Generic API for Graphics LCD (Glyph)

v1.00

www.teamfdi.com

Table of Contents

1.0 Objective

The Generic API for Graphics LCD (nicknamed "Glyph") is a reusable software design to support multiple projects where the LCD size and resolution may change, but the basic feature set will stay consistent. Glyph is a standard library interface for use with any graphic LCD. The following components are defined:

- Glyph API
- Register based control interface for
  - LCD display
  - Font/Graphics selection
  - Alphanumeric output
- Device driver implementation

2.0 Library Calls

The Glyph library has a high level interface for controlling the display. These commands are are documented in the following pages. All functions return an error code of type T_glyphError, which is 0 when there are no errors, or a non-zero value of the error type.

T_glyphError GlyphOpen(T_glyphHandle *aHandle, int32_t aAddress);

Descriptions:

Opens a Glyph session and returns its handle.

Parameters:

T_glyphHandle *aHandle – Handle of Glyph session
int32_t aAddress – Address of LCD screen

T_glyphError GlyphClose(T_glyphHandle *aHandle);

Descriptions:

Closes a previously opened Glyph session.

Parameters:

T_glyphHandle *aHandle – Pointer to Glyph session to close.

T_glyphError GlyphGetStatus(T_glyphHandle aHandle, T_glyphStatusBits *aStatus);

Descriptions:

The status of a display can be busy, ready, or its buffer full. This routine checks the display's condition. Since all Glyph functions check the status of the display, checking this routine is usually not necessary.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
T_glyphStatusBits *aStatus – Returned status bits.

T_glyphError GlyphSetX(T_glyphHandle aHandle, uint32_t aX);

Descriptions:

Sets the value of the X position in the given Glyph Handle. This position is used as the left position of the next character on the screen.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
uint32_t aX – X position in pixels.

T_glyphError GlyphSetY(T_glyphHandle aHandle, uint32_t aY);

Descriptions:

Sets the value of the Y position in the given Glyph Handle. This position is used as the top position of the next character on the screen.

NOTE: Some displays may round the Y position to a multiple for memory display reasons.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

uint32_t aY – Y position in pixels.

T_glyphError GlyphSetXY(T_glyphHandle aHandle, uint32_t aX, uint32_t aY);

Descriptions:

Sets the value of the X and Y position in the given Glyph Handle. This position is used as the left and top position of the next character on the screen.

NOTE: Some displays may round the Y position to a multiple for memory display reasons.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

uint32_t aX – X position in pixels.

uint32_t aY – Y position in pixels.

T_glyphError GlyphGetXY(T_glyphHandle aHandle, uint32_t *aX, uint32_t *aY);

Descriptions:

Gets the value of the X and Y positions in the given Glyph session. The filled values are the current top left position of the next character to be placed onto the screen.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

uint32_t *aX – Pointer to X position in pixels.

uint32_t *aY – Pointer to Y position in pixels.

T_glyphError GlyphSetFont(T_glyphHandle aHandle, T_glyphFont aFont);

Descriptions:

Set the selected font to be used as the next font for the next characters drawn.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphFont aFont – Font ID to use

T_glyphError GlyphGetFont(T_glyphHandle aHandle, T_glyphFont *aFont);

Descriptions:

Retrieve the enumeration value for the currently selected font.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphFont *aFont – Font ID being used

T_glyphError GlyphDrawCommand(T_glyphHandle aHandle, T_glyphDrawMode aMode);

Descriptions:

Causes a draw command to be performed on the LCD display. Draw commands are explained in *3.2 Draw Commands.*

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphDrawMode *aFont – Font ID being used

T_glyphError GlyphChar(T_glyphHandle aHandle, uint32_t aChar);

Descriptions:

Draws the given character in the currently chosen font at the current X and Y starting pixel on the LCD Screen.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
uint32_t aChar – The given character.

T_glyphError GlyphString(T_glyphHandle aHandle, uint8_t * aString, uint32_t aLength);

Descriptions:

Draws the given character string in the currently chosen font at the current X and Y top left starting pixel on the LCD Screen. You must count the number of characters you wish to put to the LCD screen and enter that number in the aLength variable.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
uint8_t * aString – The given character string.
uint32_t aLength -- The number of characters in the string to display on the LCD.

T_glyphError GlyphGetVersionInfo(T_glyphHandle aHandle, T_glyphVersionInfo *aInfo);

Descriptions:

Fills the given T_glyphVersionInfo with version data.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
T_glyphVersionInfo *aInfo – Info structure to fill

T_glyphError GlyphClearScreen(T_glyphHandle aHandle);

Descriptions:

Clears the LCD Screen to white.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphInvertScreen(T_glyphHandle aHandle);

Descriptions:

Inverts every pixel on the screen from white to dark and dark to white.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphNormalScreen(T_glyphHandle aHandle);

Descriptions:

Sets up the normal LCD Screen right after initialization

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphSleep(T_glyphHandle aHandle);

Descriptions:

Puts the LCD in a low power blank screen.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphWake(T_glyphHandle aHandle);

Descriptions:

Takes the LCD out of low power sleep mode and redraws the current image.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphDrawTestPattern(T_glyphHandle aHandle);

Descriptions:

Draws a test image onto the LCD Screen.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

T_glyphError GlyphDrawBlock(T_glyphHandle aHandle, uint32_t aX1, uint32_t aY1, uint32_t aX2, uint32_t aY2);

Descriptions:

Draws a set of dark pixels from the X and Y position to the X2 and Y2 position.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
uint32_t aX1 – left pixel of block region
uint32_t aY1 – top pixel of block region
uint32_t aX2 – right pixle of block region
uint32_t aY2 – bottom pixel of block region

T_glyphError GlyphEraseBlock(T_glyphHandle aHandle, uint32_t aX1, uint32_t aY1, uint32_t aX2, uint32_t aY2);

Descriptions:

Clears to a set of white pixels from the X and Y position to the X2 and Y2 position.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
uint32_t aX1 – left pixel of block region
uint32_t aY1 – top pixel of block region
uint32_t aX2 – right pixle of block region
uint32_t aY2 – bottom pixel of block region

T_glyphError GlyphSetContrast(T_glyphHandle aHandle, int32_t nContrast);

Descriptions:

Sets the contrast of the LCD Display. Any Positive number is ok. Check the documentation of your LCD and study the LCD Driver Code for correct values. If the number is too large the contrast will not be changed. When in doubt, use the power up setting.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session
int32_t nContrast – A contrast setting value

T_glyphError GlyphSetContrastBoost(T_glyphHandle aHandle, uint8_t cContrastBoost);

Descriptions:

Sets the boost for the contrast of the LCD Display. Any Positive number is ok. Check the documentation of your LCD and study the LCD Driver Code for correct values. If the number is too large the contrast boost will not be changed. When in doubt, use the power up setting.

Parameters:

T_glyphHandle aHandle – Handle to Glyph Session

uint8_t cContrastBoost – Contrast Booster setting value

3.0 Register Interface to Device Driver

The Glyph code uses a register based interface to the device driver that handles the specific hardware display in the following form. Device driver functions are executed by using the two Glyph library commands.

```
T_glyphError GlyphWrite(T_glyphHandle aHandle, unsigned aRegister, unsigned aValue);
T_glyphError GlyphRead(T_glyphHandle aHandle, unsigned aRegister, unsigned *aValue);
```

In fact, if code space is tight, the above two commands plus GlyphOpen() and GlyphClose() are the only commands needed to control a Glyph API device.

The following is a list of all the registers known by a Glyph Device:

Name	Index	Bits:	Usage:
GLYPH_STATUS	0x00	0	Not used
		1	**GLYPH_BUSY** This flag is set to 1 when the front panel is busy (such as drawing) and set to 0 when ready for drawing commands.
		2:7	Not used
GLYPH_CHAR_X	0x01	0:7	Pixel X position in which to draw the next character. Write to change the next X position. Read to get the current X position.
GLYPH_CHAR_Y	0x02	0:7	Pixel Y position in which to draw the next character. Write to change the next Y position. Read to get the current Y position.
GLYPH_FONT	0x03	0:7	Font to use. The list of fonts depends on the implementation. Writing a value of 0-255 to change the selected font. Writing an invalid font number will go back to the default of 0. Reading will return the font number being used.
GLYPH_DRAW_CHAR	0x04	0:7	Write a value to output an ASCII character at the current GLYPH_CHAR_X and GLYPH_CHAR_Y position. The X position is incremented to the next X position automatically. If the character cannot fit, the Y position is incremented by the font height and begins at the next line automatically. Must only be written to when not busy (see GLYPH_BUSY).
GLYPH_DRAW_CMD	0x05	0:7	*See Draw Commands below.* Must only be written to when not busy (see GLYPH_BUSY).
GLYPH_CHAR_ERASE	0x06	0:7	Must only be written to when not busy (see GLYPH_BUSY).
GLYPH_INVERT_CHAR	0x07	0:7	Must only be written to when not busy (see GLYPH_BUSY).
GLYPH_API_MAJOR_VERSION	0x08	0:7	Major version number of API. Should be 1.
GLYPH_API_MINOR_VERSION	0x09	0:7	Minor version number of API. Should be 0.
GLYPH_IMPLEMENTATION_ID	0x0A	0:7	Unique id for this implementation. Unique for each project. Should not be zero.
GLYPH_IMPL_MAJOR_VERSION	0x0B	0:7	Implementation's major version number.
GLYPH_IMPL_MINOR_VERSION	0x0C	0:7	Implementation's minor version number.
GLYPH_CHAR_X2	0x0D	0:7	Pixel X2 position in which to draw a rectangle.
GLYPH_CHAR_Y2	0x0E	0:7	Pixel Y2 position in which to draw a rectangle.
GLYPH_RESOLUTION_X	0x0F	0:7	Number of pixels wide
GLYPH_RESOLUTION_Y	0x10	0:7	Number of pixels high
GLYPH_GRANULARITY_X	0x11	0:7	Number of pixels per valid X position (usually 1)
GLYPH_GRANULARITY_Y	0x12	0:7	Number of pixels per valid Y position (usually 8)
GLYPH_FRAME_RATE	0x13	0:7	Value in Hz.
GLYPH_CONTRAST	0x14	0:7	Contrast setting of 0 to 255.
GLYPH_CONTRAST_BOOST	0x15	0:7	Booster setting (display specific)

3.1. Drawing Characters

Text can be drawn to the screen, but must be drawn character by character. A character is drawn by writing the Y position into GLYPH_CHAR_Y, then writing the X position into GLYPH_CHAR_X. Finally, the character is drawn by writing the character to GLYPH_DRAW_CHAR. The X position is automatically moved to the right, possibly going off the right edge (it does not word wrap). Writing another character to GLYPH_DRAW_CHAR will continue drawing characters to the right.

Writing the font's ID into GLYPH_FONT changes the character's appearance. Font IDs are unique per implementation.

3.2. Draw Commands

Writing a value to GLYPH_DRAW_CMD will do a variety of drawing and control functions. These functions will cause the GLYPH_BUSY flag to go to 1 during the drawing and then go to 0 when done. Commands received while GLYPH_BUSY is 1 are ignored.

Below is a list of drawing commands (possibly expanded with later versions):

Draw Command:	Value:	Action taken:
GLYPH_CMD_NOP	0x00	Take no action. Just use to delay.
GLYPH_CMD_SCREEN_CLEAR	0x01	Clear the whole screen.
GLYPH_CMD_SCREEN_INVERT	0x02	Show the screen in the inverse colors.
GLYPH_CMD_SCREEN_REGULAR	0x03	No longer invert the screen.
GLYPH_CMD_SCREEN_SLEEP	0x04	Put screen in sleep mode.
GLYPH_CMD_SCREEN_WAKE	0x05	Wake up the screen.
GLYPH_CMD_TEST_PATTERN	0x06	Show test pattern of grids on screen
GLYPH_CMD_DRAW_BLOCK	0x07	Draw Rectangular Block
GLYPH_CMD_ERASE_BLOCK	0x08	Erase Rectangular Block

Commands received outside this list perform no action.

3.3. Font Selection

The GLYPH_CMD selects which font is used from the following list. Please note that not all fonts may be supported on all targets and the default font may be different.

Font ID	Name	Height	Width (max)	ASCII Range	Description
1	x5x7	7 pixels	5 pixels	0x00-0x7E	Small fixed font
2	x6x13	13 pixels	6 pixels	0x00-0x7E	Small fixed font
3	rom8x8	8 pixels	8 pixels	0x00-0xFF	BIOS fixed font
4	rom8x16	16 pixels	8 pixels	0x00-0xFF	BIOS fixed font
5	helvr10	10 pixels	9 pixels	0x20-0x7E	Helvetica proportional font.
6	winfreesystem	16 pixels	14 pixels	0x20-0xFF	FreeSystem proportional font.
7	droidsanr14	14 pixels	14 pixels	0x20-0x7E	Droid Sans serif proportional font
8	droidsanr36	32 pixels	35 pixels	0x20-0x7E	Droid Sans serif proportional font

3.4. Drawing Graphics with Decals

Non-character graphics are achieved by using decals. Decals are graphics stored in custom fonts (starting at ID 0x80) where each character is a single picture (similar to Wingbat fonts). By using the above draw and erase commands, a standard picture can be made to appear or disappear at specific locations.