1. Set the Protocol Select jumpers.
   - RS422: The GLK240128-25-422/GLT240128-422 offers only RS422 protocol and does not require any jumper changes.
2. Make the connections.
   a. Screw one wire; sized 16 to 30 on the American Wire Gauge, into each of the six terminal block positions. When local power is supplied, a floppy cable may link to the alternate power header.
   b. Connect the Vcc wire to the positive terminal of your power supply and the GND terminal to the negative or ground lead to provide appropriate power as per Voltage Specifications.
   c. Secure the A and B wires to your non-inverting and inverting output signals respectively, while attaching the Z and Y wires to your inverting and non-inverting inputs.
3. Create.
   a. In a PC environment, MOGD# or a terminal program will serve to get you started. In addition, a variety of application notes are available online in a number of different languages to aid in the development of a host controller. Instructions for these programs can be found below and the simple C# example at www.matrixorbital.ca/appnotes is a great first programming reference.

3 Software

The multiple communication protocols available and simple command structure of the GLK240128-25/GLT240128 means that a variety of applications can be used to communicate with the display. Text is sent to the display as a character string, for example, sending the decimal value 41 will result in an 'A' appearing on the screen. A single control character is also available. Commands are merely values prefixed with a special command byte, 254 in decimal.

Table 2: Reserved Control Characters

Control Characters			
7	Bell / Sound Buzzer	10	Line feed / New line

Once the correct communication port is identified, the following communication settings can be applied to communicate correctly with the GLK240128-25/GLT240128.

Table 3: Communication Settings

BPS	Data Bits	Parity	Stop Bits	Flow Control
19200	8	None	1	None

Finally, with a communication port identified and correctly setup simple text strings or even command bytes can easily be transmitted to control your display.

3.1 MOGD#

The Matrix Orbital Graphic Display interface, MOGD#, is offered as a free download from www.matrixorbital.ca/software/software_graphic. It provides a simple graphical interface that allows settings, fonts, and bitmaps to be easily customised for any application.

While monochromatic bitmaps can easily be created in virtually any image editing program, MOGD# provides an extensive font generation suite to stylize your display to any project design. In addition to standard font wide modifications, character ranges can be specified by start and end values to eliminate unused symbols, and individual glyphs can be modified with a double click. Finally, text spacing can be tailored and a complete font library built with your Matrix Orbital graphic display.

Like uProject, MOGD# offers a scripting capability that provides the ability to stack, run, and save a series of commands. The most basic function is the Send Numeric tool which is used to transmit a string of values to the display to write text or execute a command.

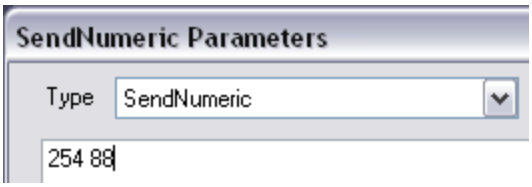


Figure 6: MOGD# Command Example

Again, the clear screen command is sent to a connected display, this time using the MOGD# Send Numeric function command style. Scripts can be run as a whole using the Play button from the toolbar or as single commands by selecting Step; once executed it must be Reset. Before issuing commands, it is a good idea to ensure communication with a display is successful using the autodetect button.

This program provides both a staging areas for your graphics display and a proving ground that will prepare it for any application environment.

3.2 Firmware Upgrade

Beginning with revision 8.1, the firmware of the GLK240128-25/GLT240128 can be upgraded in the field. All firmware revisions can be installed using software found at www.matrixorbital.ca/software/GLT Series.

If it is necessary to forgo all current and future upgrades to the filesystem and subsequent commands, firmware revision 8.0 may be ordered as a part of a custom order. Please use the Contact section to request more information from the Matrix Orbital sales team.

3.3 Application Notes

Full demonstration programs and code are available for Matrix Orbital displays in the C# language from Simple C# AppNote Pack in the Application Note section at www.matrixorbital.ca/appnotes. Difficulty increases from beginner, with the Hello World program, to advanced with the Dallas One-Wire temperature reading application.

Many additional applications are available in a number of different programming languages. These programs are meant to showcase the capability of the display and are not intended to be integrated into a final design. For additional information regarding code, please read the On Code document also found on the support site.

4 Hardware

4.1 Standard Model

Extended Communication/Power Header

Figure 7: Extended Communication/Power Header

Table 4: Extended Communication/Power Pinout

Pin	Function
1	Vcc
2	Rx (SCL)
3	Tx (SDA)
4	Gnd
5	CTS
6	RTS

The Extended Communication/Power Header provides a standard connector for interfacing to the GLK240128-25/GLT240128. Voltage is applied through pins one and four of the four pin Communication/Power Header. Please ensure the correct voltage input for your display by referencing Voltage Specifications before connecting power. Pins two and three are reserved for serial transmission, using either the RS-232/TTL or clocking data through the I²C protocol, depending on what has been selected by the Protocol Select Jumpers. Pins five and six can be used for serial transmission hardware flow control, and are ignored for I²C communications. The Molex 22-04-1061 style header used can be mated to a number of connectors, a 22-01-1062 for example.

Serial DB9 Connector

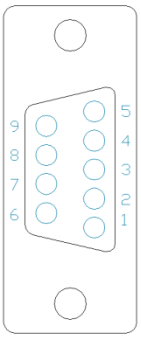


Figure 8: Serial DB9 Connector

Table 5: Serial DB9 Pinout

Pin	Function
2	Tx
3	Rx
5	Gnd
7	CTS
8	RTS
9	NC/Vcc*

The GLK240128-25/GLT240128 provides a DB-9 Connector to readily interface with serial devices using EIA232 standard signal levels. It is also possible to communicate at TTL levels of 0 to +5V by setting the Protocol Select Jumpers to TTL. As an added feature it is also possible to apply power through pin 9 of the DB-9 Connector in order to reduce cable clutter. A standard male DB9 header will provide the perfect mate for this connector.

***Note:** Do not apply voltage through pin 9 of the DB-9 Connector AND through the Communication/Power Header at the same time.

Power Through DB9 Jumper

In order to provide power through pin 9 of the DB-9 Connector you must connect the Power Through DB-9 Jumper labelled R82, as illustrated below. This connection can be made using a zero ohm resistor, recommended size 0603, or a solder bridge. The GLK240128-25/GLT240128 allows all voltage models to use the power through DB-9 option, see the Voltage Specifications for power requirements.

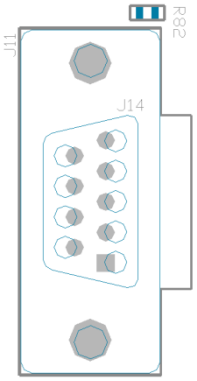


Figure 9: Power Through DB9 Jumper

Protocol Select Jumpers

The Protocol Select Jumpers provide the means necessary to toggle the GLK240128-25/GLT240128 between RS-232, TTL and I²C protocols. As a default, the jumpers are set to RS-232 mode with solder jumps on the RS232 jumpers. In order to place the display module in I²C mode you must first remove the solder jumps from the RS232 jumpers and then place them on the I²C jumpers. The display will now be in I²C mode and have a default slave address of 80, unless changed with the appropriate command. Similarly, in order to change the display to TTL mode, simply remove the zero ohm resistors from the RS232 or I²C jumpers and solder them to the TTL jumpers.

Hardware Lock

The Hardware Lock allows fonts, bitmaps, and settings to be saved, unaltered by any commands. By connecting the two pads near the memory chip, designated R60, with a zero ohm resistor, the display will be locked. This supersedes the data lock command and cannot be circumvented by any software means. To unlock the display and make changes simply remove the jumper.

4.2 USB Model

Mini USB Connector

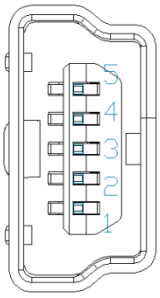


Figure 10: Mini USB Connector

Table 6: Mini USB Pinout

Pin	Function
1	Vcc
2	D-
3	D+
5	Gnd

The GLK240128-25-USB/GLT240128-USB comes with a familiar Mini USB Connector to fulfill both communication and power needs*. The standard MiniB style header can be connected to any other USB style using the appropriate cable. Most commonly used with a PC, this connection creates a virtual com port that offers a simple power solution with a familiar communication scheme.

Alternate USB Header

Some advanced applications may prefer the straight four pin connection offered through the Optional Alternate USB Header. This header offers power and communication access in a simple interface package. The Optional Alternate USB Header may be added to the GLK240128-25-USB/GLT240128-USB for an added charge as part of a custom order. Please use the Contact section to request more information from the friendly Matrix Orbital sales team.

Alternate Power Connector

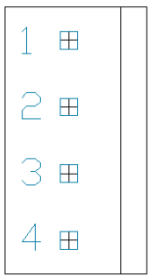


Figure 11: Alternate Power Connector

Table 7: Alternate Power Pinout

Pin	Function
1	Vcc
2	Gnd
3	Gnd
4	NC

The Alternate Power Connector provides the ability to power the GLK240128-25-USB/GLT240128-USB using a second cable*. The Tyco 171825-4 style header is particularly useful for connecting to an unmodified floppy power cable, a 171822-4 for example, from a PC power supply for a simple bench power solution.

***Note:** The YG version of the GLK240128-25-USB/GLT240128-USB may draw more than the 500mA of current permitted by USB standards and will require the alternate power connection.

4.3 RS422 Model

RS422 Header

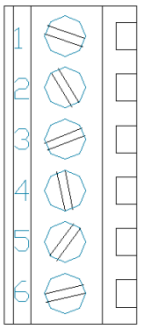


Figure 12: RS422 Header

Table 8: RS422 Pinout

Pin	Function
1	Gnd
2	Rx (Y)
3	Inv Rx (Z)
4	Inv Tx (B)
5	Tx (A)
6	Vcc

The six pin RS422 interface header of the GLK240128-25-422/GLT240128-422 offers power and ground connections as well as two differential pair communication lines. Regular and inverted lines are provided for both receive and transmit signals. Power is supplied locally to the regular or –V variants while the –VPT can receive power over a distance. The Tyco 282834-6 style header is most suited to a simple wire connection.

Alternate Power Connector

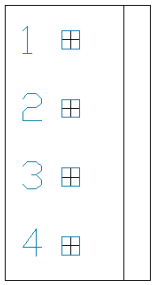


Figure 13: Alternate Power Connector

Table 9: Alternate Power Pinout

Pin	Function
1	Vcc
2	Gnd
3	Gnd
4	NC

The Alternate Power Connector provides the ability to power the GLK240128-25-USB/GLT240128-USB using a second cable. The Tyco 171825-4 style header is particularly useful for connecting to an unmodified floppy power cable, a 171822-4 for example, from a PC power supply for a simple bench power solution.

4.4 GLK Model

Keypad Header

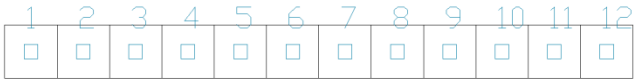


Figure 14: Keypad Header

Table 10: Keypad Pinout

Pin	Function	Pin	Function
1	Gnd	7	Column 1
2	Row 1	8	Column 2
3	Row 2	9	Column 3
4	Row 3	10	Column 4
5	Row 4	11	Column 5
6	Row 5	12	Gnd/Vcc*

To facilitate user input, the GLK240128-25 provides a Keypad Interface Connector which allows a matrix style keypad of up to twenty-five keys to be directly connected to the display module. Key presses are generated when a short is detected between a row and a column. When a key press is generated, a character specific to that key press is automatically sent on the Tx communication line. If a synchronous read method is desired in serial mode*, the "Auto Transmit Keypress" function can be turned off to allow the key presses to remain in the buffer so that they may be polled. The character that is associated with each key press may also be altered using the "Assign Key Codes" command. The straight twelve pin header of the Keypad Interface Connector will interface to a variety of different devices including the Matrix Orbital KPP4x4 keypad.

***Note:** In I^2C mode, the "Auto Transmit Keypress" function should always be on, keypresses should not be polled.

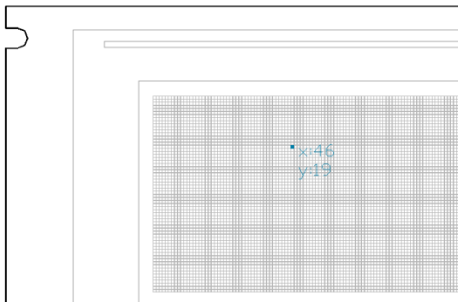
****Note:** The Ground / +5V pin is toggled by the jumper to the right of the keypad connector. Jump pads 1 & 2 for +5V or 2 & 3 for GND.

4.5 GLT Model

Touch Screen

The GLT240128 facilitates user touch input in one of two distinct ways. Coordinate mode will report events by supplying their exact position on the screen. Region mode will report events within defined boundaries on the screen. Both modes are outlined below.

Coordinate Mode

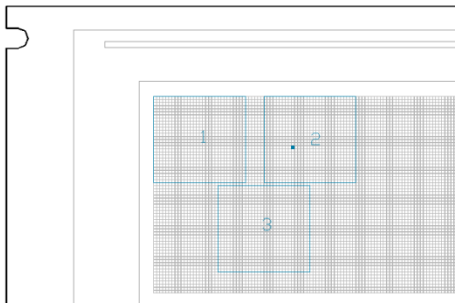


In coordinate mode all touch events are reported using three single byte values. First, the type of event is transmitted, followed by the x and y coordinates of its position. Pressure and drag thresholds must be exceeded for an event to be registered. A low drag threshold will result in greater tracking accuracy but transmits much more data to the host. Care should be taken to find balance. This mode offers a great degree of flexibility and creativity.

Table 11: Coordinate Mode Event Prefixes

Return Value	1	2	4
Touch Event	Press	Release	Drag

Region Mode



A simpler, keypad style alternative to coordinate mode, region mode offers only a single byte for each touch event. Unique regions are created by specifying a position, size, and return values. A value corresponding to a specific region is returned when an event occurs within its bounds. Events outside of regions result in transmission of the value 255. Regions can be deleted individually or collectively when no longer needed. This mode allows quick and easy set up.

Table 12: Region Mode Event Responses

Return Value	Key Down	Key Up	Key Down	255
Touch Event	Press	Release	Drag	Out of Region

4.6 Common Features

General Purpose Outputs

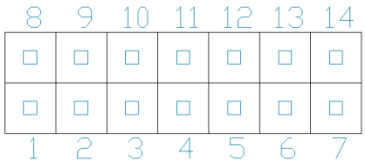


Figure 15: GPO Header

Table 13: GPO Pinout

Pin	Function	Pin	Function
1	GPO 1	8	Gnd
2	GPO 2	9	Gnd
3	GPO 3	10	Gnd
4	GPO 4	11	Gnd
5	GPO 5	12	Gnd
6	GPO 6	13	Gnd
7	Vcc	14	Gnd

A unique feature of the GLK240128-25/GLT240128 is the ability to control relays* and other external devices using either one or six General Purpose Outputs. Each can source up to 10mA of current at five volts when on or sink 20mA at zero volts when off. The two row, fourteen pin header can be interfaced to a number of female connectors to provide control to any peripheral devices required.

***Note:** If connecting a relay, be sure that it is fully clamped using a diode and capacitor in order to absorb any electro-motive force (EMF) which will be generated.

Dallas One-Wire Connector

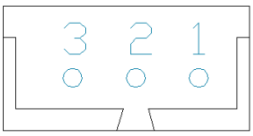


Figure 16: Dallas One-Wire Connector

Table 14: Dallas One-Wire Pinout

Pin	Function
1	Vcc
2	D
3	Gnd

In addition to the six general purpose outputs the GLK240128-25/GLT240128 offers an Optional Dallas One-Wire bridge, to allow for an additional thirty two one-wire devices to be connected to the display. This header can be populated with a Tyco 173979 connector at an added cost by custom order only. Please use the Contact section to request more information from the Matrix Orbital sales team.

5 Troubleshooting

5.1 Power

In order for your Matrix Orbital display to function correctly, it must be supplied with the appropriate power. If the power LED near the top right corner of the board is not illuminated, power is not applied correctly. Try following the tips below.

- First, check the power cable which you are using for continuity. If you don't have an ohm meter, try using a different power cable, if this does not help try using a different power supply.
- If power is applied through the DB9 connector, ensure that the Power Through DB9 Jumper is connected.
- If changes have been made to the protocol select block, ensure all the appropriate protocol select jumpers are connected and all unused protocol jumpers are disconnected.
- The last step will be to check the interface connector in use on your display. If the power connections have become loose, or you are unable to resolve the issue, please Contact Matrix Orbital for more information.

5.2 Display

If your display is powered successfully, the Matrix Orbital logo, or user created screen should display on start up. If this is not the case, check out these tips.

- Ensure the contrast is not too high or too low. This can result in a darkened or blank screen respectively. See the Manual Override section to reset to default.
- Make sure that the start screen is not blank. It is possible to overwrite the Matrix Orbital logo start screen, if this happens the screen may be blank. Try writing to the display to ensure it is functional, after checking the contrast above.

5.3 Communication

When communication of either text or commands is interrupted, try the steps below.

- First, check the communication cable for continuity. If you don't have an ohm meter, try using a different communication cable. If you are using a PC try using a different Com/USB Port.
- Next, please ensure that the display module is set to communicate on the protocol that you are using, by checking the Protocol Select Jumpers.
- In serial and USB protocols, ensure that the host system and display module are both communicating on the same baud rate. The default rate for the display module is 19200 bps.
- Match Rx from your display to the transmitting pin from your host and the Tx pin to the receiving pin.
- If you are communicating to the display via I²C* please ensure that the data is being sent to the correct address. The default slave address for the display module is 80.
- In I²C mode, connect Rx to the clock line of your controller and Tx to the data output.
- Unlock the display. See the Set and Save Data Lock command for more info.
- Finally, you may reset the display to its default settings using the Manual Override procedure outlined below.

***Note:** I²C communication will always require pull up resistors on SCL and SDA of one to ten kilohms.

5.4 Manual Override

Should the settings of your display become altered in a way that dramatically impacts usability, the default settings can be temporarily restored. To override the display, please follow the steps below.

1. Disconnect power from your display.
2. Place a jumper on the two manual override pins, for the GLK240128-25 model these are the middle two keypad pins, for the GLT240128 these are the two pins near the keypad header.
3. Reconnect power to your unit, and wait for the start screen before removing the jumper. Please note the jumper will adversely affect GLT240128 performance if left in place during use.
4. Settings will be temporarily** overridden to the defaults listed in the Manual Override Settings table. At this point any important settings, such as contrast, backlight, or baud rate, should not only be set but saved so they remain when the override is removed.

Parameter	Value
Backlight	255
Contrast	128
Baud Rate	19200
I²C Address	80

Table 15: Manual Override Settings

****Note:** The display module will revert back to the old settings once turned off, unless desired settings are saved.

6 Commands

6.1 Communication

1.1 Change Baud Rate	Dec	254 57	Speed	v8.0
	Hex	FE 39	Speed	
	ASCII	■ 9	Speed	

Immediately changes the baud rate. Not available in I2C. Baud rate can be temporarily forced to 19200 by a manual override.

Speed	Byte	Valid settings shown below.

Table 16: Accepted Baud Rate Values

Rate	9600	14400	19200	28800	38400	57600	76800	115200
Speed	207	138	103	68	51	34	25	16

1.2 Change I2C Slave Address	Dec	254 51	Address	v8.0
	Hex	FE 33	Address	
	ASCII	■ 3	Address	

Immediately changes the I2C write address. Only even values are permitted as the next odd address will become the read address. Default is 80.

Address	Byte	Even value.

1.3 Transmission Protocol Select	Dec	254 160	Protocol	v8.0
	Hex	FE A0	Protocol	
	ASCII	■ á	Protocol	

Selects the protocol used for data transmission from the display. Data transmission to the display is not affected. Must be set to the protocol in use to receive data correctly.

Protocol	Byte	1 for Serial (RS232/RS422/TTL/USB) or 0 for I2C.

1.4 Set Flow Control Mode	Dec	254 63	Mode	v8.0
	Hex	FE 3F	Mode	
	ASCII	■ ?	Mode	

Toggles flow control between hardware, software and off settings. Software and Hardware control can be further tuned using the settings above. Default is Off, or 0.

Mode	Byte	Flow control setting as below.

Table 17: Hardware Flow Control Trigger Levels

Bytes	1	4	8	14
Level	0	1	2	3

Table 18: Flow Control Settings

Flow Control	None	Software	Hardware
Mode	0	1	2

1.5 Set Hardware Flow Control Trigger Level	Dec	254 62	Level	v8.0
	Hex	FE 3E	Level	
	ASCII	■ >	Level	

Sets the hardware flow control trigger level. The Clear To Send signal will be deactivated once the number of characters in the display buffer reaches the level set; it will be reactivated once all data in the buffer is handled.

Level	Byte	Trigger level as above.

1.6 Turn Software Flow Control On	Dec	254 58	Almost Full	Almost Empty	v8.0
	Hex	FE 3A	Almost Full	Almost Empty	
	ASCII	■ :	Almost Full	Almost Empty	

Enables simple flow control. The display will return a single, Xoff, byte to the host when the display buffer is almost full and a different, Xon, byte when the buffer is almost empty. Full value should provide enough room for the largest data packet to be received without buffer overflow. No data should be sent to the display between full and empty responses to permit processing. Buffer size is 256* bytes. Not available in I^2C. Default off.

Almost Full	**Byte**	Number of bytes remaining before buffer is completely full, 0 < Full < Empty < 256*.
Almost Empty	**Byte**	Number of bytes before buffer can be considered empty enough to accept data.

***Note:** Buffer size was increased to 256 bytes from 128 bytes at firmware revision 8.3.

1.7 Turn Software Flow Control Off	Dec	254 59	v8.0
	Hex	FE 3B	
	ASCII	■ ;	

Disables flow control. Bytes sent to the display may be permitted to overflow the buffer resulting in data loss.

1.8 Set Software Flow Control Response	Dec	254 60	Xon	Xoff	v8.0
	Hex	FE 3C	Xon	Xoff	
	ASCII	■ <	Xon	Xoff	

Sets the values returned for almost full and almost empty messages when in flow control mode. This command permits the display to utilize standard flow control values of 0x11 and 0x13, note that defaults are 0xFF and 0xFE.

Xon	**Byte**	Value returned when display buffer is almost empty, permitting transmission to resume.
Xoff	**Byte**	Value returned when display buffer is almost full, signaling transmission to halt.

1.9 Echo	Dec	254 255	Length	Data	v8.3
	Hex	FE FF	Length	Data	
	ASCII	■	Length	Data	

Send data to the display that it will echo. Useful to confirm communication or return information from scripts.

Length	**Short**	Length of data array to be echoed.
Data	**Byte(s)**	An arbitrary array of data that the module will return.
Response	**Byte(s)**	The same arbitrary array of data originally sent.

1.10 Delay	Dec	254 251	Time	v8.3
	Hex	FE FB	Time	
	ASCII	■ √	Time	

Pause command execution to and responses from the display for the specified length of time.

Time	**Short**	Length of delay in ms, maximum 2000.

1.11 Software Reset	Dec	254 253 77 79 117 110	v8.4
	Hex	FE FD 4D 4F 75 6E	
	ASCII	■ ² M O u n	

Reset the display as if power had been cycled via a software command. No commands should be sent while the unit is in the process of resetting; a response will be returned to indicate the unit has successfully been reset.

Response	**Short**	Successful reset response, 254 212.

6.2 Text

2.1 Clear Screen				v8.0
	Dec	254 88		
	Hex	FE 58		
	ASCII	■ X		
Clears the contents of the screen.				

2.2 Go Home				v8.0
	Dec	254 72		
	Hex	FE 48		
	ASCII	■ H		
Returns the cursor to the top left of the screen.				

2.3 Set Cursor Position					v8.0
	Dec	254 71	Column	Row	
	Hex	FE 47	Column	Row	
	ASCII	■ G	Column	Row	
Sets the cursor to a specific cursor position where the next transmitted character is printed.					
Column	Byte	Value between 1 and number of character columns.			
Row	Byte	Value between 1 and number of character rows.			

2.4 Set Cursor Coordinate					v8.0
	Dec	254 121	X	Y	
	Hex	FE 79	X	Y	
	ASCII	■ y	X	Y	
Sets the cursor to an exact pixel position where the next transmitted character is printed.					
X	Byte	Value between 1 and screen width, represents leftmost character position.			
Y	Byte	Value between 1 and screen height, represents topmost character position.			

2.5 Initialize Text Window										v8.3
	Dec	254 43	ID	X1	Y1	X2	Y2	Font	CharSpace	LineSpace	Scroll	
	Hex	FE 2B	ID	X1	Y1	X2	Y2	Font	CharSpace	LineSpace	Scroll	
	ASCII	■ +	ID	X1	Y1	X2	Y2	Font	CharSpace	LineSpace	Scroll	
Designates a portion of the screen to which text can be confined. Font commands affect only the current window, default (entire screen) is window 0.												
ID	Byte	Unique text window identification number, value between 0 and 15.										
X1	Byte	Leftmost coordinate.										
Y1	Byte	Topmost coordinate.										
X2	Byte	Rightmost coordinate.										
Y2	Byte	Bottommost coordinate.										
Font*	Short	Unique font ID to use for this window, value between 0 and 1023.										
CharSpace	Byte	Spacing between characters to use for this window.										
LineSpace	Byte	Spacing between lines to use for this window.										
Scroll	Byte	Number of pixel rows to write to before scrolling text.										

***Note:** Font was changed from a Byte length at firmware revision 8.5

2.6 Set Text Window	Dec	254 42	ID	v8.3
	Hex	FE 2A	ID	
	ASCII	■ *	ID	

Sets the text window to which subsequent text and commands will apply. Default (entire screen) is window 0.

ID	**Byte**	Unique text window to use.

2.7 Clear Text Window	Dec	254 44	ID	v8.3
	Hex	FE 2C	ID	
	ASCII	■ ,	ID	

Clear the contents of a specific text window, similar to the clear screen command.

ID	**Byte**	Unique text window to clear.

2.8 Initialize Label	Dec	254 45	ID X1 Y1 X2 Y2 Vert Hor Font Background CharSpace	v8.3
	Hex	FE 2D	ID X1 Y1 X2 Y2 Vert Hor Font Background CharSpace	
	ASCII	■ -	ID X1 Y1 X2 Y2 Vert Hor Font Background CharSpace	

Designates a portion of the screen that can be easily updated with one line of text, often used to display variables.

ID	**Byte**	Unique label identification number, value between 0 and 15.
X1	**Byte**	Leftmost coordinate.
Y1	**Byte**	Topmost coordinate.
X2	**Byte**	Rightmost coordinate.
Y2	**Byte**	Bottommost coordinate.
Vert	**Byte**	Vertical justification of the label text; 0 for top, 1 for middle, or 2 for bottom.
Hor	**Byte**	Horizontal justification of the label text; 0 for left, 1 for centre, or 2 for right.
Font*	**Short**	Unique font ID to use for this label, value between 0 and 1023.
Background	**Byte**	State of the pixels in the label region that is not occupied by text; 0 for off or 1 for on.
CharSpace	**Byte**	Spacing between characters to use for this label.

***Note:** Font was changed from a Byte length at firmware revision 8.5

2.9 Update Label	Dec	254 46	ID Data	v8.3
	Hex	FE 2E	ID Data	
	ASCII	■ .	ID Data	

Update a previously created label with new text. Send a null character (empty string) to clear a label.

ID	**Byte**	Unique label to update, between 0 and 15.
Data	**String**	Information to display in the label, must be terminated with a null (value of zero) byte.

2.10 Auto Scroll On	Dec	254 81	v8.0
	Hex	FE 51	
	ASCII	■ Q	

The entire contents of screen are shifted up one line when the end of the screen is reached. Display default is on.

2.11 Auto Scroll Off	Dec	254 82	v8.0
	Hex	FE 52	
	ASCII	■ R	

New text is written over the top line when the end of the screen is reached. Display default is Auto Scroll on.

6.3 Drawing

3.1 Set Drawing Colour				v8.0
	Dec	254 99	Colour	
	Hex	FE 63	Colour	
	ASCII	■ c	Colour	

Set the colour to be used for all future drawing commands that do not implicitly specify colour.

Colour	**Byte**	0 for background or any other value for text colour.

3.2 Draw Pixel				v8.0
	Dec	254 112	X Y	
	Hex	FE 70	X Y	
	ASCII	■ p	X Y	

Draw a single pixel at the specified coordinate using the current drawing colour.

X	**Byte**	Horizontal position of pixel to be drawn.
Y	**Byte**	Vertical position of pixel to be drawn.

3.3 Draw a Line				v8.0
	Dec	254 108	X1 Y1 X2 Y2	
	Hex	FE 6C	X1 Y1 X2 Y2	
	ASCII	■ l	X1 Y1 X2 Y2	

Draw a line connecting two termini. Lines may be rendered differently when drawn right to left versus left to right.

X1	**Byte**	Horizontal coordinate of first terminus.
Y1	**Byte**	Vertical coordinate of first terminus.
X2	**Byte**	Horizontal coordinate of second terminus.
Y2	**Byte**	Vertical coordinate of second terminus.

3.4 Continue a Line				v8.0
	Dec	254 101	X Y	
	Hex	FE 65	X Y	
	ASCII	■ e	X Y	

Draw a line from the last point drawn to the coordinate specified using the current drawing colour.

X	**Byte**	Left coordinate of terminus.
Y	**Byte**	Top coordinate of terminus.

3.5 Draw a Rectangle				v8.0
	Dec	254 114	Colour X1 Y1 X2 Y2	
	Hex	FE 72	Colour X1 Y1 X2 Y2	
	ASCII	■ r	Colour X1 Y1 X2 Y2	

Draw a rectangular frame one pixel wide using the colour specified; current drawing colour is ignored.

Colour	**Byte**	0 for background or any other value for text colour.
X1	**Byte**	Leftmost coordinate.
Y1	**Byte**	Topmost coordinate.
X2	**Byte**	Rightmost coordinate.
Y2	**Byte**	Bottommost coordinate.

3.6 Draw a Filled Rectangle	Dec	254 120	Colour X1 Y1 X2 Y2	v8.0
	Hex	FE 78	Colour X1 Y1 X2 Y2	
	ASCII	■ x	Colour X1 Y1 X2 Y2	

Draw a filled rectangle using the colour specified; current drawing colour is ignored.

Colour	Byte	0 for background or any other value for text colour.
X1	Byte	Leftmost coordinate.
Y1	Byte	Topmost coordinate.
X2	Byte	Rightmost coordinate.
Y2	Byte	Bottommost coordinate.

3.7 Draw a Rounded Rectangle	Dec	254 128	X1 Y1 X2 Y2 Radius	v8.3
	Hex	FE 80	X1 Y1 X2 Y2 Radius	
	ASCII	■ Ç	X1 Y1 X2 Y2 Radius	

Draw a rounded rectangular frame one pixel wide using the current drawing colour.

X1	Byte	Leftmost coordinate of the rectangle.
Y1	Byte	Topmost coordinate of the rectangle.
X2	Byte	Rightmost coordinate.
Y2	Byte	Bottommost coordinate.
Radius	Byte	Radius of curvature of the rectangle corners.

3.8 Draw a Filled Rounded Rectangle	Dec	254 129	X1 Y1 X2 Y2 Radius	v8.3
	Hex	FE 81	X1 Y1 X2 Y2 Radius	
	ASCII	■ ü	X1 Y1 X2 Y2 Radius	

Draw a filled rounded rectangle using the current drawing colour.

X1	Byte	Leftmost coordinate of the rectangle.
Y1	Byte	Topmost coordinate of the rectangle.
X2	Byte	Rightmost coordinate.
Y2	Byte	Bottommost coordinate.
Radius	Byte	Radius of curvature of the rectangle corners.

3.9 Draw a Circle	Dec	254 123	X Y Radius	v8.3
	Hex	FE 7B	X Y Radius	
	ASCII	■ {	X Y Radius	

Draw a circular frame one pixel wide using the current drawing colour.

X	Byte	Horizontal coordinate of the circle centre.
Y	Byte	Vertical coordinate of the circle centre.
Radius	Byte	Distance between the circle perimeter and centre.

3.10 Draw a Filled Circle	Dec	254 124	X Y Radius	v8.3
	Hex	FE 7C	X Y Radius	
	ASCII	■ |	X Y Radius	

Draw a filled circle using the current drawing colour.

X	**Byte**	Horizontal coordinate of the circle centre.
Y	**Byte**	Vertical coordinate of the circle centre.
Radius	**Byte**	Distance between the circle perimeter and centre.

3.11 Draw an Ellipse	Dec	254 125	X Y XRadius YRadius	v8.3
	Hex	FE 7D	X Y XRadius YRadius	
	ASCII	■ }	X Y XRadius YRadius	

Draw an elliptical frame one pixel wide using the current drawing colour.

X	**Byte**	Horizontal coordinate of the ellipse centre, zero indexed from left.
Y	**Byte**	Vertical coordinate of the ellipse centre, zero indexed from top.
XRadius	**Byte**	Distance between the furthest horizontal point on the ellipse perimeter and centre.
YRadius	**Byte**	Distance between the furthest vertical point on the ellipse perimeter and centre.

3.12 Draw a Filled Ellipse	Dec	254 127	X Y XRadius YRadius	v8.3
	Hex	FE 7F	X Y XRadius YRadius	
	ASCII	■ DEL	X Y XRadius YRadius	

Draw an ellipse using the current drawing colour.

X	**Byte**	Horizontal coordinate of the ellipse centre, zero indexed from left.
Y	**Byte**	Vertical coordinate of the ellipse centre, zero indexed from top.
XRadius	**Byte**	Distance between the furthest horizontal point on the ellipse perimeter and centre.
YRadius	**Byte**	Distance between the furthest vertical point on the ellipse perimeter and centre.

3.13 Scroll Screen	Dec	254 89	X1 Y1 X2 Y2 MoveX MoveY	v8.3
	Hex	FE 59	X1 Y1 X2 Y2 MoveX MoveY	
	ASCII	■ Y	X1 Y1 X2 Y2 MoveX MoveY	

Define and scroll the contents of a portion of the screen.

X1	**Byte**	Leftmost coordinate of the scroll window, zero indexed from left.
Y1	**Byte**	Topmost coordinate of the scroll window, zero indexed from top.
X2	**Byte**	Rightmost coordinate of the scroll window, zero indexed from left.
Y2	**Byte**	Bottommost coordinate of the scroll window, zero indexed from top.
MoveX	**Signed Short**	Number of pixels to scroll horizontally.
MoveY	**Signed Short**	Number of pixels to scroll vertically.

3.14 Initialize a Bar Graph	Dec	254 103	ID Type X1 Y1 X2 Y2	v8.3
	Hex	FE 67	ID Type X1 Y1 X2 Y2	
	ASCII	■ g	ID Type X1 Y1 X2 Y2	

Initialize a bar graph in memory for later implementation. Graphs can be located anywhere on the screen, but overlapping may cause distortion. Graph should be filled using the Draw a Bar Graph command.

ID	**Byte**	Unique bar identification number, between 0 and 255.
Type	**Byte**	Graph style, see Bar Graph Types.
X1	**Byte**	Leftmost coordinate.
Y1	**Byte**	Topmost coordinate.
X2	**Byte**	Rightmost coordinate.
Y2	**Byte**	Bottommost coordinate.

Table 19: Bar Graph Types

	Direction	Base
0	Vertical	Bottom
1	Horizontal	Left
2	Vertical	Top
3	Horizontal	Right

3.15 Initialize 9-Slice Bar Graph	Dec	254 115	ID Type X1 Y1 X2 Y2 Fore 9Slice Back 9Slice	v8.3
	Hex	FE 73	ID Type X1 Y1 X2 Y2 Fore 9Slice Back 9Slice	
	ASCII	■ s	ID Type X1 Y1 X2 Y2 Fore 9Slice Back 9Slice	

Initialize a 9-slice bar graph in memory for later implementation. 9-slice graphs are also be filled using the Draw a Bar Graph command and are allocated to the same memory as regular bitmaps.

ID	**Byte**	Unique bar identification number, value between 0 and 255.
Type	**Byte**	Graph style, see Bar Graph Types.
X1	**Byte**	Leftmost coordinate of the 9-slice bar, zero indexed from left.
Y1	**Byte**	Topmost coordinate of the 9-slice bar, zero indexed from top.
X2	**Byte**	Rightmost coordinate of the 9-slice bar, zero indexed from left.
Y2	**Byte**	Bottommost coordinate of the 9-slice bar, zero indexed from top.
Fore 9Slice	**Short**	9-slice used for the foreground.
Back 9Slice	**Short**	9-slice used for the background.

3.16 Draw a Bar Graph	Dec	254 105	ID Value	v8.3
	Hex	FE 69	ID Value	
	ASCII	■ i	ID Value	

Fill in a portion of a bar graph after initialization. Any old value will be overwritten by the new. Setting a value of zero before setting a new value will restore a graph should it become corrupted.

ID	**Byte**	Unique bar identification number, between 0 and 255.
Value	**Byte**	Portion of graph to fill in pixels, will not exceed display bounds.

3.17 Initialize a Strip Chart	Dec	254 110	ID X1 Y1 X2 Y2 Min Max Step Style ID	v8.3
	Hex	FE 6E	ID X1 Y1 X2 Y2 Min Max Step Style ID	
	ASCII	■ n	ID X1 Y1 X2 Y2 Min Max Step Style ID	

Designate a portion of the screen for a chart. Visual changes will occur when the update command is issued.

ID	**Byte**	Unique chart identification number, value between 0 and 7.
X1	**Byte**	Leftmost coordinate of the strip chart, zero indexed from left.
Y1	**Byte**	Topmost coordinate of the strip chart, zero indexed from top.
X2	**Byte**	Rightmost coordinate of the strip chart, zero indexed from left.
Y2	**Byte**	Bottommost coordinate of the strip chart, zero indexed from top.
Min	**Short**	Minimum chart value.
Max	**Short**	Maximum chart value. For line styles, make max-min at least one pixel less than chart height.
Step	**Byte**	Scroll distance between updates, in pixels.
Style	**Byte**	Chart style value which is an OR'd combination of type and direction, as per the tables below.
ID	**Short**	9-slice file ID, if a 9-slice style strip chart is not desired send any value for this parameter.

Table 20: Strip Chart Directions (Bytes 7-4)

Direction	Description
0	Bottom origin, left shift
32	Left origin, upward shift
64	Top origin, right shift
96	Right origin, downward shift
128	Bottom origin, right shift
160	Left origin, downward shift
192	Top origin, left shift
224	Right origin, upward shift

Table 21: Strip Chart Types (Bytes 3-0)

Type	Description
0	Bar
1	Line
2	Step
3	Box
4	9-slice
5	Separated Bar
6	Separated Box

3.18 Update a Strip Chart	Dec	254 111	ID Value	v8.3
	Hex	FE 6F	ID Value	
	ASCII	■ o	ID Value	

Shift the specified strip chart and draw a new value.

ID	**Byte**	Chart identification number, value between 0 and 7.
Value	**Short**	Value to add to the chart.

6.4 Fonts

4.1 Upload a Font File				v8.0
	Dec	254 36	ID Size Data	
	Hex	FE 24	ID Size Data	
	ASCII	■ $	ID Size Data	

Upload a font to a graphic display. To create a font see the Font File Creation section, for upload protocol see the File Transfer Protocol or XModem Transfer Protocol entries. Default font is ID 1.

ID*	**Short**	Unique font identification number, value between 0 and 1023.
Size*	**Integer**	Size of the entire font file.
Data	**Byte(s)**	Font file data, see the Font File Creation example.

***Note:** ID and Size were changed from Byte and Short lengths respectively at firmware revision 8.1

4.2 Set the Current Font				v8.0
	Dec	254 49	ID	
	Hex	FE 31	ID	
	ASCII	■ 1	ID	

Set the font in use by specifying a unique identification number. Characters sent after the command will appear in the font specified; previous text will not be affected. Default is 1.

ID*	**Short**	Unique font identification number, value between 0 and 1023.

***Note:** ID was changed from a Byte length at firmware revision 8.5

4.3 Set Font Metrics				v8.0
	Dec	254 50	LineMargin TopMargin CharSpace LineSpace Scroll	
	Hex	FE 32	LineMargin TopMargin CharSpace LineSpace Scroll	
	ASCII	■ 2	LineMargin TopMargin CharSpace LineSpace Scroll	

Set the font spacing, or metrics, used with the current font. Changes only appear in text sent after command.

LineMargin	**Byte**	Space between left of display and first column of text. Default 0.
TopMargin	**Byte**	Space between top of display area and first row of text. Default 0.
CharSpace	**Byte**	Space between characters. Default 0.
Line Space	**Byte**	Space between character rows. Default 1.
Scroll	**Byte**	Point at which text scrolls up screen to display additional rows. Default 1.

4.4 Set Box Space Mode				v8.0
	Dec	254 172	Switch	
	Hex	FE AC	Switch	
	ASCII	■ ¼	Switch	

Toggle box space on or off. When on, a character sized box is cleared from the screen before a character is written. This eliminates any text or bitmap remnants behind the character. Default is on.

Switch	**Byte**	1 for on or 0 for off.

Font File Creation

Matrix Orbital graphic displays are capable of displaying text in a wide variety of styles customizable to suit any project design. Front files alter the style of text and appearance of the display.

By default, a Matrix Orbital graphic display is loaded with a small filled font in slot one and a future bk bt 16 style in slot two. Both are available at www.matrixorbital.ca/software/graphic_fonts.

The easiest way to create, add, or modify the fonts of any graphic display is through the MOGD# tool. This provides a simple graphic interface that hides the more complex intricacies of the font file.

Table 22: Example Font File Header

Maximum Width	Character Height	ASCII Start Value	ASCII End Value
5	7	104	106

The font file header contains four bytes: First, the number of columns in the widest character; usually 'w', second, the pixel height of each character, and finally, the start and end values of the character range. The range represents the values that must be sent to the display to trigger the characters to appear on the screen. In the example, the decimal values corresponding to the lowercase letters 'h' through 'j' will be used resulting in the range shown.

Table 23: Example Character Table

	MSB	LSB	Width
h	0	13	5
i	0	18	3
j	0	21	4

The character table contains information that allows the display to locate each individual character in a mass of character data. Each character has three bytes; two indicating it's offset in the character data and one indicating its width. The offset takes into account the header and table bytes to point to the first byte of the character data it references. The first byte of the file, maximum width, has an offset of zero. The width byte of each character can be identical as in a fixed width font, or in our case, variable. The character table will become clearer after analyzing the final part of the font file, character data.

Table 24: Character 'h' Bitmap

1	0	0	0	0
1	0	0	0	0
1	0	1	1	0
1	1	0	0	1
1	0	0	0	1
1	0	0	0	1
1	0	0	0	1

Table 25: Character 'h' Data

1	0	0	0	0	1	0	0	84	**132**
0	0	1	0	1	1	0	1	2D	**45**
1	0	0	1	1	0	0	0	98	**152**
1	1	0	0	0	1	1	0	C6	**198**
0	0	1	0	0	0	0	0	20	**32**

The character data is a binary graphical representation of each glyph in a font. Each character is drawn on a grid containing as many rows as the height specified in the header and as many columns as the width specified in the character table. Cells are drawn by writing a one in their location and cleared by setting a value of zero. Starting at the top left, moving right, then down, eight of these cells form a character data byte. When all cells are accounted for, zeroes may be added to the last byte to complete it. A sample of an 'h' glyph is shown above. The data for the 'i' and 'j' characters will follow to complete the custom font file displayed below.

Table 26: Example Font File

Header	5 7 104 106
Character Table	0 13 5
	0 18 3
	0 21 4
Character Data	132 45 152 198 32
	67 36 184
	16 49 25 96

6.5 Bitmaps

5.1 Upload a Bitmap File	Dec	254 94	ID Size Data	v8.0
	Hex	FE 5E	ID Size Data	
	ASCII	■ ^	ID Size Data	

Upload a bitmap to a graphic display. To create a bitmap see the Bitmap File Creation section, for upload protocol see the File Transfer Protocol or XModem Transfer Protocol entries. Start screen is ID 1.

ID*	**Short**	Unique bitmap identification number, value between 0 and 1023.
Size*	**Integer**	Size of the entire bitmap file.
Data	**Byte(s)**	Bitmap file data, see the Bitmap File Creation example.

***Note:** ID and Size were changed from Byte and Short lengths respectively at firmware revision 8.1

5.2 Upload a Bitmap Mask	Dec	254 92 5	ID Size Data	v8.3
	Hex	FE 5C 05	ID Size Data	
	ASCII	■ \ ENQ	ID Size Data	

Upload a bitmap mask that can clear areas of the screen before a bitmap is drawn. Programmatically, (bitmap&mask) | (screen&~mask) is shown when a bitmap is drawn. To create a mask see the Bitmap File Creation section, for upload protocol see the File Transfer Protocol or XModem Transfer Protocol entries.

ID	**Short**	Unique bitmap mask identification number, value between 0 and 1023.
Size	**Integer**	Size of the entire mask file.
Data	**Byte(s)**	Bitmap mask file data, see the Bitmap File Creation example.

5.3 Draw a Bitmap from Memory	Dec	254 98	ID X Y	v8.0
	Hex	FE 62	ID X Y	
	ASCII	■ b	ID X Y	

Draw a previously uploaded bitmap from memory. Top left corner must be specified for drawing.

ID*	**Short**	Unique bitmap identification number, value between 0 and 1023.
X	**Byte**	Leftmost coordinate of bitmap.
Y	**Byte**	Topmost coordinate of bitmap.

***Note:** ID and Size were changed from Byte and Short lengths respectively at firmware revision 8.1

5.4 Draw a Partial Bitmap	Dec	254 192	ID X1 Y1 X2 Y2	v8.5
	Hex	FE C0	ID X1 Y1 X2 Y2	
	ASCII	■ └	ID X1 Y1 X2 Y2	

Draw a portion of a previously uploaded bitmap confined to the width and height specified.

ID	**Short**	Unique bitmap identification number, value between 0 and 1023.
X1	**Byte**	Leftmost coordinate of bitmap.
Y1	**Byte**	Topmost coordinate of bitmap.
X2	**Byte**	Rightmost coordinate of the partial bitmap.
Y2	**Byte**	Bottommost coordinate of the partial bitmap.

5.5 Draw a Bitmap Directly	Dec	254 100	X1	Y1	Data	v8.0
	Hex	FE 64	X1	Y1	Data	
	ASCII	■ d	X1	Y1	Data	

Draw a bitmap directly to the graphic display without saving to memory. Cannot be implemented in a script.

X1	**Byte**	Leftmost coordinate of bitmap.
Y1	**Byte**	Topmost coordinate of bitmap.
Data	**Byte(s)**	Bitmap file data, see the Bitmap File Creation example.

Bitmap File Creation

In addition to fonts, Matrix Orbital graphic displays can also hold a number of customizable bitmaps to provide further stylistic product integration. Like font files, bitmaps files are most easily uploaded to a display using MOGD#. However, the critical data component of the bitmap upload command is detailed below for reference.

The bitmap data block is similar to that of a font. However, as a bitmap is a single glyph, only a simple two byte header is required. First, one byte representing the bitmap width is sent, then one byte for the height. Each bitmap is merely encoded in binary fashion using a series of ones and zeroes. Again a grid can be created using the width and height specified in the upload command, populated in the manner above, and converted into byte values. A smiley face example is shown below to indicate the ultimate affect of the Matrix Orbital graphic stylization ability.

Table 27: Smiley Face Bitmap

0	**1**	0	**1**	0
0	0	0	0	0
1	0	0	0	**1**
0	**1**	**1**	**1**	0

Table 28:Smiley Face Data

0	**1**	0	**1**	0	0	0	0	50	80
0	0	**1**	0	0	0	**1**	0	22	34
1	**1**	**1**	0	0	0	0	0	E0	224

Table 29: Example Bitmap File

Header	5 4
Bitmap Data	80 34 224

Bitmap Masking

Like a regular bitmap, a mask can be loaded to the display and used to create a more polished result when drawing in populated areas. When defining a mask, all active values will clear any background information, while any inactive values will leave it untouched. This is best described with an example.

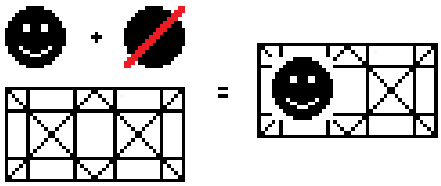


Figure 17: Drawing without a Mask

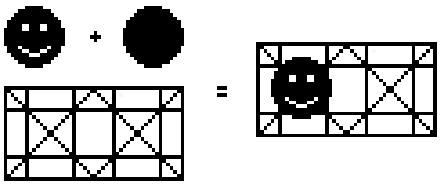


Figure 18: Drawing with a Mask

6.6 9-Slices

6.1 Upload a 9-Slice File	Dec	254 92 3	ID Size Data	v8.3
	Hex	FE 5C 03	ID Size Data	
	ASCII	■ \ ETX	ID Size Data	

Upload a 9-slice file to a graphic display. To create a 9-slice see the 9-Slice File Creation section, for upload protocol see the
File Transfer Protocol or XModem Transfer Protocol entries.

ID	**Short**	Unique 9-slice identification number, value between 0 and 1023.
Size	**Integer**	Size of the 9-slice file.
Data	**Byte(s)**	9-slice file data, see the 9-Slice File Creation example.

6.2 Upload a 9-Slice Mask	Dec	254 92 6	ID Size Data	v8.3
	Hex	FE 5C 06	ID Size Data	
	ASCII	■ \ ACK	ID Size Data	

Upload a 9-slice mask that can clear areas of the screen before a 9-slice is drawn. Programmatically, (9slice&mask) | (screen&~mask) is shown when a bitmap is drawn. To create a mask see the 9-Slice File Creation section, for upload protocol see the
File Transfer Protocol or XModem Transfer Protocol entries.

ID	**Short**	Unique 9-slice mask identification number, value between 0 and 1023.
Size	**Integer**	Size of the entire mask file.
Data	**Byte(s)**	9-slice mask file data, see the 9-Slice File Creation example.

6.3 Display a 9-Slice	Dec	254 91	ID X1 Y1 X2 Y2	v8.3
	Hex	FE 5B	ID X1 Y1 X2 Y2	
	ASCII	■ [	ID X1 Y1 X2 Y2	

Displays a previously loaded 9-slice at the specified location.

ID	**Short**	Unique 9-slice identification number, value between 0 and 1023.
X1	**Byte**	Leftmost coordinate of the 9-slice.
Y1	**Byte**	Topmost coordinate of the 9-slice.
X2	**Byte**	Rightmost coordinate of the 9-slice.
Y2	**Byte**	Bottommost coordinate of the 9-slice.

9-Slice File Creation

A 9-slice file is a scalable graphic composed of nine different bitmap sections as shown below.

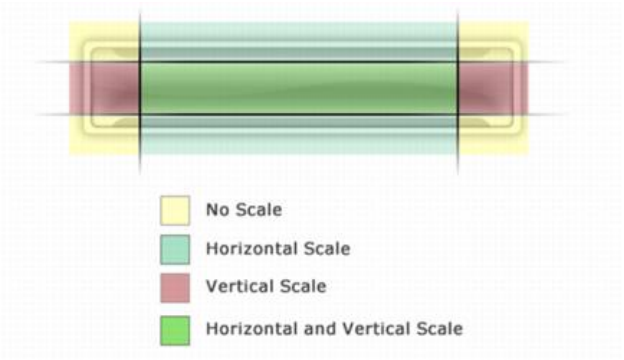


Figure 19: Adobe 9-slice Representation

The 9-slice file format requires that the bitmap dimensions and the locations of divisions be defined before a graphic is uploaded normally as shown in the Bitmap File Creation example.

Table 30: 9-slice file format

Width	One byte representing the width of the entire bitmap.
Height	One byte representing the height of the entire bitmap.
Top	One byte specifying the height of the top row section of the 9-slice.
Bottom	One byte specifying the height of the bottom row section of the 9-slice.
Left	One byte specifying the width of the left column section of the 9-slice.
Right	One byte specifying the width of the right column section of the 9-slice.
Bitmap Data	Data outlining the entire bitmap, as per the Bitmap File Creation example.

6.7 Animations

7.1 Upload an Animation File				v8.3
	Dec	254 92 4	File ID Size Data	
	Hex	FE 5C 04	File ID Size Data	
	ASCII	■ \ EOT	File ID Size Data	
Upload an animation file to a graphic display. To create an animation see the Animation File Creation section, for upload protocol see the File Transfer Protocol or XModem Transfer Protocol entries. Up to 16 animations can be displayed on the screen at one time, using the Display Animation command, but up to 1024 can be stored in memory for later use. Please note the total graphic memory size is 256KB.				
File ID	**Short**	Unique animation file identification number, value between 0 and 1023.		
Size	**Integer**	Size of the animation file.		
Data	**Byte(s)**	Animation file data, see the Animation File Creation example.		

7.2 Display Animation				v8.3
	Dec	254 193	ID File ID* X Y	
	Hex	FE C1	ID File ID* X Y	
	ASCII	■ ⊥	ID File ID* X Y	
Load the first frame of the specified animation in its stopped state at the specified location. If an animation is already in use at that index it will be overwritten. Use the start animation command to play the displayed file.				
ID	**Byte**	Unique animation identification number, value between 0 and 15.		
File ID	**Short**	Unique animation file identification number, value between 0 and 1023.		
X	**Byte**	Leftmost coordinate of animation.		
Y	**Byte**	Topmost coordinate of animation.		

***Note:** File ID short length variable was removed from this command at v8.4, and reintroduced at v8.5.

7.3 Delete Animation				v8.3
	Dec	254 199	ID	
	Hex	FE C7	ID	
	ASCII	■ ╟	ID	
Stop and delete the displayed animation specified.				
ID	**Byte**	Animation number to delete, value between 0 and 15.		

7.4 Start/Stop Animation				v8.3
	Dec	254 194	ID Start	
	Hex	FE C2	ID Start	
	ASCII	■ ┬	ID Start	
Start or stop an animation that has been displayed.				
ID	**Byte**	Animation number to start/stop, value between 0 and 15.		
Start	**Byte**	Any non-zero value will start the specified animation, 0 will stop it.		

7.5 Set Animation Frame				v8.3
	Dec	254 197	ID Frame	
	Hex	FE C5	ID Frame	
	ASCII	■ ┼	ID Frame	
Set the current frame of a displayed animation. If the frame exceeds the total number present, the animation will be set to the first frame.				
ID	**Byte**	Animation number to control, value between 0 and 15.		
Frame	**Byte**	Number of the frame to be displayed, value between 0 and 31.		

7.6 Get Animation Frame	Dec	254 196	ID	v8.3
	Hex	FE C4	ID	
	ASCII	■ –	ID	
Get the current frame of a displayed animation.				
ID	**Byte**	Animation number to request frame number, value between 0 and 15.		
Response	**Byte**	Current frame number of the animation specified, value between 0 and 31.		

Animation File Creation

An animation file is a series of bitmaps, each displayed for a specified length of time within a continuous rotation. The file begins by specifying the number of frames, the offset of each block of bitmap information, and the time to display each frame. After which bitmap headers and data are transmitted for each frame, in the same manner as the Bitmap File Creation example.

Table 31: Animation file format

Total Frames	One byte representing the total number of frames in the animation
Offsets	One entry for each frame, 4 bytes indicating the start of the bitmap file. Maximum 32 frames.
Times	Two bytes for each frame representing the length of time (100ms) for which it is displayed.
Header 1	Two bytes, one representing the width and one the height of the first bitmap.
Bitmap 1 Data	The first bitmap data, as per the Bitmap File Creation example.
…	…
Header 9	Two bytes, one representing the width and one the height of the last bitmap.
Bitmap 9 Data	The last bitmap data, as per the Bitmap File Creation example.

6.8 General Purpose Output

8.1 General Purpose Output On				v8.0
	Dec	254 87	Number	
	Hex	FE 57	Number	
	ASCII	■ W	Number	

Turns the specified GPO on, sourcing current from an output of five volts.

Number	**Byte**	GPO to be turned on.

8.2 General Purpose Output Off				v8.0
	Dec	254 86	Number	
	Hex	FE 56	Number	
	ASCII	■ V	Number	

Turns the specified GPO off, sinking current to an output of zero volts.

Number	**Byte**	GPO to be turned off.

8.3 Set Start Up GPO State					v8.0
	Dec	254 195	Number	State	
	Hex	FE C3	Number	State	
	ASCII	■ ├	Number	State	

Sets and saves the start up state of the specified GPO in non volatile memory. Changes will be seen on start up.

Number	**Byte**	GPO to be controlled.
State	**Byte**	1 for on or 0 for off.

6.9 Piezo Buzzer

9.1 Activate Piezo Buzzer*	Dec	254 187	Frequency Time	v8.0
	Hex	FE BB	Frequency Time	
	ASCII	■ ╗	Frequency Time	

Activates a buzz of specific frequency from the onboard piezo buzzer for a specified length of time.

Frequency	**Short**	Frequency of buzz in hertz.
Time	**Short**	***Duration of the beep in milliseconds.

***Note:** When a beep precedes a delay command, the duration of the beep must be shorter than that of the delay.

9.2 Set Default Buzzer Beep	Dec	254 188	Frequency Duration	v8.3
	Hex	FE BC	Frequency Duration	
	ASCII	■ ╝	Frequency Duration	

Set the frequency and duration of the default beep transmitted when the bell character is transmitted.

Frequency	**Short**	Frequency of the beep in Hertz, default 440Hz.
Duration	**Short**	Duration of the beep in milliseconds, default 100ms.

9.3 Set Keypad Buzzer Beep	Dec	254 182	Frequency Duration	v8.4
	Hex	FE B6	Frequency Duration	
	ASCII	■ ╢	Frequency Duration	

Set the frequency and duration of the default beep transmitted when a key is pressed.

Frequency	**Short**	Frequency of the beep in Hertz, default is 0 or off.
Duration	**Short**	Duration of the beep in milliseconds, default is 0 or off.

9.4 Set Touch Buzzer Beep	Dec	254 182	Down Freq Up Freq	v8.4
	Hex	FE B6	Down Freq Up Freq	
	ASCII	■ ╢	Down Freq Up Freq	

Set the frequency of the default beep transmitted when a touch event occurs. Duration of each is 50ms.

Down Freq	**Short**	Frequency of the down event beep in Hertz, default is 0 or off.
Up Freq	**Short**	Frequency of the up event beep in Hertz, default is 0 or off.

6.10 Keypad

10.1 Auto Transmit Key Presses On				v8.0
	Dec	254 65		
	Hex	FE 41		
	ASCII	■ A		

Key presses are automatically sent to the host when received by the display. Use this mode for I2C transactions.

10.2 Auto Transmit Key Presses Off				v8.0
	Dec	254 79		
	Hex	FE 4F		
	ASCII	■ O		

Key presses are held in the 10 key buffer to be polled by the host using the Poll Key Press command. Default is Auto Transmit on.

10.3 Poll Key Press				v8.0
	Dec	254 38		
	Hex	FE 26		
	ASCII	■ &		

Reads the last unread key press from the 10 key display buffer. If another key is stored in the buffer the MSb will be 1, the MSb will be 0 when the last key press is read. If there are no stored key presses a value of 0 will be returned. Auto transmit key presses must be turned off for this command to be successful, do not use with I^2C.

Response	Byte	Value of key pressed (MSb determines additional keys to be read).

10.4 Clear Key Buffer				v8.0
	Dec	254 69		
	Hex	FE 45		
	ASCII	■ E		

Clears all key presses from the key buffer.

10.5 Set Debounce Time				v8.0
	Dec	254 85	Time	
	Hex	FE 55	Time	
	ASCII	■ U	Time	

Sets the time between a key press and a key read by the display. Most switches will bounce when pressed; the debounce time allows the switch to settle for an accurate read. Default is 8 representing approximately 52ms.

Time	Byte	Debounce increment (debounce time = Time * 6.554ms).

10.6 Set Auto Repeat Mode				v8.0
	Dec	254 126	Mode	
	Hex	FE 7E	Mode	
	ASCII	■ DEL	Mode	

Sets key press repeat mode to typematic or hold. In typematic mode if a key press is held, by default the key value is transmitted immediately, then 5 times a second after a 1 second delay. In hold mode, the key down value is transmitted once when pressed, and then the key up value is sent when the key is released. Default is typematic.

Mode	Byte	1 for hold mode or 0 for typematic.

10.7 Auto Repeat Mode Off	Dec	254 96		v8.0
	Hex	FE 60		
	ASCII	■ `		
Turns auto repeat mode off. Default is on (typematic).				

10.8 Assign Keypad Codes	Dec	254 213	Key Down Key Up	v8.0
	Hex	FE D5	Key Down Key Up	
	ASCII	■ Õ	Key Down Key Up	
Assigns the key down and key up values sent to the host when a key press is detected. A key up and key down value must be sent for every key, a value of 255 will leave the key unaltered. Defaults are shown below.				
Key Down	Bytes [25]	Key down values.		
Key Up	Bytes [25]	Key up values.		

Table 32: Default Key Down Values

Key Down				
A(65)	B(66)	C(67)	D(68)	E(69)
F(70)	G(71)	H(72)	I(73)	J(74)
K(75)	L(76)	M(77)	N(78)	O(79)
P(80)	Q(81)	R(82)	S(83)	T(84)
U(85)	V(86)	W(87)	X(88)	Y(89)

Table 33: Default Key Up Values

Key Up				
a(97)	b(98)	c(99)	d(100)	e(101)
f(102)	g(103)	h(104)	i(105)	j(106)
k(107)	l(108)	m(109)	n(110)	o(111)
p(112)	q(113)	r(114)	s(115)	t(116)
u(117)	v(118)	w(119)	x(120)	y(121)

10.9 Set Typematic Delay	Dec	254 159	Delay	v8.4
	Hex	FE 9F	Delay	
	ASCII	■ ƒ	Delay	
Sets the delay between the first key press and first typematic report when a key is held in typematic mode.				
Delay	Byte	Time key must be held to trigger typematic reports, specified in 100ms, default is 10 (1s).		

10.10 Set Typematic Interval	Dec	254 158	Interval	v8.4
	Hex	FE 9E	Interval	
	ASCII	■ Þ	Interval	
Sets the interval between reported key presses when a key is held and the display is in typematic mode.				
Interval	Byte	Time between key reports, specified in 100ms increments, default is 2 (200ms).		

6.11 Touchpad

11.1 Set Touch Mode				v8.0
	Dec	254 135	Mode	
	Hex	FE 87	Mode	
	ASCII	■ ç	Mode	
Sets the method used to return touch events. Region mode will return a single value for events in defined areas. Coordinate mode will return event, x position, and y position bytes for each press, drag, or release.				
Mode	Byte	Touch reporting mode, 0 for region or 1 for coordinate mode. Default is coordinate.		

11.2 Set Region Reporting Mode				v8.0
	Dec	254 136	Mode	
	Hex	FE 88	Mode	
	ASCII	■ ê	Mode	
Defines the events transmitted in region mode. Allows only events specified to return a value to the host. Key down values are transmitted for press and drag events, key up for release, and the value 255 for out of region.				
Mode	Byte	Defines the events reported, see Region Reporting Mode. Default reporting returns all events.		

Table 34: Region Reporting Mode

Bit	7-4	3	2	1	0
Event	Reserved	Out of Region	Drag	Release	Press

11.3 Set Touch Region				v8.0
	Dec	254 132	ID X Y Width Height Key Down Key Up	
	Hex	FE 84	ID X Y Width Height Key Down Key Up	
	ASCII	■ ä	ID X Y Width Height Key Down Key Up	
Creates a region of the screen that responds when pressed and released with a defined single byte.				
ID	Byte	Unique region identification number, maximum 32 regions. Value between 0 and 31.		
X	Byte	Leftmost coordinate.		
Y	Byte	Topmost coordinate.		
Width	Byte	Width of region, must be within screen bounds.		
Height	Byte	Height of region, must be within screen bounds.		
Key Down	Byte	Value returned when region is pressed.		
Key Up	Byte	Value returned when region is released.		

11.4 Delete a Touch Region				v8.0
	Dec	254 133	ID	
	Hex	FE 85	ID	
	ASCII	■ à	ID	
Deletes a previously created touch region. Events from undefined regions return the value 255 by default.				
ID	Byte	Unique region identification number.		

11.5 Delete All Touch Regions				v8.0
	Dec	254 134		
	Hex	FE 86		
	ASCII	■ å		
Deletes all previously created touch regions. Recommended for use before dividing the screen into new regions.				

11.6 Create a Slider	Dec	254 186	ID Type X Y Width Height Control Width Min Max	v8.3
	Hex	FE BA	ID Type X Y Width Height Control Width Min Max	
	ASCII	■ ‖	ID Type X Y Width Height Control Width Min Max	

Draw a slider on the screen that responds visually and numerically when tapped or slid. Slider regions respond with a value of 83, their ID, then two byte length current X and Y coordinates when activated.

ID	**Byte**	Unique slider identification number, maximum 32 regions/sliders. value between 0 and 31
Type	**Byte**	Defines slider direction and starting point for the control, as below.
X	**Byte**	Leftmost coordinate.
Y	**Byte**	Topmost coordinate.
Width	**Short**	Width of slider.
Height	**Short**	Height of slider.
Control Width	**Byte**	Width of the slider control.
Min	**Short**	Minimum slider value.
Max	**Short**	Maximum slider value.

Table 35: Slider Definition

Value	Description
16	Horizontal slider, starting at minimum position
17	Vertical slider, starting at minimum position
32	Horizontal slider, starting at maximum position
33	Vertical slider, starting at maximum position
64	Horizontal slider, starting at middle position
65	Vertical slider, starting at middle position

11.7 Delete a Slider	Dec	254 189	ID	v8.3
	Hex	FE BD	ID	
	ASCII	■ ╜	ID	

Deletes a previously created slider. Memory is shared with touch regions, this command will free space.

ID	**Byte**	Unique region identification number.

11.8 Delete All Sliders	Dec	254 190	v8.3
	Hex	FE BE	
	ASCII	■ ╛	

Deletes all previously created sliders. Does not remove touch regions.

11.9 Set Dragging Threshold	Dec	254 137	Threshold	v8.0
	Hex	FE 89	Threshold	
	ASCII	■ ë	Threshold	

Sets the distance a press is required to travel before a drag event is reported. Precision will vary inversely to data transmitted; care should be taken to find a suitable balance. Distance is calculated as$\Delta x^2 + \Delta y^2 = d^2$.

Threshold	**Byte**	Dragging threshold value. Default is 8.

11.10 Set Pressure Threshold	Dec	254 138	Threshold	v8.0
	Hex	FE 8A	Threshold	
	ASCII	■ è	Threshold	
Sets the pressure required to trigger a touch event.				
Threshold	**Short**	Pressure threshold value. Default is 1000.		

11.11 Run Touchpad Calibration	Dec	254 139	v8.0
	Hex	FE 8B	
	ASCII	■ ï	
Triggers an interactive calibration of the touchpad. User will be required to touch various points on the screen during calibration. This command is recommended for use when environmental or user conditions change to ensure correct operation.			
Response	**Short**	Command byte 254, then 21 for success or 20 for failure.	

6.12 Display Functions

12.1 Backlight On				v8.0
	Dec	254 66	Minutes	
	Hex	FE 42	Minutes	
	ASCII	■ B	Minutes	

Turns the display backlight on for a specified length of time. If an inverse display color is used this command will essentially turn on the text.

Minutes	Byte	Number of minutes to leave backlight on, a value of 0 leaves the display on indefinitely.

12.2 Backlight Off				v8.0
	Dec	254 70		
	Hex	FE 46		
	ASCII	■ F		

Turns the display backlight off. If an inverse display colour is used this command will turn off the text.

12.3 Set Brightness				v8.0
	Dec	254 153	Brightness	
	Hex	FE 99	Brightness	
	ASCII	■ Ö	Brightness	

Immediately sets the backlight brightness. If an inverse display color is used this represents the text colour intensity instead. Default is 255.

Brightness	Byte	Brightness level from 0(Dim) to 255(Bright).

12.4 Set and Save Brightness				v8.0
	Dec	254 152	Brightness	
	Hex	FE 98	Brightness	
	ASCII	■ ÿ	Brightness	

Immediately sets and saves the backlight brightness. Although brightness can be changed using the set command, it is reset to this saved value on start up. Default is 255.

Brightness	Byte	Brightness level from 0(Dim) to 255(Bright).

12.5 Set Contrast				v8.0
	Dec	254 80	Contrast	
	Hex	FE 50	Contrast	
	ASCII	■ P	Contrast	

Immediately sets the contrast between background and text. If an inverse display color is used this also represents the text brightness. Default is 128.

Contrast	Byte	Contrast level from 0(Light) to 255(Dark).

12.6 Set and Save Contrast				v8.0
	Dec	254 145	Contrast	
	Hex	FE 91	Contrast	
	ASCII	■ æ	Contrast	

Immediately sets and saves the contrast between background and text. Although contrast can be changed using the set command, it is reset to this saved value on start up. Default is 128.

Contrast	Byte	Contrast level from 0(Light) to 255(Dark).

6.13 Scripting

13.1 Upload a Script File				v8.3
	Dec	254 92 2	ID Length Data	
	Hex	FE 5C 02	ID Length Data	
	ASCII	■ \ STX	ID Length Data	
Save a list of commands to be executed at a later time. Bytes are saved as if they are being sent by the host, for upload protocol see the File Transfer Protocol or XModem Transfer Protocol entries.				
ID	**Short**	Unique identification number of the script, value between 0 and 1023.		
Length	**Integer**	Length of the script in bytes.		
Data	**Byte(s)**	Data to be sent to the display when the script executes.		

13.2 Set Scripted Key				v8.4
	Dec	254 141	ID Row Column Down Script Up Script	
	Hex	FE 8D	ID Row Column Down Script Up Script	
	ASCII	■ ì	ID Row Column Down Script Up Script	
Create a key behaviour that responds to a press event by executing an uploaded script.				
ID	**Byte**	Unique key identification number, maximum based on number of keys available.		
Row	**Byte**	The row value of the key to be linked to the specified scripts.		
Column	**Byte**	The column value of the key to be linked to the specified scripts.		
Down Script	**Short**	Identification number of the script to run on a down event, value between 0 and 1023.		
Up Script	**Short**	Identification number of the script to run on an up event, value between 0 and 1023.		

***Note:** The command number for Set Scripted Key is 142 at all firmware revisions less than 8.4.

13.3 Set Scripted Button				v8.3
	Dec	254 142	ID X Y Width Height Type Down Script Up Script	
	Hex	FE 8E	ID X Y Width Height Type Down Script Up Script	
	ASCII	■ Ä	ID X Y Width Height Type Down Script Up Script	
Create a button region that responds to a touch event by executing an uploaded script.				
ID	**Byte**	Identification number of the touch region, value between 0 and 31		
X	**Byte**	Leftmost coordinate.		
Y	**Byte**	Topmost coordinate.		
Width	**Byte**	Width of touch region.		
Height	**Byte**	Height of touch region.		
Type	**Byte**	Type of touch region. Must be 1.		
Down Script	**Short**	Identification number of the script to run on a down event, value between 0 and 1023.		
Up Script	**Short**	Identification number of the script to run on an up event, value between 0 and 1023.		

13.4 Run Script File				v8.3
	Dec	254 93	ID	
	Hex	FE 5D	ID	
	ASCII	■]	ID	
Execute a previously loaded script. Script 0 is loaded automatically on startup, unless in override mode.				
ID	**Short**	Identification number of the script to run, value between 0 and 1023.		

6.14 Filesystem

14.1 Delete Filesystem	Dec	254 33 89 33	v8.0
	Hex	FE 21 59 21	
	ASCII	■ ! Y !	

Completely erase all fonts and bitmaps from a graphic display. Extended length of the command is intended to prevent accidental execution. To ensure filesystem integrity, cycle power to the display after erasure.

14.2 Delete a File	Dec	254 173	Type ID	v8.0
	Hex	FE AD	Type ID	
	ASCII	■ ¡	Type ID	

Removes a single font or bitmap file given the type and unique identification number. Cycle power after deletion.

Type	**Byte**	0 for font or 1 for bitmap.
ID*	**Short**	Unique identification number of font or bitmap to be deleted, value between 0 and 1023.

***Note:** ID was changed from a Byte length at firmware revision 8.1

14.3 Get Filesystem Space	Dec	254 175	v8.0
	Hex	FE AF	
	ASCII	■ »	

Returns the amount of space remaining in the display for font or bitmap uploads.

Response	**Integer**	Number of bytes remaining in memory.

14.4 Get Filesystem Directory	Dec	254 179	v8.0
	Hex	FE B3	
	ASCII	■ |	

Returns a directory to the contents of the filesystem. The total number and type of each entry will be provided.

Response	**Short**	Number of entries.
	Byte(s) [8]	8 identification bytes for each entry.

Table 36: Filesystem Identification Bytes

Byte	7	6	5	4	3	2	1	0
Description	Size(MSB)	Size	Size	Size(LSB)	Type(4)/ID(4)	ID (LSB)	Start Page (MSB)	Start Page (LSB)

Table 37: Extended Byte Descriptions

Size	The complete file size.
Type/ID	First four bits designate file type, 0 for font or 1 for bitmap, remaining 12 bits indicate ID number.
Start Page	Memory start page, a value of 0 indicates entry is not in use.

***Note:** ID and Size were changed from Byte and Short lengths respectively at firmware revision 8.1

14.5 Filesystem Upload	Dec	254 176	Size Data	v8.0
	Hex	FE B0	Size Data	
	ASCII	■ ▒	Size Data	

This command will upload a filesystem image to the display. The size used is almost always the entire memory. Filesystem data can be uploaded LSB to MSB using the
File Transfer Protocol.

Size	**Integer**	Size of the filesystem to upload.
Data	**Byte(s)**	Filesystem data to upload.

14.6 Filesystem Download	Dec	254 48		v8.0
	Hex	FE 30		
	ASCII	■ 0		
Downloads complete filesystem containing all fonts and bitmaps stored in the display using the File Transfer Protocol. A veritable heap of data.				
Response	Integer	Size of the filesystem to download.		
	Byte(s)	Filesystem data to download.		

14.7 File Download	Dec	254 178	Type ID	v8.0
	Hex	FE B2	Type ID	
	ASCII	■ ▒	Type ID	
Downloads a single font or bitmap file from the display to the host using the File Transfer Protocol.				
Type	Byte	Variable length, see File Types .		
ID	Short	Unique identification number of font or bitmap to download, value between 0 and 1023.		
Response	Integer	File size.		
	Byte(s)	File data.		

***Note:** ID was changed from a Byte length at firmware revision 8.1

14.8 File Move	Dec	254 180	Old Type	Old ID	New Type	New ID	v8.0
	Hex	FE B4	Old Type	Old ID	New Type	New ID	
	ASCII	■ ┤	Old Type	Old ID	New Type	New ID	
Used to move a single file and/or alter the type of an existing file. Old ID location must be valid and new ID empty.							
Old Type	Byte	Original file type, value between 0 and 1023, see File Types .					
Old ID	Short	Original unique file identification number, value between 0 and 1023.					
New Type	Byte	New file type, see File Types .					
New ID	Short	New unique file identification number.					

Table 38: File Types

Font	Bitmap	Script	9-Slice	Animation
0	1	2	3	4

***Note:** ID was changed from a Byte length at firmware revision 8.1

14.9 XModem Filesystem Upload	Dec	254 219 133 6 48	Size	Data	v8.1
	Hex	FE DB 85 6 30	Size	Data	
	ASCII	■ ▌à ACK 0	Size	Data	
Upload a filesystem image to the display using the XModem protocol. The size used is almost always the entire memory. Filesystem data is uploaded LSB to MSB using the XModem Transfer Protocol.					
Size	Integer	Size of the filesystem to upload.			
Data	Byte(s)	Filesystem data to upload, must be padded to an even multiple of 256 bytes.			

14.10 XModem Filesystem Download	Dec	254 222 133 6 48	v8.3
	Hex	FE DE 85 6 30	
	ASCII	■ ▌à ACK 0	

Downloads the complete filesystem using the XModem Transfer Protocol. A veritable heap of data, transmitted at a decent pace.

Response	**Integer**	Size of the filesystem to download.
	Byte(s)	Filesystem data to download, an even multiple of 256 bytes.

14.11 XModem File Upload	Dec	254 220 133 6 48	File ID Type Size Data	v8.3
	Hex	FE DC 85 6 30	File ID Type Size Data	
	ASCII	■ ▄ à ACK 0	File ID Type Size Data	

Uploads a single file to the display using the XModem Transfer Protocol. Unlike the standard protocol, there is one XModem upload command for all file types, see File Types for a complete list.

File ID	**Short**	Unique identification number for the file to upload, value between 0 and 1023.
Type	**Byte**	Type of file to upload, see File Types .
Size	**Integer**	Size of the file to upload.
Data	**Byte(s)**	File data to upload, must be padded to an even multiple of 128 bytes.

14.12 XModem File Download	Dec	254 221 133 6 48	File ID Type	v8.3
	Hex	FE DD 85 6 30	File ID Type	
	ASCII	■ ▌ à ACK 0	File ID Type	

Downloads a single file from the display to the host using the XModem Transfer Protocol.

File ID	**Short**	Unique identification number for the file to download, value between 0 and 1023.
Type	**Byte**	Type of file to download, see File Types .
Response	**Integer**	Size of the filesystem to download.
	Byte(s)	Filesystem data to download, an even multiple of 128 bytes, may be padded with 255s.

File Transfer Protocol

Once a bitmap or font file has been created and paired to its command it must be sent using a file protocol developed specifically for Matrix Orbital displays. Once a file upload command has been sent requesting a unique reference number and specifying the file size required, the display will respond indicating whether it has enough room to save the file or not. As is the case throughout the upload protocol, a response of 1 will indicate confirmation while an 8 corresponds to rejection and will terminate the session.

Table 39: Upload Protocol Responses

Value	Action	Description
1	Acknowledged	Transfer successful, upload continues
8	Not Acknowledged	Transfer failed, abort upload

Once a file is confirmed to fit within the display, the upload will begin. A protocol is used here to ensure each byte is uploaded successfully. After each byte is sent, the module will echo it back to the host. It should then be checked against the value originally sent before a confirmation byte of 1 is returned. If the transmitted and echoed values do not match the upload should be aborted by sending a value of 8 instead. The upload will continue in this manner as indicated by the examples below which utilize familiar font and bitmap files.

Table 40: Font Upload Protocol

Host	Display	Comments
254		Command Prefix
36		Upload Font File Command
1		Reference ID LSB
0		Reference ID MSB
31		Font File Size LSB
0		Font File Size
0		Font File Size
0		Font File MSB
	1	Acknowledge Size
5		First Font Data Byte
	5	Echo Data Byte
1		Acknowledge Data Byte
7		Second Font Data Byte
…	…	…
96		Last Font Data Byte
	96	Echo Data Byte
1		Acknowledge Data Byte

Table 41: Bitmap Upload Protocol

Host	Display	Comments
254		Command Prefix
94		Upload Bitmap File Command
1		Reference ID LSB
0		Reference ID MSB
5		Bitmap File Size LSB
0		Bitmap File Size
0		Bitmap File Size
0		Bitmap File MSB
	1	Acknowledge Size
5		First Bitmap Data Byte
	5	Echo Data Byte
1		Acknowledge Data Byte
4		Second Bitmap Data Byte
…	…	…
224		Last Bitmap Data Byte
	224	Echo Data Byte
1		Acknowledge Data Byte

It should be noted that the display has a timeout setting of 2.1 seconds before it resets to prevent it from hanging during the upload process. Upon reset, the values 254 and 212 will be returned to indicate an error or lengthy delay has occurred in the upload process. If everything goes smoothly, the protocol will end with the host transmitting a final confirmation byte and the font will be stored in the display ready for any application.

XModem Transfer Protocol

In addition to its original simple upload format, Matrix Orbital has added an XModem based protocol. This facilitates much faster download speeds by increasing the packet size from 1 byte to 128 bytes and using only a two byte CRC for error checking, greatly increasing throughput. To begin the upload, a series of command bytes are sent, a list of valid file type bytes is show in the File Types table. Once the command bytes are sent, the true size of the file is sent in four bytes, least significant byte first. At this point the display will respond with a C if the file fits or a NAK otherwise. Please note that these values are different than those of the original protocol as seen in the XModem Message Bytes table. If a NAK is seen at any point by the host, the upload is to be aborted in the same fashion as the regular protocol. If the file will fit, the start of header byte will be sent by the host, followed by a block count, in regular and inverted format, representing the number of 128 byte blocks remaining to be sent. The display will then check to make sure the block count value matches its own, if it doesn't it will NAK. The host can then

send a 128 byte block of data followed by that blocks high and low CRC16 bytes. The display then performs a CRC check on the data receive and ACKs if it matches that which was sent. Transfer continues with a block count and continues in this way until the end of file is reached. Files may be padded with 255 values to reach an even multiple of 128 bytes in size, but the download command will always report true size. Once the end of the upload file is reached, the host should transmit a single end of transmission byte. If the end of file is expected, the display will ACK one last time.

Table 42: XModem File Upload Protocol

Host	Display	Comments
254		Command Prefix
220		XModem Upload Command
133		Command Byte One
6		Command Byte Two
48		Command Byte Three
1		File ID LSB
0		File ID MSB
1		File Type
0		Size LSB
0		Size
1		Size
0		Size MSB
	67	C (If file fits)
1		Start of Header
128		Block Count
127		Inverted Block Count (255-Count)
<128 B>		128 Byte Data Block
30		*CRC MSB
71		*CRC LSB
	6	ACK (NAK if counts don't match)
…	…	…
4		End of Transmission
	6	ACK (NAK if EOT is not expected)

Table 43: XModem File Download Protocol

Host	Display	Comments
254		Command Prefix
221		XModem Download Command
133		Command Byte One
6		Command Byte Two
48		Command Byte Three
1		File ID LSB
0		File ID MSB
1		File Type
	0	Size LSB (NAK if not found)
	0	Size
	1	Size
	0	Size MSB
67		C
	1	Start of Header
	128	Block Count
	127	Inverted Block Count (255-Count)
	<128 B>	128 Byte Data Block
	30	*CRC MSB
	71	*CRC LSB
6		ACK (NAK if counts don't match)
…	…	…
	4	End of Transmission
6		ACK (NAK if EOT is not expected)

Table 44: XModem Message Bytes

Value	Action	Description
1	Start of Header	Begin upload transfer
4	End of Transmission	End completed upload transfer
6	Acknowledged	Transfer successful, upload continues
21	Not Acknowledged	Transfer failed, upload aborted
67	C	Confirmation that file will fit

***Note:** CRC bytes are calculated using the XMODEM CRC-CCITT algorithm available at: http://www.matrixorbital.ca/appnotes/XModem/ymodem.txt.

6.15 Data Security

15.1 Set Remember	Dec	254 147	Switch	v8.0
	Hex	FE 93	Switch	
	ASCII	■ ô	Switch	

Allows changes to specific settings to be saved to the display memory. Writing to non-volatile memory can be slow and each change consumes 1 write of at least 100,000 available. The Command Summary outlines which commands are saved always, never, and when this command is on only. Remember is off by default.

Switch	Byte	1 for on or 0 for off.

15.2 Set Data Lock	Dec	254 202 245 160	Level	v8.0
	Hex	FE CA F5 A0	Level	
	ASCII	■ ╩ ⌡ á	Level	

Temporarily locks certain aspects of the display to ensure no inadvertent changes are made. The lock is released after a power cycle. A new level overrides the old, and levels can be combined. Default is 0.

Level	Byte	Lock level, see Data Lock Bits table.

Table 45: Data Lock Bits

Display	Command	Filesystem	Setting	Address	Reserved	Reserved	Reserved
7	6	5	4	3	2	1	0

Table 46: Lock Parameters

Reserved	Place holders only, should be 0
Address	Locks the Baud Rate and I2C address
Setting	Locks all settings from being saved
Filesystem	Locks all bitmaps and fonts
Command	Locks all commands, text can still be written
Display	Locks entire display, no new text can be displayed

15.3 Set and Save Data Lock	Dec	254 203 245 160	Level	v8.0
	Hex	FE CB F5 A0	Level	
	ASCII	■ ╦ ⌡ á	Level	

Locks certain aspects of the display to ensure no inadvertent changes are made. The lock is not affected by a power cycle. A new level overrides the old, and levels can be combined. Default is 0.

Level	Byte	See Data Lock Bits table.

6.16 Miscellaneous

16.1 Write Customer Data					v8.0
	Dec	254 52	Data		
	Hex	FE 34	Data		
	ASCII	■ 4	Data		
Saves a user defined block of data to non-volatile memory. Useful for storing display information for later use.					
Data	Byte [16]	User defined data.			

16.2 Read Customer Data			v8.0
	Dec	254 53	
	Hex	FE 35	
	ASCII	■ 5	
Reads data previously written to non-volatile memory. Data is only changed when written, surviving power cycles.			
Response	Byte [16]	Previously saved user defined data.	

16.3 Write to Scratchpad						v8.3
	Dec	254 204	Address	Length	Data	
	Hex	FE CC	Address	Length	Data	
	ASCII	■ ╠	Address	Length	Data	
Write information to a 256 byte volatile memory bank for later use.						
Address	Short	Address where data is to be saved in volatile memory. Value between 0 and 256.				
Length	Short	Length of data to be saved, in bytes. Value between 0 and 256, address limited.				
Data	Byte(s)	Data to be saved in volatile memory.				

16.4 Read from Scratchpad					v8.3
	Dec	254 205	Address	Length	
	Hex	FE CD	Address	Length	
	ASCII	■ ═	Address	Length	
Read information previously saved in 256 byte volatile memory bank.					
Address	Short	Address where data is saved in volatile memory. Value between 0 and 256.			
Length	Short	Length of data to be read, in bytes. Value between 0 and 256, address limited.			
Response	Byte(s)	Data saved at the specified location in volatile memory.			

16.5 Read Version Number			v8.0
	Dec	254 54	
	Hex	FE 36	
	ASCII	■ 6	
Causes display to respond with its firmware version number. Test.			
Response	Byte	Convert to hexadecimal to view major and minor revision numbers.	

16.6 Read Module Type	Dec	254 55	v8.0
	Hex	FE 37	
	ASCII	■ 7	
Causes display to respond with its module number.			
Response	**Byte**	Module number, see Sample Module Type Responses for a partial list.	

Table 47: Sample Module Type Responses

122	GLT240128	**114**	GLK240128-25
123	GLT240128-USB	**108**	GLK240128-25-USB
112	GLT240128-422	**111**	GLK240128-25-422

16.7 Read Screen	Dec	254 184	v8.1
	Hex	FE B8	
	ASCII	■ ╕	
Return a two byte screen size, followed by the current commanded state of each pixel on the screen.			
Response	**Byte**	Width of the screen in pixels.	
	Byte	Height of the screen in pixels.	
	Byte(s)	Boolean values of each pixel on the screen, starting top left moving right then down.	

7 Appendix

7.1 Command Summary

Available commands below include identifying number, required parameters, the returned response and an indication of whether settings are remembered always, never, or with remember set to on.

Table 48: Communication Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Change Baud Rate	57	39	9	Byte	None	Always
Change I2C Slave Address	51	33	3	Byte	None	Always
Transmission Protocol Select	160	A0	á	Byte	None	Remember On
Set Flow Control Mode	63	3F	?	Byte	None	Remember On
Set Hardware Flow Control Trigger Level	62	3E	>	Byte	None	Remember On
Turn Software Flow Control On	58	3A	:	Byte[2]	None	Remember On
Turn Software Flow Control Off	59	3B	;	None	None	Remember On
Set Software Flow Control Response	60	3C	<	Byte[2]	None	Remember On
Echo	255	FF		Short, Byte[]	Byte[]	Never
Delay	251	FB	√	Short	None	Never
Software Reset	253	FD	²	Byte[4]	Byte[2]	Never

Table 49: Text Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Clear Screen	88	58	X	None	None	Never
Go Home	72	48	H	None	None	Never
Set Cursor Position	71	47	G	Byte[2]	None	Never
Set Cursor Coordinate	121	79	y	Byte[2]	None	Never
Initialize Text Window	43	2B	+	Byte[5], Short, Byte[3]	None	Remember On
Set Text Window	42	2A	*	Byte	None	Never
Clear Text Window	44	2C	,	Byte	None	Never
Initialize Label	45	2D	-	Byte[7], Short, Byte{2}	None	Remember On
Update Label	46	2E	.	Byte, String	None	Never
Auto Scroll On	81	51	Q	None	None	Remember On
Auto Scroll Off	82	52	R	None	None	Remember On

Table 50: Drawing Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Set Drawing Colour	99	63	c	Byte	None	Remember On
Draw Pixel	112	70	p	Byte[2]	None	Never
Draw a Line	108	6C	l	Byte[4]	None	Never
Continue a Line	101	65	e	Byte[2]	None	Never
Draw a Rectangle	114	72	r	Byte[5]	None	Never
Draw a Filled Rectangle	120	78	x	Byte[5]	None	Never
Draw a Rounded Rectangle	128	80	Ç	Byte[5]	None	Never
Draw a Filled Rounded Rectangle	129	81	ü	Byte[5]	None	Never
Draw a Circle	123	7B	{	Byte[3]	None	Never
Draw a Filled Circle	124	7C	|	Byte[3]	None	Never
Draw an Ellipse	125	7D	}	Byte[4]	None	Never
Draw a Filled Ellipse	127	7F	DEL	Byte[4]	None	Never
Scroll Screen	89	59	Y	Byte[4], Short[2]	None	Never
Initialize a Bar Graph	103	67	g	Byte[6]	None	Remember On
Initialize 9-Slice Bar Graph	115	73	s	Byte[6], Short[2]	None	Remember On
Draw a Bar Graph	105	69	i	Byte[2]	None	Never
Initialize a Strip Chart	106	6A	n	Byte[5], Short[2], Byte[2], Short	None	Remember On
Update a Strip Chart	107	6B	o	Byte, Short	None	Never

Table 51: Font Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Upload a Font File	36	24	$	Short, Integer, Byte[]	See Font File Creation	Always
Set the Current Font	49	31	1	Short	None	Never
Set Font Metrics	50	32	2	Byte[5]	None	Remember On
Set Box Space Mode	172	AC	¼	Byte	None	Remember On

Table 52: Bitmap Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Upload a Bitmap File	94	5E	^	Short, Integer, Byte[]	See Bitmap File Creation	Always
Upload a Bitmap Mask	92 5	5C 05	\ ENQ	Short, Integer, Byte[]	See Bitmap File Creation	Always
Draw a Bitmap from Memory	98	62	b	Short, Byte[2]	None	Never
Draw a Partial Bitmap	192	C0	└	Short, Byte[4]	None	Never
Draw a Bitmap Directly	100	64	d	Byte[2], Byte[]	None	Never

Table 53: 9-Slice Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Upload a 9-Slice File	92 3	5C 03	\ ETX	Short, Integer, Byte[]	See 9-Slice File Creation	Always
Upload a 9-Slice Mask	92 6	5C 06	\ ACK	Short, Integer, Byte[]	See 9-Slice File Creation	Always
Display a 9-Slice	91	5B	[	Short, Byte[4]	None	Never

Table 54: Animation Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Upload an Animation File	92 4	5C 04	\ EOT	Short, Integer, Byte[]	See Animation File Creation	Always
Display Animation	193	C1	⊥	Byte[4], Byte[]	None	Never
Delete Animation	199	C7	╟	Byte	None	Always
Start/Stop Animation	194	C2	┬	Byte[2]	None	Never
Set Animation Frame	197	C5	┼	Byte[2]	None	Never
Get Animation Frame	196	C4	–	Byte	Byte	Never

Table 55: General Purpose Output Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
General Purpose Output On	86	56	V	Byte	None	Never
General Purpose Output Off	87	57	W	Byte	None	Never
Set Start Up GPO State	195	C3	├	Byte[2]	None	Always

Table 56: Piezo Buzzer Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Activate Piezo Buzzer	187	BB	╗	Short[2]	None	Never
Set Default Buzzer Beep	188	BC	╝	Short[2]	None	Remember On
Set Keypad Buzzer Beep	182	B6	╢	Short[2]	None	Remember On
Set Touch Buzzer Beep	182	B6	╢	Short[2]	None	Remember On

Table 57: Keypad Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Auto Transmit Key Presses On	65	41	A	None	None	Remember On
Auto Transmit Key Presses Off	79	4F	`	None	None	Remember On
Poll Key Press	38	26	&	None	Byte	Never
Clear Key Buffer	69	45	E	None	None	Never
Set Debounce Time	85	55	U	Byte	None	Remember On
Auto Repeat Mode Off	96	60	`	None	None	Remember On
Assign Keypad Codes	213	D5	┌	Byte[25], Byte[25]	None	Always
Set Typematic Delay	159	9F	ƒ	Byte	None	Remember On
Set Typematic Interval	158	9E	₧	Byte	None	Remember On

Table 58: Touchpad Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Set Touch Mode	135	87	ç	Byte	None	Remember On
Set Region Reporting Mode	136	88	ê	Byte	None	Remember On
Set Touch Region	132	84	ä	Byte[7]	None	Remember On
Delete a Touch Region	133	85	à	Byte	None	Remember On
Delete All Touch Regions	134	86	╢	None	None	Remember On
Create a Slider	186	BA	╣	Byte[7], Short[2]	None	Remember On
Delete a Slider	189	BD	╜	Byte	None	Always
Delete All Sliders	190	BE	╛	None	None	Always
Set Dragging Threshold	137	89	ë	Byte	None	Remember On
Set Pressure Threshold	138	8A	è	Short	None	Remember On
Run Touchpad Calibration	139	8B	ï	None	Byte[2]	Always

Table 59: Display Functions Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Backlight On	66	42	B	Byte	None	Remember On
Backlight Off	70	46	F	None	None	Remember On
Set Brightness	153	99	Ö	Byte	None	Remember On
Set and Save Brightness	152	98	ÿ	Byte	None	Always
Set Contrast	80	50	P	Byte	None	Remember On
Set and Save Contrast	145	91	æ	Byte	None	Always

Table 60: Scripting Functions Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Upload a Script File	92 2	5C 02	\ STX	Short, Integer, Byte[]	None	Always
Set Scripted Button	142	8E	Ä	Byte[3], Short[2], Byte, Short[2]	None	Remember On
Set Scripted Key	141	8D	ì	Byte[3], Short[2]	None	Remember On
Run Script File	93	5D	]	Short	None	Never

Table 61: Filesystem Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Delete Filesystem	33, 89, 33	21, 59, 21	!, Y, !	None	None	Always
Delete a File	173	AD	¡	Byte, Short	None	Always
Get Filesystem Space	175	AF	»	None	Integer	Never
Get Filesystem Directory	179	B3	│	None	Byte[][8]	Never
Filesystem Upload	176	B0	░	Integer, Byte[]	None	Always
Filesystem Download	48	30	0	None	Integer, Byte[]	Never
File Download	178	B2	▓	Byte, Short	Integer, Byte[]	Never
File Move	180	B4	┤	Byte, Integer, Byte, Integer	None	Always
XModem Filesystem Upload	219, 133, 6, 48	DB, 85, 6, 30	Û, à, ACK, 0	Short, Byte, Integer, Byte[]	None	Always
XModem Filesystem Download	222, 133, 6, 48	DE, 85, 6, 30	Þ, à, ACK, 0	None	Integer, Byte[]	Never
XModem File Upload	220, 133, 6, 48	DC, 85, 6, 30	Ü, à, ACK, 0	Short, Byte, Integer, Byte[]	None	Always
XModem File Download	221, 133, 6, 48	DD, 85, 6, 30	Ý, à, ACK, 0	Short, Byte	Integer, Byte[]	Never

Table 62: Data Security Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Set Remember	147	93	ô	Byte	None	Always
Set Data Lock	202, 245, 160	CA, F5, A0	╩, ⌡, á	Byte	None	Remember On
Set and Save Data Lock	203, 245, 160	CB, F5, A0	╦, ⌡, á	Byte	None	Always

Table 63: Miscellaneous Command Summary

Name	Dec	Hex	ASCII	Parameters	Response	Remembered
Write Customer Data	52	34	4	Byte[16]	None	Always
Read Customer Data	53	35	5	None	Byte[16]	Never
Read Version Number	54	36	6	None	Byte	Never
Read Module Type	55	37	7	None	Byte	Never
Read Screen	184	B8	┐	None	Byte, Byte, Byte[]	Never
Write to Scratchpad	204	CC	╠	Byte, Short, Byte[]	None	Never
Read from Scratchpad	205	CD	═	Byte, Short	Byte[]	Never

7.1 Block Diagram

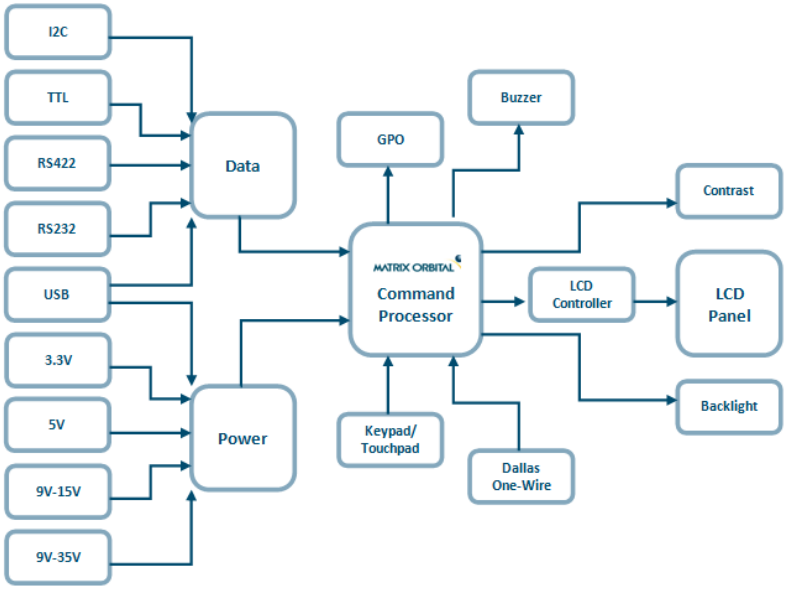


Figure 20: Functional Diagram

7.2 Environmental Specifications

Table 64: Environmental Limits

	Standard	*Extended (-E)
Operating Temperature	0°C to +50°C	-20°C to +70°C
Storage Temperature	-10°C to +60°C	-30°C to +80°C
Operating Relative Humidity	Maximum 90% non-condensing	

***Note:** The Extended Temperature option is not available for any variant of the GLT240128.

7.3 Electrical Tolerances

Current Consumption

Table 65: Current Consumption

Board		Backlight		GPOs
150mA	+	80 to 265 mA	+	20mA each maximum

Table 66: Backlight Current Draw

YG	GW & WB
265mA	80mA

Input Voltage Specifications

Table 67: Voltage Specifications

Standard	Extended Wide Voltage (-VPT)
4.75-5.25V	9.0-35.0V

7.4 Dimensional Drawings

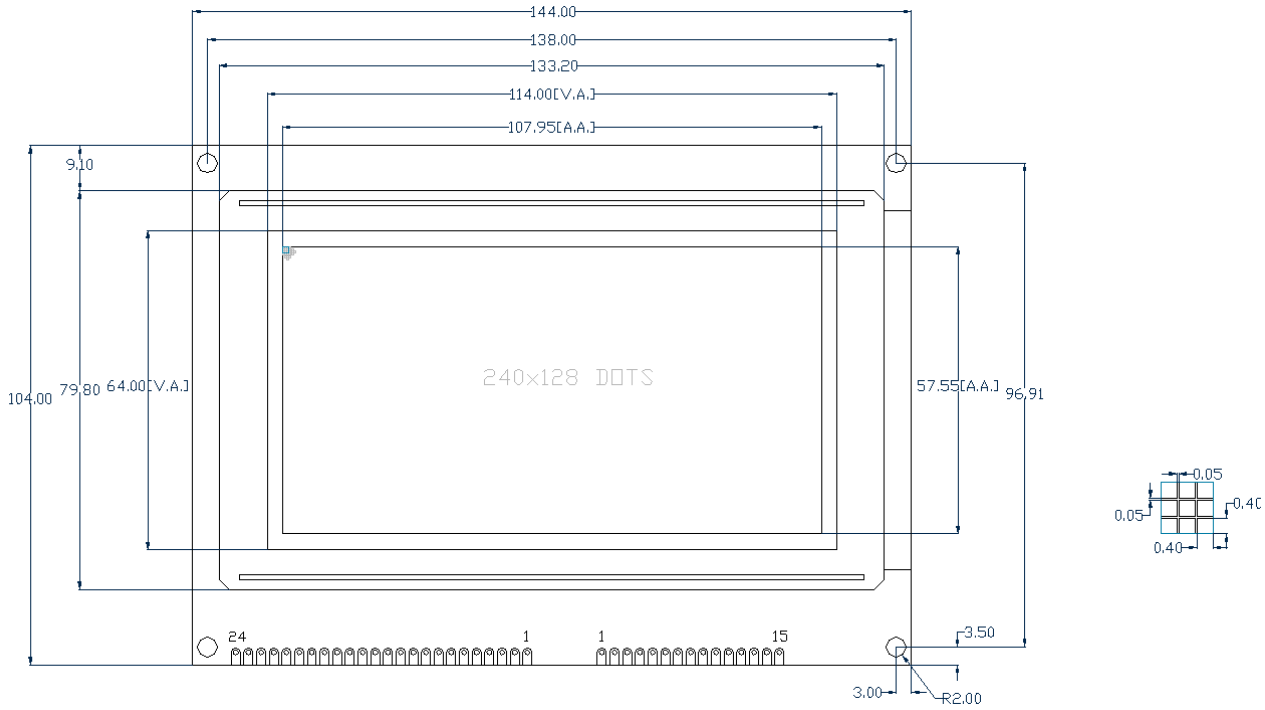


Figure 21: Display Dimensional Drawing

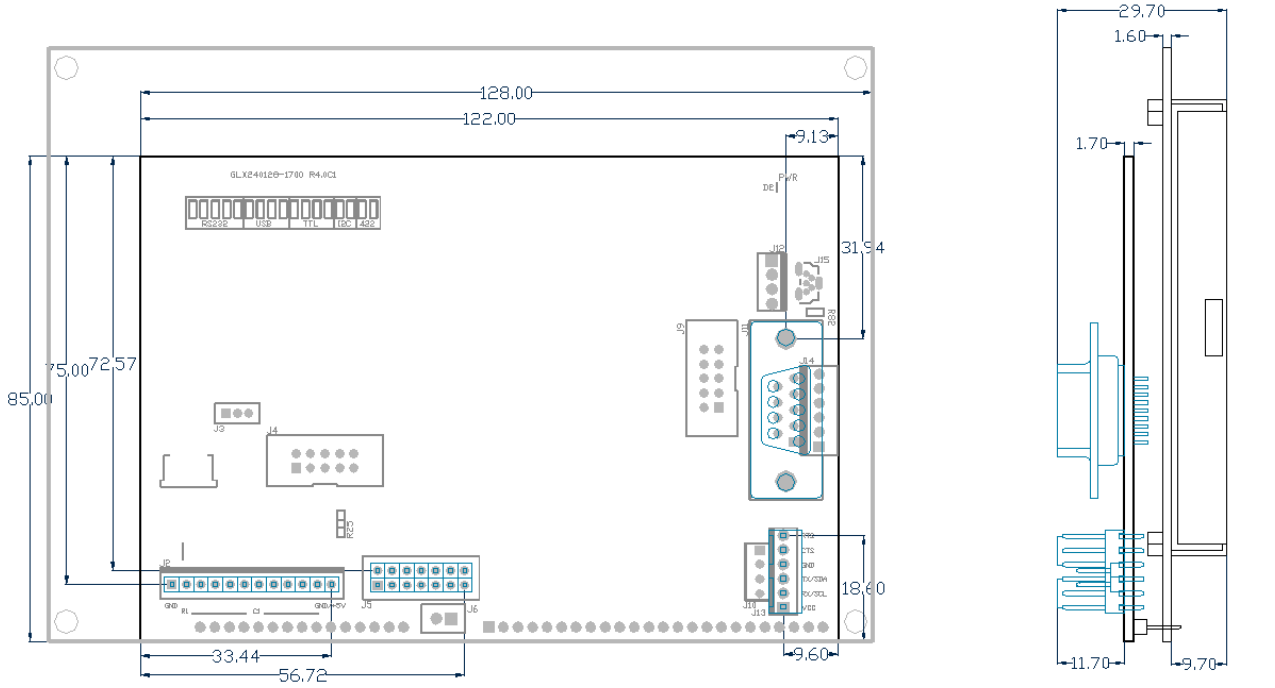


Figure 22: Standard Model Dimensional Drawing

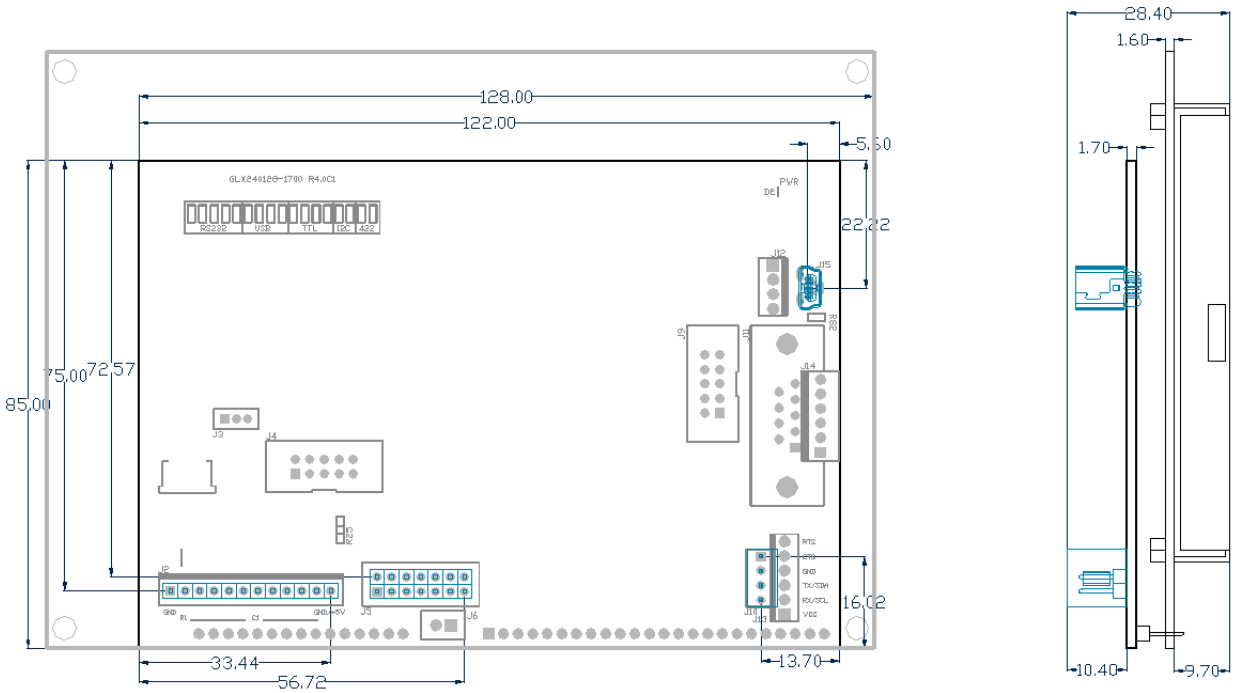


Figure 23: USB Model Dimensional Drawing

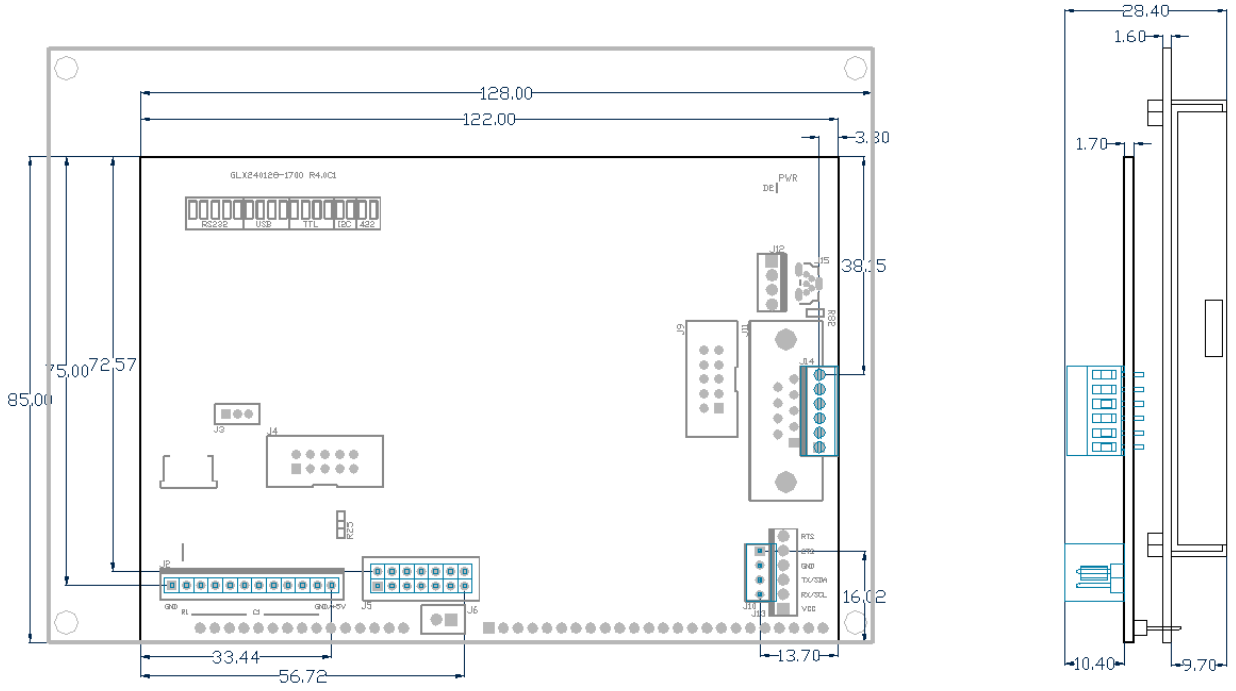


Figure 24: RS422 Model Dimensional Drawing

7.1 Optical Characteristics

Table 68: Display Optics

Module Size	144.00 x 104.00 x 27.8	mm
Viewing Area	114.0 x 64.0	mm
Active Area	107.95 x 57.55	mm
Pixel Size	0.40 x 0.40	mm
Pixel Pitch	0.45 x 0.45	mm
Viewing Direction	12	O'clock
Viewing Angle	-30 to +30	°
Contrast Ratio	3	
Backlight Half-Life (YG)	50,000	Hours
Backlight Half-Life (WB & GW)	20,000	Hours

***Note:** Backlight half-life is rated for normal operating conditions only: 25±10°C and 45±20% Relative Humidity.

8 Ordering

8.1 Part Numbering Scheme

Table 69: Part Numbering Scheme

GLT	-240128		-422	-WB	-VPT	-E
1	2	3	4	5	6	7

8.2 Options

Table 70: Display Options

#	Designator	Options
1	Product Type	GLK: Graphic Liquid Crystal Display with Keypad Input GLT: Graphic Liquid Crystal Display with Touchpad Input
2	Display Size	240128: 240 pixel columns by 128 rows
3	Keypad Size	*NP: No keypad 25: 25 key maximum
4	Protocol	*NP: Standard Model -USB: USB Only Model **-422: RS422 Only Model
5	Colour	*NP: Black Text with Yellow-Green Background FGW: Black Text with Grey-White Background WB: White Text with Blue Background
6	Voltage	*NP: Standard Voltage -VPT: Wide Voltage with Efficient Switching Power Supply
7	Temperature	*NP: Standard ***-E: Extended Temperature

***Note:** NP means No Populate; skip this designator in the part number and move to the next option.

****Note:** The RS422 model should only be powered from a local source, unless the –VPT variant is used.

*****Note:** Extended Temperature is available for keypad input units only; -E is not available for GLT models.

8.3 Accessories

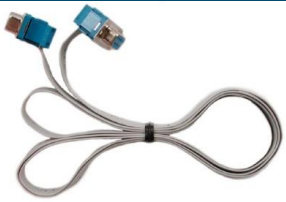
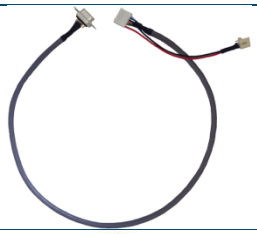
Power

Table 71: Power Accessories

PCS	Standard Power Cable	

Communication

Table 72: Communication Accessories

CSS1FT	1 ft. Serial Cable	
CSS4FT	4 ft. Serial Cable	
EXTMUSB3FT	Mini-USB Cable	
INTMUSB3FT	Internal Mini-USB Cable	
ESCCPC5V	Extended Serial Communication/5V Power Cable	
BBC	Breadboard Cable	

Peripherals

Table 73: Peripheral Accessories

KPP4x4	16 Button Keypad	

9 Definitions

ASCII: American standard code for information interchange used to give standardized numeric codes to alphanumeric characters.

BPS: Bits per second, a measure of transmission speed.

Byte: An unsigned data packet that is eight bits long.

FFSTN: Double film super-twisted nematic in reference to an LCD. The addition of two layers of film between the STN display and polarizer improves contrast.

GPO: General purpose output, used to control peripheral devices from a display.

GUI: Graphical user interface.

Hexadecimal: A base 16 number system utilizing symbols 0 through F to represent the values 0-15.

I^2C: Inter-integrated circuit protocol uses clock and data lines to communicate short distances at slow speeds from a master to up to 128 addressable slave devices. A display is a slave device.

Integer: An unsigned data packet that is thirty-two bits long, in little Endian format.

LSB: Least significant bit or byte in a transmission, the rightmost when read.

MSB: Most significant bit or byte in a transmission, the leftmost when read.

RS232: Recommended standard 232, a common serial protocol. A low level is -30V, a high is +30V.

RS422: Recommended standard 422, a more robust differential pair serial protocol.

SDA: Serial data line used to transfer data in I^2C protocol. This open drain line should be pulled high through a resistor. Nominal values are between 1K and 10K Ω.

SCL: Serial clock line used to designate data bits in I^2C protocol. This open drain line should be pulled high through a resistor. Nominal values are between 1K and 10K Ω.

Short: An unsigned data packet that is sixteen bits long, in little Endian format.

STN: Super-twisted nematic in reference to an LCD. In a relaxed or nematic state, crystals orientate themselves in the same direction and allow light to pass. In an excited state these crystals align to block light. Super-twisted crystals move from 180 to 270 degrees providing greater contrast than TN models.

TTL: Transistor-transistor logic applied to serial protocol. Low level is 0V while high logic is 5V.

10 Contact

Sales
Phone: 403.229.2737
Email: sales@matrixorbital.ca

Support
Phone: 403.204.3750
Email: support@matrixorbital.ca

Online
Purchasing: www.matrixorbital.com
Support: www.matrixorbital.ca

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Matrix Orbital:

GLK240128-25-WB-VPT-E GLK240128-25-E GLK240128-25-VPT GLK240128-25-422 GLK240128-25-FGW-422 GLK240128-25-FGW-USB GLK240128-25-USB GLK240128-25-WB-422 GLK240128-25-WB-USB GLK240128-25-422-E GLK240128-25-422-VPT GLK240128-25-FGW-422-E GLK240128-25-FGW-422-VPT GLK240128-25-FGW-USB-E GLK240128-25-FGW-V GLK240128-25-USB-E GLK240128-25-WB-422-VPT GLK240128-25-WB-USB-E GLK240128-25-WB-V GLK240128-25-VPT-E GLK240128-25-FGW-VPT GLK240128-25-FGW-VPT-E GLK240128-25-WB-VPT GLK240128-25-422-VPT-E GLK240128-25-422-FGW GLK240128-25-422-FGW-E GLK240128-25-422-FGW-VPT GLK240128-25-422-FGW-VPT-E GLK240128-25-422-WB GLK240128-25-422-WB-E GLK240128-25-422-WB-VPT GLK240128-25-422-WB-VPT-E GLK240128-25-USB-FGW GLK240128-25-USB-FGW-E GLK240128-25-USB-WB GLK240128-25-USB-WB-E GLK240128-25-WB-E GLK240128-25-FGW-E GLK240128-25-FGW