

Noritake VFD-AppNote-01 (#27970)

Warning: Always double check your wiring before powering up the VFD. The wiring is different from most other displays!

General Description

The Noritake GU112X16G-7002 VFD is a feature-rich display module providing many functions that are much easier to use on the BASIC Stamp[®], SX or Propeller microcontrollers when you have the details necessary to utilize them. This AppNote is the first of several AppNotes to cover more details of various functions of this display. There is an accompanying source code file for the BS2 which is what this AppNote will explain. Each function covered will be in the order they are presented in that source code file. The concepts and details here apply to most other microcontrollers as well. All notation for values will be shown in Hexadecimal, although you could easily use Decimal or even Binary if you wanted.

Features Covered

The features covered in this AppNote are Display On/Off Control, Screen-Saver Functions, Display Blinking, Reverse Display, Brightness Control and Font Magnification. It would help to have the source code handy when following through this document as we will not be including full code. Instead relevant lines from the source code will be referred to in this document.

Display On/Off Control

One of the features of this VFD is the ability to power off the main display. This may not seem significant until you consider that a VFD consumes a fair amount of current when on. In this case up to 260 mA when the entire display is lit up (all pixels on). However by turning the display off, current consumption drops down to ~26 mA. The Display On/Off function is directly related to the Screen-Saver Function covered next, in that the same command string is used, but the parameter is different.

To turn the display On/Off you need to send the following command string:

Note: VFD is a PIN declaration and Baud is a Baud Rate constant used through this document.

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, x]
```

The 'x' is the parameter and will be a \$00 to turn the display off and a \$01 to turn the display on. An interesting thing you can do (as shown in the demo code) is to redraw the display while it is off and then turn it back on. Also notice the effect of the display powering on. It appears to almost fade in.

Turn Display Off:

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $00]
```

Turn Display On:

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $01]
```

Screen-Saver Functions

The Screen-Saver Function is similar to the Display On/Off Function above except that instead of powering the display on or off you are activating other functions using the same command string.

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, x]
```

Again 'x' is the parameter and a value of \$02 will turn all pixels on the display off, effectively blanking the display. A value of \$03 will turn all pixels on, effectively lighting up the whole display, and a value of \$04 will cause the display to blink between normal display (whatever is shown) and a reverse version of that at ~2 second intervals. These functions remain in effect until disabled by a Display On command.

Turn all pixels off (blank display):

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $02]
```

Turn all pixels on:

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $03]
```

Alternately switch display between normal and reverse modes:

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $04]
```

Disable Screen-Saver Mode (Display On):

```
SEROUT VFD, Baud, [$1F, $28, $61, $40, $01]
```

Display Blinking

Blinking of the display could easily be done by alternately drawing and erasing the screen. That is both memory (code-space) consuming and processor intensive (requires the controller to do the work). This VFD has a function that blinks the display for you, and offers several options for blinking. The format of this command is as follows.

```
SEROUT VFD, Baud, [$1F, $28, $61, $11, p, t1, t2, c]
```

The 'p' is the pattern. A value of \$00 is Normal Display, \$01 is Blink between Normal/Blank Display and \$02 is Blink between Normal/Reverse Display. The normal display time and alternate (Blank/Reverse) time are specified by 't1' and 't2' and can range from 1 to 255. The higher the value, the longer the delay, with 255 being ~4 seconds. The value of 'c' is the number of repetitions and can be from 1 to 255.

Note: The VFD buffers most commands, so if you specify multiple repetitions you need a PAUSE value long enough to last through the time it takes to complete the blinking. If you don't you might send too much data before the VFD has a chance to get to it. Notice in the sample code that a PAUSE 4000 (4 seconds) is more than ample time to allow the 4 repetitions to occur for each mode.

The FOR...NEXT loop is just cycling through both modes by setting the pattern (p) and also basing the time (t1/t2) on that value as well. So the first pass through the loop pattern 1 will be used and 't1' will be equal to 20 (1 * 20) and t2 will be equal to 10 (1 * 10). The second pass through the loop pattern 2 will be used and 't1' will be equal to 40 (2 * 20) and 't2' will be equal to 20 (2 * 10).

Display Reverse (Reverse Mode)

Note: This command does not affect the current screen information already displayed.

The Reverse Function allows you to write new information to the screen in reverse mode. This could be good for a status indication where the main text is normal but a value out of range is in reverse. It could also for work a menu system to highlight a current selection. The command is formatted as follows.

```
SEROUT VFD, Baud, [$1F, $72, x]
```

A value of \$01 for 'x' will turn Reverse Mode On. A value of \$00 for 'x' will turn Reverse Mode Off.

Turn Reverse Mode On:

```
SEROUT VFD, Baud, [$1F, $72, $01]
```

Turn Reverse Mode Off:

```
SEROUT VFD, Baud, [$1F, $72, $00]
```

Brightness Control

One neat thing about this VFD is the ability to adjust the brightness of the display. This is especially good for applications where the display will be used at night and you need to dim the display. There are 8 levels of brightness. The command is formatted as follows:

```
SEROUT VFD, Baud, [$1F, $58, x]
```

The value of 'x' can be from 1 to 8 and determines the brightness as follows:

Value of x	Brightness
1	12.5%
2	25%
3	37.5%
4	50%
5	62.5%
6	75%
7	87.5%
8	100%

The following code sets the brightness to 50%.

```
SEROUT VFD, Baud, [$1F, $58, $04]
```

Note: The Initialize Display command used in the Initialization section of the code always resets the brightness back to 100%. That command reset the default settings of the VFD, but does clear the input buffer (the contents/commands remain). The Initialize Display command is as follows:

```
SEROUT VFD, Baud, [$1B, $40]
```

Font Magnification

This command will directly affect the size of characters printed following the command. Characters may be expanded in the x direction up to 4X and in the y direction up to 2X. You may expand only 1 direction at a time or both. The command is formatted as follows:

```
SEROUT VFD, Baud, [$1F, $28, $67, $40, x, y]
```

The value of 'x' can be from 1 to 4 and the value of y can be from 1 to 2. A value of 1 means no magnification. The demo code shows several variations of using this command and the results produced on the display.

This example will double the size of the characters in both directions:

```
SEROUT VFD, Baud, [$1F, $28, $67, $40, $02, $02]
```

This example will double the height of the characters while leaving the width unaffected:

```
SEROUT VFD, Baud, [$1F, $28, $67, $40, $01, $02]
```

We hope this AppNote provides you will a good explanation of how the functions covered work as well as providing example source code. More AppNotes will follow to cover other functions in the near future, such as Bitmap Mode, Custom Characters, User Windows, etc.