



# On Code:

Application Note Code Samples

Rev 1.1

## Document Revisions

Version	Date	Author	Remarks
1.0	03-Mar-09	Clark	Initial Version
1.1	06-May-09	Clark	Removed Touchpad Section

## Table of Contents

Introduction.....	1
Serial Communication.....	2
Sending Text.....	3
Issuing Commands.....	3
On Reading.....	3
Conclusion.....	4



## Introduction

This document provides some insight into the coding techniques used to create the GLT240128 Application Note. These examples can be used to re-create the files employed in the demonstration program, or to create an entirely new application. While the code provided has been written in C# with Visual Basic 2008, it is documented in such a way as to be portable to any language.

## Serial Communication

Serial communication is the backbone of most Matrix Orbital display applications. Whether a device is in RS232 or USB mode, it will communicate to the host PC using a communication port. First, it is important to ensure that any libraries required to use the serial port are included in your program. For C# you use the following:

```
using System.IO.Ports;
```

Next, a serial port object will be required. At this point it may be necessary to know which com port your device is attached to and what baud rate it is communicating at. Using a program such as MOGD#, it is easy to autodetect any connected devices and procure the pertinent information. For C# you can use the following line to initialize your serial port object to communicate through com 1 at 19200 baud.

```
SerialPort port = new SerialPort("COM1", 19200);
```

Finally, it is important to be able to control the port to allow information to pass through. When starting a serial conversation it is important to open the serial port. In C# you'll use the following:

```
port.Open();
```

When communication is complete, it is equally important to remember to close the serial port, like so:

```
port.Close();
```

This simple coding technique will allow you to control your serial port so that text and commands can be successfully sent to and responses read from your Matrix Orbital display.

## Sending Text

When sending text to your Matrix Orbital display, it is important that each character is sent as its ASCII equivalent. For example to send the character 'A' to the screen, you would send the hexadecimal byte 0x41. It can be quite tedious to create an array of bytes, looking up the values as needed, however, you may find a function that does this work for you. In C# a byte array is returned using the following function.

```
byte[] message = System.Text.Encoding.ASCII.GetBytes("Hello World");
```

Once a byte array is created, it can then be sent to your display using the write command. In C# a parameter is required for both the offset or starting byte of the array and its length. The write function is shown below:

```
port_com.Write(message, 0, message.Length);
```

Once the write command is mastered, it can be used to send any message, advanced text and graphics, and even commands to your Matrix Orbital display.

## Issuing Commands

Like sending text, commands are issued to your Matrix Orbital display using byte arrays. However, in this case, these arrays must be populated by hand, there is no built in function to store these proprietary values. These byte arrays may be populated using a variety of different methods including decimal and hexadecimal notation as shown below. It is important to locate the proper values in your manual.

```
byte[] clear_screen = {254, 88}; //decimal notation
```

```
byte[] clear_screen = {0xFE, 0x58}; //hexadecimal notation
```

Once these byte arrays are populated, they can be sent through the serial port in much the same way the text arrays were transmitted. As you can see in the example below, there is little difference between sending text and issuing commands once their byte arrays are populated.

```
port.Write(clear_screen, 0, clear_screen.Length);
```

With the ability to issue commands and send text, you have the ability to control virtually every aspect of your Matrix Orbital display.

## On Reading

As with writing, you can read from your Matrix Orbital display using the serial port. In this case it is important to remember that in many instances a command must be issued before the display will provide a byte to be read. In C# the read function has three parameters; a byte array to receive the data, an offset, and the number of bytes to be read. A sample read of the version number, to be issued after the read version number command is shown below.

```
port.Read(version, 0, 1);
```

With the addition of the read command, full control of the Matrix Orbital display is yours.

## Conclusion

Using the code samples provided to create a serial port object, send text, issue commands, and read responses, you will be able to incorporate your Matrix Orbital display into any application you can imagine. If you do run into any bumps on the road to realizing your display dreams, Matrix Orbital technical support would be more than happy to help using any one of the methods listed below.

### **Matrix Orbital**

Support Site: [www.MatrixOrbital.ca](http://www.MatrixOrbital.ca)

Support Email: [support@matrixorbital.ca](mailto:support@matrixorbital.ca)

Support Line: 403.294.3750x250