

Display Setup & Programming Guide

NCD RS-232 Networkable Graphic Display Controllers



GU128x32-800

128x32 Vacuum Florescent Display



GU128x64-800

128x32 Vacuum Florescent Display



GU160x16-800

128x32 Vacuum Florescent Display



GU160x32-800

128x32 Vacuum Florescent Display



GU192x16-800

128x32 Vacuum Florescent Display



GU256x32-800

128x32 Vacuum Florescent Display

Device Features

- Control 256 Devices Simultaneously or Individually from a Single Serial Port
- User-Selectable Communication Rates from 9600 to 115.2K Baud
- E3C Compliant Command Set
- Holds more than 3,000 Icons for Large Characters and Small Animations
- Store and Recall Full-Screen Images
- 8 User-Programmable Character Fonts
- 16-Level Software Controlled Brightness
- Horizontal and Vertical Image Scrolling
- High-Contrast Long-Distance Visibility

Warranty

NCD Warrants its products against defects in materials and workmanship for a period of 90 days. If you discover a defect, NCD will, at its option, repair, replace, or refund the purchase price. Simply return the product with a description of the problem and a copy of your invoice (if you do not have your invoice, please include your name and telephone number). We will return your product, or its replacement, using the same shipping method used to ship the product to NCD.

This warranty does not apply if the product has been modified or damaged by accident, abuse, or misuse.

30-Day Money-Back Guarantee

If, within 30 days of having received your product, you find that it does not suit your needs, you may return it for a refund. NCD will refund the purchase price of the product, excluding shipping/handling costs. This guarantee does not apply if the product has been altered or damaged.

Copyrights and Trademarks

Copyright 2000 by NCD. All rights reserved. Other brand and product names are trademarks of registered trademarks of their respective holders.

Disclaimer of Liability

NCD is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, and any costs or recovering, reprogramming, or reproducing any data stored in or used with NCD products.

Technical Assistance

Technical questions should be e-mailed to Ryan Sheldon at ryan@controlanything.com. Technical questions submitted via e-mail are answered up to 20 times daily. Technical support is also available by calling (417) 646-5644.

NCD Contact Information**Mailing Address:**

National Control Devices
P.O. Box 455
Osceola, MO 64776

Telephone:

(417) 646-5644

FAX:

(417) 646-8302

Internet:

ryan@controlanything.com
www.controlanything.com
www.controleverything.com

Introduction to the Noritake –800 Series Vacuum Florescent Displays

The Noritake –800 series Graphic Vacuum Florescent Displays were designed to offer high-contrast, long-distance viewing area with a viewing angle exceeding most display technologies. If your application requires optimal visibility or a really flashy interface, the –800 series may be exactly what you are looking for.

The Noritake –800 series features two scrollable graphic layers. While they do NOT include a hardware character generator, they are low-cost by comparative technologies and are available in 6 convenient sizes.

The firmware we have custom developed for communicating to these displays adds a user-programmable character generator for displaying ASCII text. Since there are two hardware graphic layers built into the display, text may appear on one layer while graphics is shown on the other layer.

The Noritake –800 series firmware adds many features to these displays, making them much more powerful than their native hardware would normally allow. All NCD display controllers support a software-controlled brightness, E3C compliance (for allowing 256 devices to share a single serial port), a user-programmable character generator with 8 fonts that may be mixed on the display screen at any location. All NCD display controllers are capable of storing and recalling full-screen images, over 3,000 icons (small graphic images that may be pasted anywhere on the screen, often used for buttons and small screen animations). Other features include direct screen dump, allowing you to directly upload image data to the screen from the serial port, user-definable startup screen, user-upgradeable memory for storing additional full-screen images, vertical scrolling, and much more.

The NCD Image Loader Utility

It is possible to use your favorite Windows paint program to draw custom fonts, icons, and full-screen graphics. Once you have completed your custom artwork, the NCD Image Loader Utility (ILU) is used to permanently store your graphics into the display controller. This utility allows you to easily manage the storage of all art work into the controller as well as review store images, icons, and fonts. The ILU is also used to set default hardware options such as default display brightness and E3C device number. Additionally, the ILU helps you learn how to send commands to the display controller using an integrated help guide. The help guide is used to show actual data being sent to the display.

Additional information and documentation on the ILU can be found at www.controlanything.com. This guide should be used specifically for wiring diagrams and programming information. This guide assumes you have already used the ILU to store images in the display controller. Do NOT proceed to the programming section of this guide until you have explored the functions of this controller using the ILU. The ILU is a powerful and necessary Windows application that is used to configure your controller and graphically teach you how to use it.

Command Sets

This display controller has three types of command sets. ILU Commands, E3C Commands, and User Display Commands.

ILU commands are used exclusively by the NCD Image Loader Utility to configure the user interface. ILU commands are used to permanently store images and icons in the display controllers memory. They are also used to define the limits of scroll bars and identify display models and backlight options to the user interface. ILU commands are issued by the user using the NCD Image Loader Utility. The ILU command set is proprietary and will remain undocumented, as these commands are used strictly for the purposes of initially setting up and customizing the graphics in the display controller. In daily operation of the display controller, ILU commands have little value.

E3C commands are used to control up to 256 displays or other E3C compliant devices using a single serial port. An extensive explanation of the E3C command set can be found on page 6. The NCD Image Loader Utility is used to define the E3C device number of the display controller.

User Display Commands are used by the user in daily operation of the controller. They allow you to scroll the graphic screen, upload image data directly to the screen from the serial port, recall stored images and icons, and display text at any location on the screen. UDCs make the display controller very easy to use and greatly reduce the programming required to control the graphic display.

Memory Notice

This Display Controller uses nonvolatile EEPROM memory, which is rated to store data for 100+ years without the need for power. EEPROM memory is self-destructive in that it can only be written to ~100,000 times. EEPROM memory may be read an infinite number of times without any degradation of life.

National Control Devices

**DO NOT EXCEED +5 VOLTS DC
USE REGULATED SUPPLY ONLY**

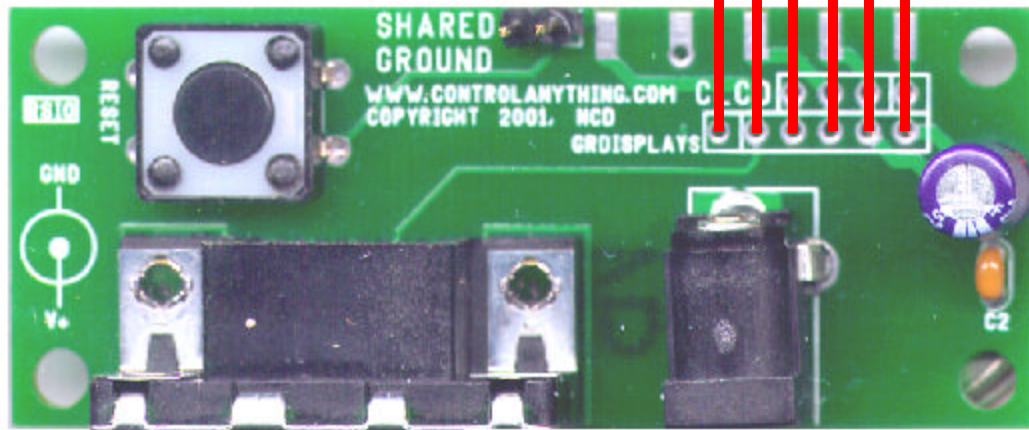
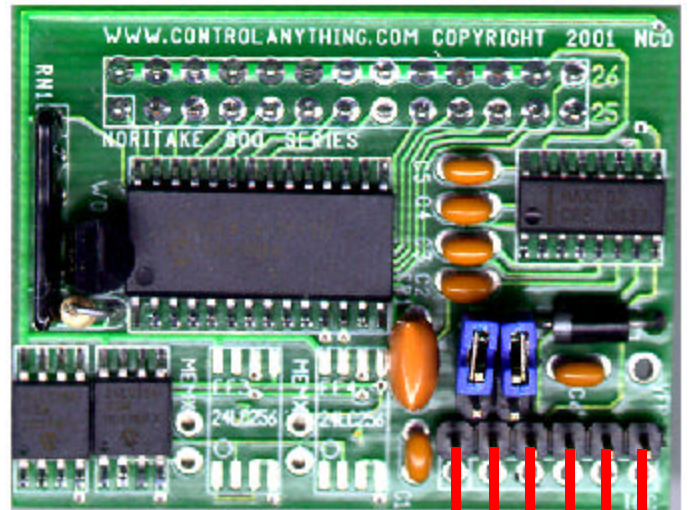
**2.5 mm Power Connector:
Center: +5 Volts DC, 1 Amp MAX
Outer Ring: Ground**

DB-9 Serial Connections:

- 1 Not Used
- 2 RS-232 Data Input Connects to RS-232 Output
- 3 RS-232 Data Output Connects to RS-232 Input
- 4 Not Used
- 5 RS-232 Signal Ground
- 6-9 Not Used

**Version 1.0 Firmware Supports
up to 4 Memory Chips.
Your controller comes with 2.
Please contact us for upgrade
information.**

**CPU RESET
BUTTON**



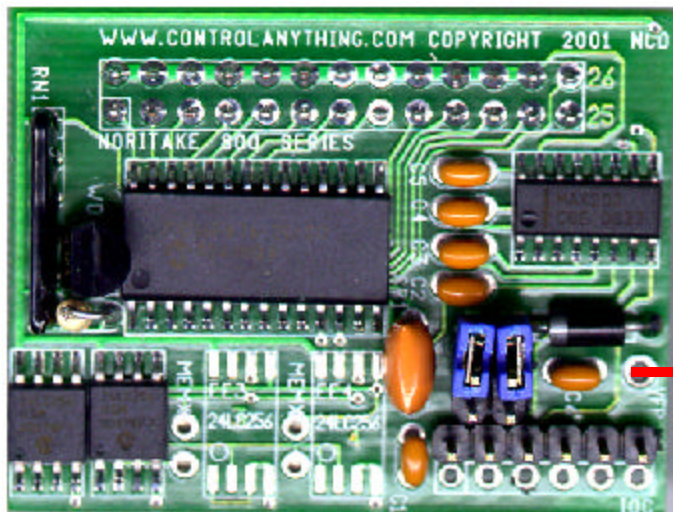
**Power
Connector**

**DB-9 Serial
Connector**

Baud Select:

Baud	J1	J2
9600	Off	Off
19.2K	On	Off
38.4K	Off	On
115.2K	On	On

**+5 Volt Supply
Connects to
Power Connector
on Display.**



1 2 3 4 5 6

- 1 RESET: Normally High, Connect to Ground Momentarily for RESET**
- 2 RS-232 DATA OUTPUT: Connect to RS-232 Data Input of Computer**
- 3 RS-232 DATA INPUT: Connect to RS-232 Data Output of Computer**
- 4 GROUND for RS-232 Data Signal AND Power Supply**
- 5 GROUND for RS-232 Data Signal AND Power Supply**
- 6 +5 Volt 1 Amp MAX Regulated Power Supply Input**

Sending Commands to the Display Controller

The display controller is capable of sending and receiving data via RS-232 serial communications and is compatible with just about any computer or microcontroller ever produced, including the Macintosh, Amiga, Basic Stamp, and of course, Windows & DOS based machines.

Regardless of the system you are using, you will need access to a programming language that supports program control of the serial port on your system.

A terminal program is not suitable for controlling this device. Commands should be sent using ASCII character codes 0-255 rather than ASCII characters (A, B, C etc.). See "ASCII Codes vs. Characters" on this page.

Most systems require you to open the appropriate serial port (COM port) prior to sending or receiving data.

Because there are so many different ways to send and receive data from various languages on various platforms, we will provide generic instructions that can be easily converted to your favorite language.

For example, if this manual says "Send ASCII 254", the user will need to translate this instruction into a command that is capable of sending ASCII character code 254.

To Send ASCII 254 from Visual Basic, you will use the following line:

```
MSComm1.Output = Chr$(254)
```

In Qbasic, you can send ASCII 254 using the following line of code:

```
Print #1, Chr$(254);
```

Note that sending ASCII character code 254 is NOT the same as sending ASCII characters 2, 5, and 4 from a terminal program. Typing 2, 5, and 4 on the keyboard will transmit three ASCII character codes.

Your programming language should support commands for reading data from the serial port. It is important to familiarize yourself with these instructions for reliable operation.

For your convenience, we have provided several programming examples in various popular programming languages for controlling this display controller. These examples should greatly speed development time. You may want to visit www.controlanything.com for the latest software and programming examples.

Programming examples for this device are much more extensive for Visual Basic 6 users than for any other programming language. If you are not a VB programmer, you may consider looking at the VB6 source code, as it is easily translated into other popular languages.

Regardless of your programming background, the provided Visual Basic 6 source code is very easy to understand and will likely resolve any communication questions you may have. VB6 programming examples may be viewed in any text editor.

ASCII Codes vs. Characters

The differences between ASCII codes and ASCII characters tend to generate a lot of confusion among first-time RS-232 programmers. It is important to understand that a computer only works with numbers. With regard to RS-232 data, the computer is only capable of sending and receiving numbers from 0 to 255.

What confuses people is the simple idea that the numbers 0 to 255 are assigned letters. For instance, the number 65 represents the letter A. The number 66 represents the letter B. Every character (including numbers and punctuation) is assigned a numeric value. This standard of assignments is called ASCII, and is a universal standard adopted by all computers with an RS-232 serial port.

ASCII characters codes can be clearly defined as numbers from 0 to 255.

ASCII characters however are best defined as letters, A, B, C, D, as well as punctuation, !@#\$%, and even the numbers 0-9.

Virtually all programming languages permit you to send ASCII in the form of letters or numbers. If you wanted to send the word "Hello" out the serial port, it is much easier to send the letters H, e, l, l, and o than it is to send the ASCII character codes that represent each letter.

For the purposes of controlling NCD devices however, it is much easier to build a numeric command set. Especially when communicating to devices that have numeric parameters, such as the E3C command set.

Put simply, it is easier to control NCD devices using ASCII character codes 0 to 255 than it is to use ASCII characters A, B, C, D, etc.

Because terminal programs are ASCII character based, it may be difficult to generate the proper series of keystrokes that would be necessary to activate a particular function. Therefore, terminal programs are not suitable for controlling NCD devices. In a real world control application, a terminal program would not likely be used to control NCD devices anyway. Therefore, a programming language that supports the transmission and reception of ASCII character codes 0 to 255 is highly recommended.

The E3C Command Set: Software Control of Multiple NCD Devices

The E3C command set allows you to control up to 256 NCD devices from a single serial port. It is OK to mix different types of devices, as long as the devices are E3C compliant. This display controller supports the full set of E3C commands.

How does E3C Work?

First of all, each device must be assigned a device number from 0 to 255. The display controller must be programmed with a device number using the NCD Image Loader Utility.

E3C stands for Enabled 3-Wire Communication. Put simply, when you first power up your computer and all the devices attached to the serial port, all devices will respond to your commands.

Using the E3C command set, you can specify which devices will listen and which devices will ignore your commands. Note that E3C commands are never ignored by any device, regardless of the commands you send to the controller.

The number to the left of each command indicates the ASCII character code that must be sent to issue the command. All commands must be preceded with ASCII character code 254 to place the device in command mode. See examples at right.

The E3C Command Set

248 Enable All Devices:

Tells all devices to respond to your commands.

249 Disable All Devices:

Tells all devices to ignore your commands.

250 Enable a Selected Device:

Tells a specific device to listen to your commands.

251 Disable Selected Device:

Tells a specific device to ignore your commands.

252 Enable Selected Device Only:

Tells a specific device to listen to your commands, all other devices will ignore your commands.

253 Disable a Selected Device Only:

Tells a specific device to ignore your commands, all others will listen.

E3C Visual Basic Programming Examples

The E3C command set is easily used from any programming language that supports serial communication. The following Visual Basic 6 Example source code demonstrates subroutines that can be used to control which devices will listen and which devices will ignore your commands.

Many commands issued to the display controller are acknowledged by sending ASCII character code 85 back to the host compute. E3C commands are not acknowledged.

Sample Code: The E3C Command Set

```
Public Sub EnableAllDevices()  
    'Enable All E3C Devices  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(248) 'E3C Enable All Device Command  
End Sub  
  
Public Sub DisableAllDevices()  
    'Disable All E3C Devices  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(249) 'E3C Disable All Device Command  
End Sub  
  
Public Sub EnableSpecificDevice(Device)  
    'Enable A Specific E3C Devices, Other Devices will be unchanged  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(250) 'E3C Disable Specific Device Command  
    MSComml.Output = Chr$(Device) 'Device Number that will be Disabled  
End Sub  
  
Public Sub DisableSpecificDevice(Device)  
    'Disable A Specific E3C Devices, Other Devices will be unchanged  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(251) 'E3C Disable Specific Device Command  
    MSComml.Output = Chr$(Device) 'Device Number that will be Disabled  
End Sub  
  
Public Sub DisableAllDevicesExcept(Device)  
    'Disable All E3C Devices Except (Device)  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(252) 'E3C Disable All Device Except Command  
    MSComml.Output = Chr$(Device) 'Device Number that will be Active  
End Sub  
  
Public Sub EnableAllDevicesExcept(Device)  
    'Enable All E3C Devices Except (Device)  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(253) 'E3C Enable All Device Except Command  
    MSComml.Output = Chr$(Device) 'Device Number that will be Inactive  
End Sub
```

The Noritake –800 Series Display Controller Command Set

This display controllers supports an extensive command set, used to control many types of graphic functions directly on the display screen. The best way to familiarize yourself with the capabilities of this display device is to carefully read through the command set in this section. The “plain English” examples provide a quick, easy to understand definition of what each command does.

The number to the left of each command indicates the ASCII character code that must be sent to issue the command. All commands must be preceded with ASCII character code 254 to place the device in command mode. See examples at right.0

Controlling the Display

0 Test 2-Way

This command is used to test 2-way communication between the display controller and the host computer. The host computer will receive ASCII character code 85 if the device is properly connected and functioning.

1 Send Command

This command is used to send a direct hardware command to the display. This function will not report data back to the computer, nor can it be used to ask the display for screen information. A list of commands that may be issued to the display can be found in the data sheet for the controller built into the display screen. For the most part, this command was used for development purposes but may be useful for issuing commands not yet supported by our firmware. For experienced users only.

2 Send Data

This command is used to send parameter or screen data. This command should only be used in conjunction with **Send Command** above. For experienced users only.

3 Get Model Info

4 Get Stored Settings

5 Store Settings

6 Store Icon Bank

7 Store Image

The commands above are issued by the NCD Image Loader Utility only. The protocol for using this command is complex and will remain undocumented.

GetReply

Some commands report ASCII character code 85 back to the host computer to tell your program that the display has completed the last command sent to it. The **GetReply** function is used to stall your program until the display has completed its drawing operations. This function is ONLY recommended for some commands, look for the **GetReply** function call as the last line of the programming examples.

GetReply is NOT required for proper operation of the display. A two-second delay in your program can be used to stall your program to ensure all graphic operations have completed. **GetReply** does allow your program to run a little faster because most drawing operations can be completed in less than 2 seconds.

Visual Basic Programming Examples

Many Visual Basic 6 programming examples are provided in the following pages to assist in the development of software for controlling this display controller. Additional source code can be found on our web site at www.controlanything.com.

Sample Code: Controlling the Display

```
Public Sub Test2Way()  
    MSCOMM1.Output = Chr$(254) 'Enter Command Mode  
    MSCOMM1.Output = Chr$(0) 'Send Test-2-Way Command  
    GetReply 'Wait for ASCII 85 from Controller  
End Sub  
  
Public Sub CMD(Byte)  
    MSCOMM1.Output = Chr$(254) 'Enter Command Mode  
    MSCOMM1.Output = Chr$(1) 'Issue Hardware Command to Display  
    MSCOMM1.Output = Chr$(Byte) 'Send Hardware Command Byte  
End Sub  
  
Public Sub DAT(Byte)  
    MSCOMM1.Output = Chr$(254) 'Enter Command Mode  
    MSCOMM1.Output = Chr$(2) 'Issue Hardware Data to Display  
    MSCOMM1.Output = Chr$(Byte) 'Send Hardware Data Byte  
End Sub  
  
Public Function GetReply() 'Get a Byte of Data from the Displays  
    'Wait for a Byte to Be Received  
    Do Until Form1.MSCOMM1.InBufferCount >= 1  
        TimeOutCounter = TimeOutCounter + 1 'Start the Counter  
        If TimeOutCounter >= 90000 Then 'If the Counter Times Out  
            'The Device Did NOT Respond  
            Debug.Print "Device Did Not Respond."  
            GoTo Escape 'Exit the Routine  
        End If  
        DoEvents 'Allow Windows to Service Other Tasks  
    Loop  
    DataIn = Asc(Form1.MSCOMM1.Input) 'Read the Serial Port  
    If DataIn = 85 Then 'If 85 Is Received  
        Debug.Print "OK. "; DataIn 'Device Responded with OK  
    Else  
        'Device Responded with Alternate Value  
        Debug.Print "Device Returned: "; DataIn  
    End If  
Escape:  
    GetReply = DataIn 'GetReply Holds Received Data if Any  
    For Dela = 0 To 63: DoEvents: Next Dela  
End Function
```

The Noritake –800 Series Display Controller Command Set

Recalling Stored Images

8 Copy Stored Image to Screen

This command copies an image stored in non-volatile EEPROM to the display screen. Images can be drawn using your favorite paint program and stored into the controller using the NCD Image Loader Utility.

To successfully copy an image, you will need to supply this command with an image number. The image number tells the controller which stored image to copy to the display screen. The maximum available image number is determined by how much memory is installed on your display controller. When in doubt, use the **GetFrames** command to ask the controller how many images are available. Note that you can also use the NCD Image Loader Utility to find out how many images may be stored in the display controller. Simply click on the **Hardware Settings** button. The maximum image number is shown in the lower right corner of the window. Also note that the startup screen is ALWAYS image 0.

Images are stored in EEPROM as 128x64 or 256x32 images. Though the display area of the VF screen may be smaller, the virtual screen area will be the size of the stored image. You can then use the scroll command to view the rest of the virtual screen area. For instance, the Noritake GU160X16-800 can hold a screen that is 256x32, but it can only display a screen that is 160x16. The scroll commands must be used to view the rest of the 256x32 image.

This command also requires that you specify a layer to hold the graphic image. The display has 2 layers, each capable of holding a separate image, one on top of the other. Valid layer values are 0 or 1.

Once drawing operations have completed, the display controller will send ASCII character code 85 back to the host computer.

NEVER SEND DATA TO THE CONTROLLER WHILE DRAWING OPERATIONS ARE IN PROGRESS. DOING SO MAY HANG THE DISPLAY CONTROLLER CPU.

Sample Code: Recalling Stored Images

```
'For Use with the DMF50427 ONLY
Public Sub CopyStoreImageToScreen(Image, Layer)
  MSCmm1.Output = Chr$(254) 'Enter Command Mode
  MSCmm1.Output = Chr$(8) 'Copy Store Image to Screen Command
  MSCmm1.Output = Chr$(Image) 'Stored Image to Copy
  MSCmm1.Output = Chr$(Layer) 'Graphic Layer to Copy Image To 0/1
  GetReply 'Wait for Display to Finish Command
End Sub
```


The Noritake –800 Series Display Controller Command Set

NCD Graphical Text & Scrolling

9 Horizontal Scrolling

It is possible to scroll the entire contents of the screen horizontally. Simply specify a horizontal scroll position of 0-255, the display controller will limit the user input to a valid range depending on which screen is being used.

10 Shift Up

Vertically scrolls the contents of the display up 1 pixel.

11 Shift Down

Vertically scrolls the contents of the display down 1 pixel.

12 Graphical Text

This text command is used to draw text using the NCD Graphical Character Generator. Up to eight 6x8 character fonts may be displayed using this command (fonts are changed using the **SetFont** command below). Fonts are stored as Icon banks 1-8 using the NCD Image Loader Utility. The source file for fonts can be found in the Graphics directory of the NCD Image Loader Utility. Look for the file, ICONS.BMP. This command allows you to set the X and Y start location of text. Up to 84 characters may be stored in the display controllers receive buffer. Additional characters beyond 84 will be ignored. Text will begin drawing on the graphic layer of the screen when ASCII character code 255 is received by the controller. When drawing operations have completed, the controller will send ASCII character code 85 back to the host computer. NEVER SEND DATA TO THE CONTROLLER WHILE DRAWING OPERATIONS ARE IN PROGRESS. DOING SO MAY HANG THE DISPLAY CONTROLLER CPU.

13 Set Font

The **SetFont** command is used to change text fonts. Up to 8 fonts may be displayed on the screen at one time. This command only affects the **Graphical Text** command and does not have any affect on displays that have a hardware character generator. This command has 2 parameters, Font and Kerning. Font is a value of 1 to 8, which directly corresponds to Icon banks 1 to 8 stored using the NCD Image Loader Utility. Kerning sets the character width, in this case, 6x8. The kerning for this display controller should always be set to 6. Other kerning options may become available in later versions of the firmware. The effects of this command will not be seen until the next time the **Graphical Text** command is issued.

Sample Code: NCD Graphical Text & Scrolling

```
Public Sub HorizontalScroll(HPOS)
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(9) 'Horizontal Scroll Command
  MSComml.Output = Chr$(HPOS) 'Horizontal Position to Scroll To
End Sub

Public Sub ShiftUp()
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(10) 'Vertical Shift Up Command
End Sub

Public Sub ShiftDown()
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(11) 'Vertical Shift Down Command
End Sub

Public Sub GraphicalText(Xpos, Ypos, Text$)
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(12) 'NCD Graphical Text Command
  MSComml.Output = Chr$(Xpos) 'X Start Position of Text
  MSComml.Output = Chr$(Ypos) 'Y Start Line of Text
  MSComml.Output = Text$ 'Text to Send, 84 Characters MAX
  MSComml.Output = Chr$(255) 'Terminate Function and Draw Text
  GetReply 'Wait for Display to Finish Command
End Sub

Public Sub SetFont(IconBank)
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(13) 'Set Font Command
  MSComml.Output = Chr$(IconBank) 'Select Icon Bank 1-8 for Text Font
  MSComml.Output = Chr$(6) 'Set Kerning Width to 6
End Sub
```

The Noritake –800 Series Display Controller Command Set

Pasting Icons on the Display Screen

14 Paste Icon

The **Paste Icon** command is one of the most powerful commands offered by our display controller. An Icon is simply a small graphical image. The display controller is capable of storing 3,076 icons in non-volatile EEPROM. Icons are often used for buttons, small logos, and even small animation. Icons bring the display to life by graphically updating user-selected portions of the graphical display screen. Icon graphics are stored under the file name "ICONS.BMP" located in the Graphics folder, which is located in the NCD Image Loader folder. This file may be manipulated using your favorite paint program. Once you have completed changes to the file, use the NCD Image Loader Utility to store your icons in the display controller. Then, use the **Paste Icon** command to draw your icons on the screen.

The **Paste Icon** command has 4 parameters outlined below:

BANK is a value from 1 to 16 and directly corresponds to the Banks shown in the ICONS.BMP file. A value outside the range of 1 to 16 will be truncated to the highest or lowest available bank number.

ICON specifies an individual icon to paste. Icons are numbered within each bank. Icon 0 is always in the upper left corner of each bank. Icons are numbered from left to right, and from top to bottom. Some icon banks store more icons than others. The controller will limit the user-input to the maximum number of available icons for the selected bank.

XPOS Specifies the horizontal X Position for the upper left corner of the icon. Valid ranges are from 0 to 255. Ranges outside the display range will cause the icon to wrap around to the left side of the screen.

YPOS Specifies the vertical LINE (8-pixel intervals). Valid ranges are from 0 to 7. Values outside the display range will truncate the icon.

15 Reserved for Future Expansion

16 Clear Text

These displays do NOT have a hardware character generator. This function will return ASCII character code 85. The display will be unaffected.

17 Clear Graphic

Clears the Display Screen

18 Set Bright

Sets the Brightness Level of the Vacuum Florescent display. Valid ranges are 0 (dimmiest) to 15 (brightest).

Sample Code: Pasting Icons

```
Public Sub PasteIcon(Bank, Icon, Xpos, Ypos)
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(14) 'Paste Icon Command
  MSComml.Output = Chr$(Bank) 'Select an Icon Bank
  MSComml.Output = Chr$(Icon) 'Select an Icon to Paste
  MSComml.Output = Chr$(Xpos) 'X Start Position of Icon
  MSComml.Output = Chr$(Ypos) 'Y Start Line of Icon
  GetReply 'Wait for Display to Finish Command
End Sub

Public Sub ClearScreen()
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(17) 'Send Clear Screen Command
  GetReply 'Wait for Display to Finish Command
End Sub

Public Sub SetBright(Level)
  MSComml.Output = Chr$(254) 'Enter Command Mode
  MSComml.Output = Chr$(18) 'Send Backlight Brightness Command
  MSComml.Output = Chr$(Level) 'Brightness: Level=0;DIM 15;Bright
End Sub
```

The Noritake –800 Series Display Controller Command Set

Miscellaneous Commands

19 Get Frames

Asks the display controller how many images can be stored in EEPROM. Frame 0 is always the startup screen. The more memory installed, the larger the number of available frames.

20 Full Frame Image Transfer

It may be necessary to upload image data directly to the full display buffer (even the area outside the viewing area) from the RS-232 serial port. The Full Frame Image Transfer command allows you to do this with minimal programming. The examples shown fill the screen (including the virtual display area outside the viewing area) by turning on all pixels. A byte value of 0 will clear the display. Try different byte values to explore this command further. The image will draw beginning in the upper-left corner, ending in the lower-right corner. For best results, 115.K baud operation will yield the highest screen refresh rates.

21 Screen Size Image Transfer

It may be necessary to upload image data directly to the display screen from the RS-232 serial port. This command is used to ONLY update the viewing area beginning in the upper left corner of the screen. The Screen Size Transfer command allows you to do this with minimal programming. The examples shown fill the screen by turning on all pixels. A byte value of 0 will clear the display. Try different byte values to explore this command further. The image will draw beginning in the upper-left corner, ending in the lower-right corner. For best results, 115.K baud operation will yield the highest screen refresh rates.

Sample Code: Miscellaneous Commands

```
Public Function GetFrames()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(19) 'Paste Icon Command  
    GetFrames = GetReply 'Report Available Frames Back to User  
End Sub  
  
'Full Frame Image Transfer (All Models)  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(20) 'Full Frame Image Transfer  
    For n = 1 to 1024  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU128x32 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 512  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU128x64 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 1024  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU160x16 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 640  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU160x32 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 320  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU192x16 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 384  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub  
  
'For GU240x64 ONLY  
Public Sub ScreenSize()  
    MSComml.Output = Chr$(254) 'Enter Command Mode  
    MSComml.Output = Chr$(21) 'Screen Size Image Transfer  
    For n = 1 to 1024  
        Byte = 255  
        MSComml.Output = Chr$(Byte) 'Data to Send to Display  
    Next n  
End Sub
```