Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

GLK12232-25

Including GLK12232-25-USB

Technical Manual

Revision 3.1

PCB Revision: 3.0 or Higher

Firmware Revision: 5.8 or Higher

Revision History

Revision	Date	Description	Author
3.1	November 23, 2015	Initial Release	Divino

Contents

1 Introduction 5

2 Quick Connect Guide 6

2.1 Available Headers 6

2.2 Standard Module 7

Recommended Parts 7

Serial Connections 7

I^{2}C Connections 8

2.3 USB Module 9

Recommended Parts 9

USB Connections 9

3 Software 10

3.1 MOGD# 10

3.2 Application Notes 11

4 Hardware 12

4.1 Standard Model 12

I^{2}C Communication/Power Header 12

4.2 USB Model 12

Mini USB Connector 12

Alternate USB Header 12

4.3 Common Features 13

General Purpose Outputs 13

Keypad Header 13

Protocol Select Jumpers 14

Hardware Lock 14

5 Troubleshooting 15

5.1 Power 15

5.2 Display 15

5.3 Communication 15

5.4 Manual Override 16

6 Commands 17

6.1 Communication ... 17
6.2 Text ... 18
6.3 Drawing ... 19
6.4 Fonts ... 21
Font File Creation ... 22
6.5 Bitmaps ... 23
Bitmap File Creation ... 24
6.6 General Purpose Output ... 25
6.7 Keypad ... 25
6.8 Display Functions ... 27
6.9 Filesystem ... 28
File Transfer Protocol ... 29
6.10 Data Security ... 30
6.11 Miscellaneous ... 31
7 Appendix ... 33
7.1 Command Summary ... 33
7.2 Block Diagram ... 35
7.3 Environmental Specifications ... 36
7.4 Electrical Tolerances ... 36
7.5 Dimensional Drawings ... 37
7.6 Optical Characteristics ... 39
8 Ordering ... 39
8.1 Part Numbering Scheme ... 39
8.2 Options ... 39
8.3 Accessories ... 40
9 Definitions ... 41
10 Contact ... 41

1 Introduction

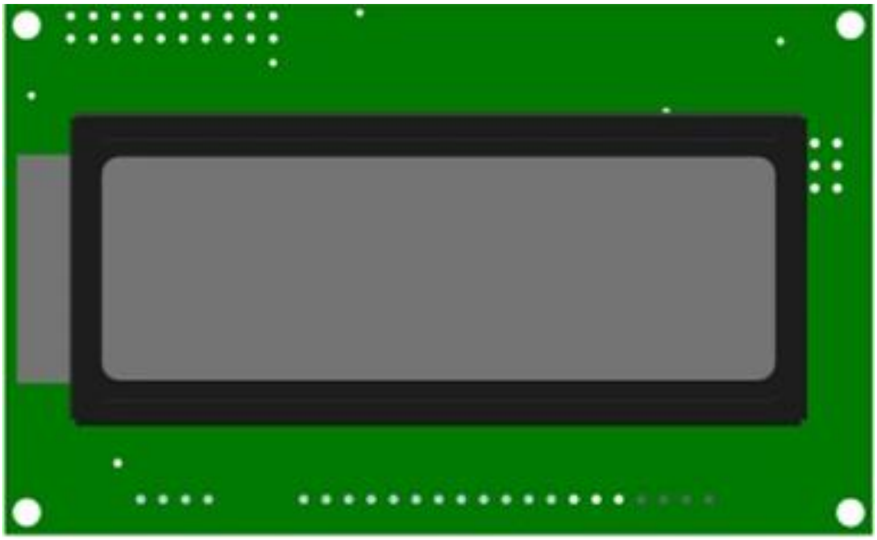


Figure 1: GLK12232-25 Display

The GLK12232-25 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, a USB communication model allows the GLK12232-25 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 16KB of customizable fonts and bitmaps to enhance the graphical user experience.

User input on the GLK12232-25 is available through a five by five matrix style keypad. Two general purpose outputs provide simple switchable five volt sources on each model.

The versatile GLK12232-25, with all the features mentioned above, is available in a variety of colour and temperature options to suit almost any application.

2 Quick Connect Guide

2.1 Available Headers

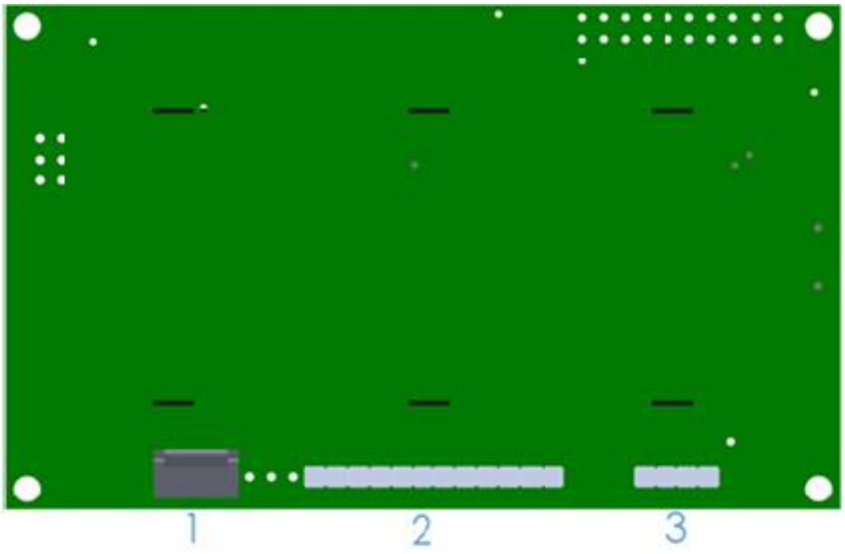


Figure 2: GLK12232-25 Standard Module Header Locations

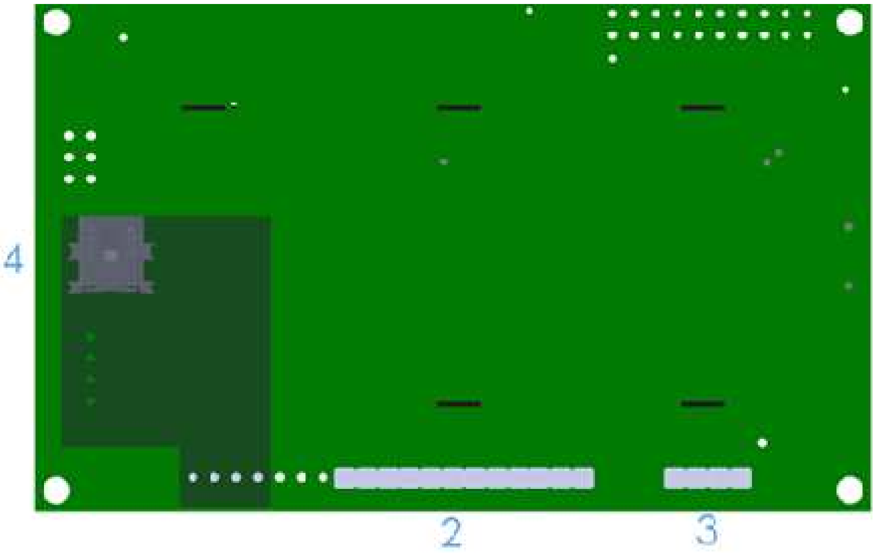


Figure 3: GLK12232-25 USB Model Header Locations

Table 1: List of Available Headers

#	Header	Mate	Population
1	Communication/Power Connector	ESCCPC5V/BBC	Standard Model Only
2	Keypad	KPP4x4	USB Model Only
3	GPO Header	None Offered	All Models
4	Mini USB Connector	EXTMUSB3FT/INTMUSB3FT	All Models

2.2 Standard Module

The standard version of the GLK12232-25 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^{2}C protocol. Connections for each protocol can be accessed through the four pin Communication/Power Header as outlined in the Serial Connections and I^{2}C Connections sections below.

Recommended Parts

Figure 4: 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 5: Breadboard Cable (BBC)

For a more flexible interface to the GLK12232-25 , 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 GLK12232-25 . The 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 Communication/Power Header of your GLK12232-25 .
   b. Insert the male end of your serial cable to the corresponding DB9 header of the 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^{2}C Connections

A more advanced connection to the GLK12232-25 is provided by the I^{2}C 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 GLK12232-25 in I^{2}C mode, get started with the guidelines below.

1. Set the Protocol Select switches.
   - I^{2}C: Ensure that the two I^{2}C jumpers in the corresponding protocol box are connected while all others are open.
2. Make the connections.
   a. Connect the Breadboard Cable to the Communication/Power Header on your GLK12232-25 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 GLK12232-25-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 6: Mini USB Cable (EXTMUSB3FT)

The External Mini USB cable is recommended for the GLK12232-25-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 GLK12232-25-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 GLK12232-25-USB please follow the steps below.

1. Set the Protocol Select jumpers.
   - USB: The GLK12232-25-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 GLK12232-25-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.

3 Software

The multiple communication protocols available and simple command structure of the GLK12232-25 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 GLK12232-25 .

Table 3: Communication Settings

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

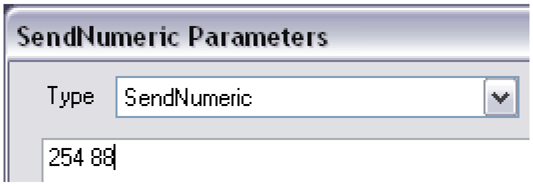
Finally, with a communication port identified and correctly setup simple text strings or even command bytes can easily be transmitted to control your display.

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.



SendNumeric Parameters

Type: SendNumeric

254 88

Figure 7: 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 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

I^{2}C Communication/Power Header

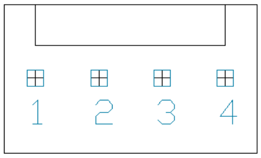


Figure 8: I2C Communication/Power Header

Table 4: I^{2}C Communication/Power Pinout

Pin	Function
1	Vcc
2	Rx (SCL)
3	Tx (SDA)
4	Gnd

Voltage is applied through pins one and four of the header, please reference the electrical specifications before applying power. Pins two and three are reserved for I^{2}C clock and data signals respectively, both of which should be pulled up to five volts using a resistance between one and ten kilohms. The Tyco 640456-4-LF style header used can be mated to a number of connectors, including Molex 22-01-3047.

4.2 USB Model

Mini USB Connector

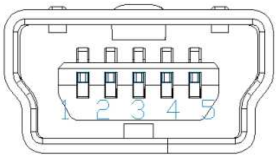


Figure 9: Mini USB Connector

Table 5: Mini USB Pinout

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

The GLK12232-25-USB -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 GLK12232-25-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.

4.3 Common Features

General Purpose Outputs

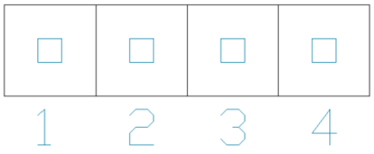


Figure 10: GPO Header

Table 6: GPO Pinout

Pin	Function
1	GPO 1
2	GND
3	GPO 2
4	GND

A unique feature of the GLK12232-25 is the ability to control relays***** and other external devices using either one of two General Purpose Outputs. Each can source up to 20mA of current at five volts when on or sink 20mA at zero volts when off. The straight, four 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.

Keypad Header

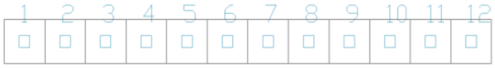


Figure 11: Keypad Header

Table 7: 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 GLK12232-25 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^{2}C 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 above the right of the keypad connector. Jump pads 1 & 2 for GND or 2 & 3 for +5V.

Protocol Select Jumpers

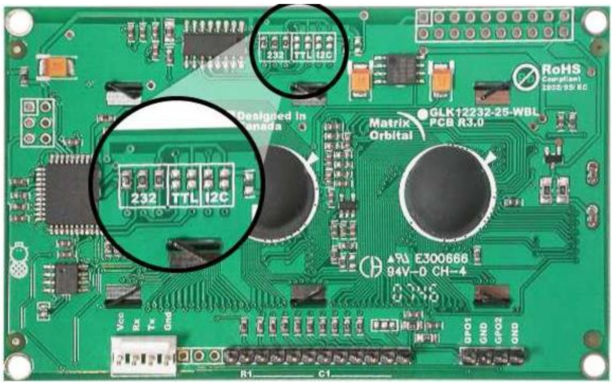


Figure 12: Protocol select Jumpers

The Protocol Select Jumpers provide the means necessary to toggle the GLK12232-25 between RS-232, TTL and I^2C 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 I^2C mode you must first remove the solder jumps from the RS232 jumpers and then place them on the I^2C jumpers. The display will now be in I^2C 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 I^2C jumpers and solder them to the TTL jumpers. Protocol resistors should be set to TTL for USB, and cannot be moved.

Hardware Lock

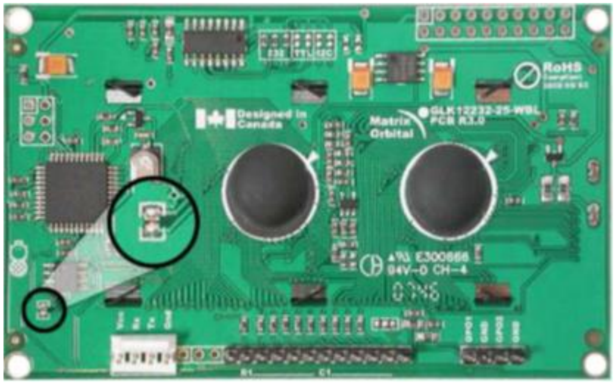


Figure 13: FileSystem Lock Jumper

The Hardware Lock allows fonts, bitmaps, and settings to be saved, unaltered by any commands. By connecting the two pads near the memory chip 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.

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 I^{2}C* please ensure that the data is being sent to the correct address. The default slave address for the display module is 80.

- In I^{2}C 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:** I^{2}C 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 GLK12232-25 model these are the middle two keypad pins.
3. Reconnect power to your unit, and wait for the start screen before removing the jumper. Please note the jumper will adversely affect 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
I^{2}C Address	80

Table 8: 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	v5.8
	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 9: 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	v5.8
	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 Set a Non-Standard Baud Rate	Dec	254 164	Speed	v5.0
	Hex	FE A4	Speed	
	ASCII	■ ñ	Speed	

Immediately changes the baud rate to a non-standard value. Speed must be a whole number between 977 and 153800. Due to rounding, error increases with baud rate, actual baud must be within 3% of desired baud to ensure accurate communication. Not available in I2C. Can be temporarily forced to 19200 by a manual override.

Speed	Short	Calculations shown below, standard crystal speed is 16MHz.

$$Speed = \frac{CrystalSpeed}{(8 \times DesiredBaud)} - 1$$

Equation 1: Speed Byte Calculation

$$ActualBaud = \frac{CrystalSpeed}{\big(8 \times (Speed + 1)\big)}$$

Equation 2: Actual Baud Rate Calculation

$$\frac{|DesiredBaud - ActualBaud|}{DesiredBaud} < 0.03$$

Equation 3: Baud Rate Error Calculation

1.4 Turn Software Flow Control On	Dec	254 58	AlmostFull AlmostEmpty	v5.8
	Hex	FE 3A	AlmostFull AlmostEmpty	
	ASCII	■ :	AlmostFull AlmostEmpty	

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 I2C. Default off.

AlmostFull	Byte	Number of bytes remaining before buffer is completely full. Value between 0 and 128.
AlmostEmpty	Byte	Number of bytes before buffer can be considered empty enough to accept data.

1.5 Turn Software Flow Control Off	Dec	254 59	v5.8
	Hex	FE 3B	
	ASCII	■ ;	
Disables flow control. Bytes sent to the display may be permitted to overflow the buffer resulting in data loss.			

6.2 Text

2.1 Clear Screen	Dec	254 88	v5.8
	Hex	FE 58	
	ASCII	■ X	
Clears the contents of the screen.			

2.2 Go Home	Dec	254 72	v5.8
	Hex	FE 48	
	ASCII	■ H	
Returns the cursor to the top left of the screen.			

2.3 Set Cursor Position	Dec	254 71	Column Row	v5.8
	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	Dec	254 121	X Y	v5.8
	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 Auto Scroll On	Dec	254 81	v5.8
	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.			

2.6 Auto Scroll Off	Dec	254 82	v5.8
	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	Dec	254 99	Colour	v5.8
	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	Dec	254 112	X Y	v5.8
	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, zero indexed from left.
Y	Byte	Vertical position of pixel to be drawn, zero indexed from top.

3.3 Draw a Line	Dec	254 108	X1 Y1 X2 Y2	v5.8
	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 the first terminus, zero indexed from left.
Y1	Byte	Vertical coordinate of the first terminus, zero indexed from top.
X2	Byte	Horizontal coordinate of second the terminus, zero indexed from left.
Y2	Byte	Vertical coordinate of second the terminus, zero indexed from top.

3.4 Continue a Line	Dec	254 101	X Y	v5.8
	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 the terminus, zero indexed from left.
Y	Byte	Top coordinate of the terminus, zero indexed from top.

3.5 Draw a Rectangle	Dec	254 114	Colour X1 Y1 X2 Y2	v5.8
	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 of the rectangle, zero indexed from left.
Y1	Byte	Topmost coordinate of the rectangle, zero indexed from top.
X2	Byte	Rightmost coordinate of the rectangle, zero indexed from left.
Y2	Byte	Bottommost coordinate of the rectangle, zero indexed from top.

3.6 Draw a Filled Rectangle	Dec	254 120	Colour X1 Y1 X2 Y2	v5.8
	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 of the filled rectangle, zero indexed from left.
Y1	Byte	Topmost coordinate of the filled rectangle, zero indexed from top.
X2	Byte	Rightmost coordinate of the filled rectangle, zero indexed from left.
Y2	Byte	Bottommost coordinate of the filled rectangle, zero indexed from top.

3.7 Initialize a Bar Graph	Dec	254 103	ID Type X1 Y1 X2 Y2	V5.8
	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 of the bar, zero indexed from left.
Y1	Byte	Topmost coordinate of the bar, zero indexed from top.
X2	Byte	Rightmost coordinate of the bar, zero indexed from left.
Y2	Byte	Bottommost coordinate of the bar, zero indexed from top.

Table 10: Bar Graph Types

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

3.8 Draw a Bar Graph	Dec	254 105	ID Value	V5.8
	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, value between 0 and 255.
Value	Byte	Portion of graph to fill in pixels, will not exceed display bounds.

3.9 Initialize a Strip Chart	Dec	254 110	ID X1 Y1 X2 Y2	V5.8
	Hex	FE 6E	ID X1 Y1 X2 Y2	
	ASCII	■ n	ID X1 Y1 X2 Y2	

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.

Table 11: Strip Chart Types (Bytes 3-0)

Type	Description
0	Bar
1	Line
2	Step
3	Box

3.10 Update a Strip Chart	Dec	254 111	ID Value	v5.8
	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	Dec	254 36	ID Size Data	v5.8
	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 entry. Default font is ID 1.

ID	Byte	Unique font identification number, value between 0 and 255.
Size	Short	Size of the entire font file.
Data	Byte(s)	Font file data, see the Font File Creation example.

4.2 Set the Current Font	Dec	254 49	ID	v5.8
	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	Byte	Unique font identification number, value between 0 and 255.

4.3 Set Font Metrics				
	Dec	254 50	LineMargin TopMargin CharSpace LineSpace Scroll	v5.8
	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.
LineSpace	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				
	Dec	254 172	Switch	v5.8
	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 12: 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 13: 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 14: 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 15: 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 16: 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	**v5.8**
	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 entry. Start screen is ID 1.

ID	**Byte**	Unique bitmap identification number, value between 0 and 255.
Size	**Short**	Size of the entire bitmap file.
Data	**Byte(s)**	Bitmap file data, see the Bitmap File Creation example.

5.2 Draw a Bitmap from Memory	Dec	254 98	ID X Y	v5.8
	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	Byte	Unique bitmap identification number, value between 0 and 255.
X	Byte	Leftmost coordinate of bitmap.
Y	Byte	Topmost coordinate of bitmap.

5.3 Draw a Bitmap Directly	Dec	254 100	X1 Y1 Data	v5.8
	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 effect of the Matrix Orbital graphic stylization ability.

Table 17: Smiley Face Bitmap

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

Table 18: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 19: Example Bitmap File

Header	5 4
Bitmap Data	80 34 224

6.6 General Purpose Output

6.1 General Purpose Output On	Dec	254 87	Number	v5.8
	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.

6.2 General Purpose Output Off	Dec	254 86	Number	v5.8
	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.

6.3 Set Start Up GPO State	Dec	254 195	Number State	v5.8
	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.

6.7 Keypad

7.1 Auto Transmit Key Presses On	Dec	254 65	v5.8
	Hex	FE 41	
	ASCII	■ A	

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

7.2 Auto Transmit Key Presses Off	Dec	254 79	v5.8
	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.

7.3 Poll Key Press	Dec	254 38	v5.8
	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).