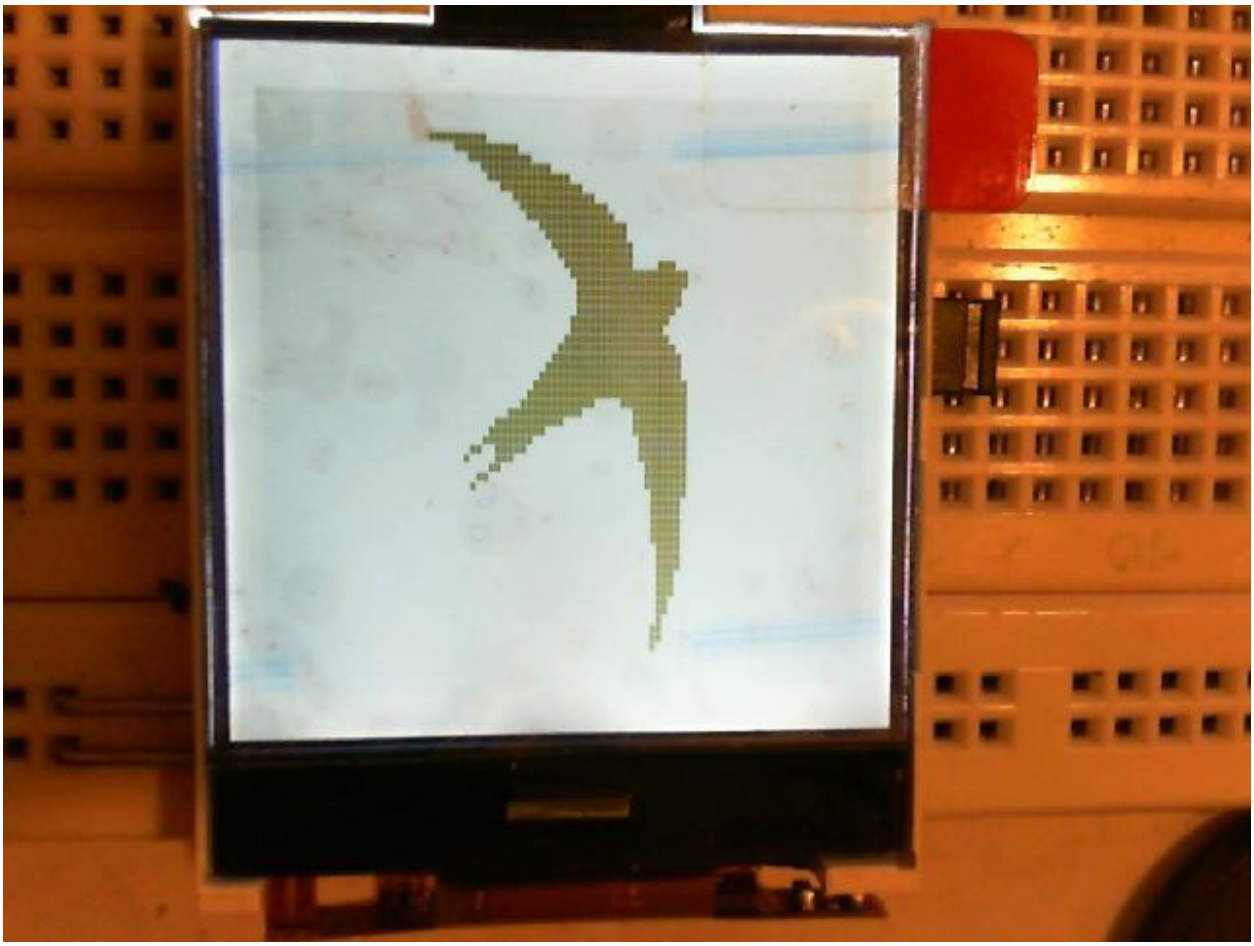
Arduino librería para el display RE9664WRF-004-I02



Indicé

El display

El display RE9664WRF de OKAYA es un display monocromático de 96x64 pixeles.

Cuenta con el controlador ST7579 que lleva un interfaz paralelo y un interfaz serial para la fácil conexión a un micro controlador. El siguiente documento describe una librería para Arduino para facilitar el uso de este display.

Especificaciones

1.1 Features

Item	Standard Value
Display Type	96*64 Dots
LCD Type	FSTN, Positive, Transflective, Extended Temp.
Driver Condition	LCD Module : 1/68Duty , 1/9Bias
Viewing Direction	6 O'clock
Backlight	White LED B/L
Weight	TBD
Interface	4-Line interface
Other(controller / driver IC)	ST7579
ROHS	THIS PRODUCT CONFORMS THE ROHS OF

1.2 Mechanical Specifications

Item	Standard Value	Unit
Outline Dimension	33.35(W)*41.64(L)*3.0(H)MAX	mm
Viewing Area	29.3 (W)* 29.9 (L)	mm
Active Area	27.34(W)*26.86(L)	mm
Dot Size	0.265(L)*0.400(W)	mm
Dot Pitch	0.285(L)*0.420(W)	mm

Note：For detailed information please refer to LCM drawing

1.3 Absolute Maximum Ratings

Item	Symbol	Condition	Min.	Max.	Unit
Power Supply Voltage	VDD	-	-0.3	3.6	V
LCD Driver Supply Voltage	V_{LCD}	-	-0.5	15.0	V
Operating Temperature	T_{OP}	-	-20	70	℃
Storage Temperature	T_{ST}	-	-30	80	℃
Storage Humidity	H_D	Ta＜60 ℃	-	90	%RH

SPI interfaz

El controlador ST7579 cuenta con un interfaz SPI que puede operar en un modo de 4 hilos o de 3 hilos.

La presente librería utiliza **solo el modo de 3 hilos**. Los señales y las conexiones al Arduino son las siguientes:

MOSI(SDA) SPI data conectarlo a MOSI del Arduino (pin 11 en el nano)

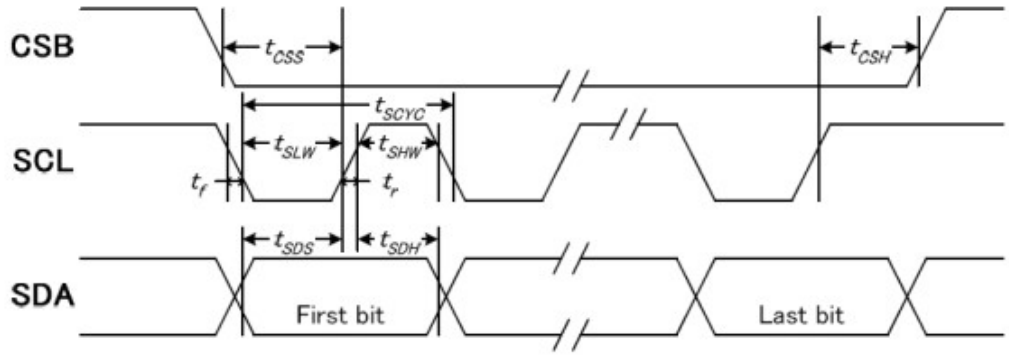
SCK(SCL) SPI clock, conectarlo a SCK del Arduino (pin 13 en el nano)

CS(CSB) SPI chip select, conectarlo a un pin digital del Arduino

RESET Display reset, conectarlo a un pin digital del Arduino

(En el modo de 4 hilos se necesita adicionalmente un señal denominada A0. La presente librería no esta soportando este modo!)

SERIAL INTERFACE (3-Line Interface)



(VDD = 3.3V , Ta =-30~85°C)

Item	Signal	Symbol	Condition	Min.	Max.	Unit
Serial clock period	SCLK	tSCYC		120	—	ns
SCLK "H" pulse width		tSHW		60	—	
SCLK "L" pulse width		tSLW		60	—	
Data setup time	SDA	tSDS		20	—	
Data hold time		tSDH		10	—	
CSB-SCLK time	CSB	tCSS		20	—	
CSB-SCLK time		tCSH		130	—	

Funcionamiento

Texto

El display cuenta con una memoria de datos que representa la área visible del display. Esta memoria esta organizado en 10 paginas (pages) y 64 columnas. Cada pagina es una columna pequeña de 8 bits (pxeles) en orientación vertical. Si por ejemplo uno quiere mostrar el carácter ! (signo de exclamación) es necesario de escribir el valor 95 (5F hexadecimalmente) a la columna deseada.

95 dec = 1011111 binario.

O en orientación vertical:

1
1
1
1
0
1

como las caracteres no son todos tan delgado como este signo de exclamación el código total para este carácter es: `{0x00, 0x00, 0x5f, 0x00, 0x00, 0x00}`

esto significa que el signo esta pintada en la tercera columna de un total de 6 columnas.

El O mayúscula tiene el siguiente código: `{0x3e, 0x41, 0x41, 0x41, 0x3e, 0x00}`

en orientación vertical y representación binario:

011100
100010
100010
100010
100010
011100

Puedes ver el O?

Gráficas

Pintar gráficas funciona en la misma manera como testo. Solo se extiende sobre muchas paginas y columnas. Por ejemplo la gráfica de una cara feliz esta representado por el siguiente código:

```
0x00, 0x00, 0x00, 0x00, 0xC0, 0x60, 0x20, 0x10, 0x08, 0x08, 0x04, 0x04, 0x04, 0x00, 0x00, 0x00,
0x00, 0x04, 0x04, 0x04, 0x04, 0x08, 0x08, 0x18, 0x30, 0x60, 0xC0, 0x80, 0x00, 0x00, 0x00, 0x00,
0x00, 0xF8, 0x0E, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x1F, 0x1F, 0x1F, 0x1E, 0x00, 0x00,
0x00, 0x00, 0x38, 0x7C, 0x7C, 0x7C, 0x78, 0x00, 0x00, 0x00, 0x00, 0x01, 0x07, 0xFE, 0xF8, 0x00,
0x00, 0x1F, 0x78, 0xE0, 0x80, 0x00, 0x04, 0x04, 0x06, 0x3A, 0x70, 0xE0, 0xE0, 0xE0, 0xE0, 0xE0,
0xE0, 0x60, 0x60, 0x38, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0xE0, 0xF8, 0x7F, 0x1F, 0x00,
0x00, 0x00, 0x00, 0x01, 0x03, 0x07, 0x0E, 0x0C, 0x1C, 0x18, 0x38, 0x30, 0x30, 0x30, 0x30, 0x30,
0x30, 0x30, 0x30, 0x30, 0x38, 0x38, 0x1C, 0x1C, 0x0E, 0x07, 0x07, 0x03, 0x00, 0x00, 0x00, 0x00
```

Uno puede imaginarse que es bastante difícil de convertir un imagen a este colona de números. Pero la buena noticia es, que por este trabajo hay computadores!

En el Internet se encuentran varias programas para convertir imágenes en código para displays de LCD.

Un programa que nos parece bien para el uso con este librería es “LCD Assistant” que se puede bajar de la siguiente dirección: http://en.radzio.dxp.pl/bitmap_converter/LCDAssistant.zip

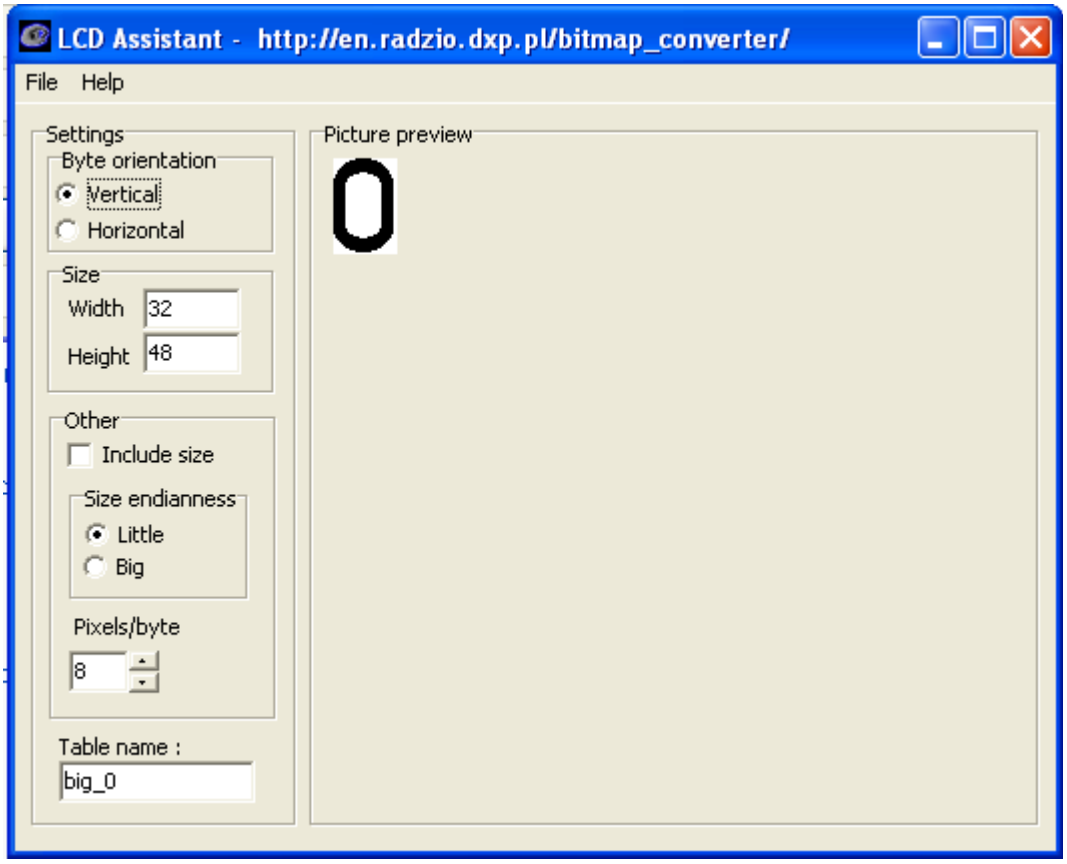
El uso es muy simple. Arranque LCD Assistant, y abre un imagen en formato BMP. Los ajustes deben ser:

Byte orientation: Vertical

Size endianess: Little

Pixels/byte: 8

Entra el tamaño del imagen y guarda el codigo de salida en un archivo de texto.



El archivo de salida para este gráfica se ve así:

```
//-----------------------------------------------------------------------------
// File generated by LCD Assistant
// http://en.radzio.dxp.pl/bitmap_converter/
//-----------------------------------------------------------------------------

const unsigned char big_0 [] = {
0x00, 0x00, 0xC0, 0xE0, 0xF0, 0xF8, 0xFC, 0xFC, 0xFE, 0xFE, 0xFE, 0xFF, 0xFF, 0x7F, 0x7F, 0x7F,
0x7F, 0xFF, 0xFF, 0xFE, 0xFE, 0xFE, 0xFC, 0xFC, 0xF8, 0xF0, 0xE0, 0xC0, 0x00, 0x00, 0x00, 0x00,
0xFC, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x0F, 0x03, 0x01, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x01, 0x01, 0x03, 0x0F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC, 0x00, 0x00,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00,
0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF8, 0xE0, 0xC0, 0xC0, 0x80, 0x80, 0x80, 0x00, 0x00, 0x00,
0x00, 0x80, 0x80, 0x80, 0xC0, 0xC0, 0xE0, 0xF8, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x3F, 0x00, 0x00,
0x00, 0x00, 0x01, 0x03, 0x07, 0x0F, 0x1F, 0x1F, 0x3F, 0x3F, 0x3F, 0x7F, 0x7F, 0x7F, 0x7F, 0x7F,
0x7F, 0x7F, 0x7F, 0x3F, 0x3F, 0x3F, 0x1F, 0x1F, 0x0F, 0x07, 0x03, 0x01, 0x00, 0x00, 0x00, 0x00
};
```

Ahora puedes copiar esto al archivo graphics.h y utilizar la gráfica con el nombre big_0. **Pero ojo,** es muy importante de ajuntar la palabra PROGMEM antes del signo = . La definición del array debe ser como así:

```
const unsigned char big_0 [] PROGMEM = {…};
```

La librería

Para facilitar el uso de este display microelectronicos dispone una librería para Arduino.

Con esta librería es muy simple de presentar textos y gráficas en la pantalla. La librería cuenta con

- un juego de caracteres de 5x7 pixeles
- un juego de números grandes de 32x48 pixeles

y adicionalmente con dos gráficas de muestra (un pájaro de 96x64 pixeles y una cara feliz de 32x32 pixeles)

El juego de caracteres esta ubicado en el archivo font5x7.h y los números grandes u los dos gráficas se encuentran en el archivo graphics.h

Instalación

Para instalar la librería con el sketch de ejemplo, simplemente copia la carpeta OkayaLCD en la carpeta libraries de tu instalación de Arduino. Ahora arranque Arduino y elige Archivo/Ejemplos/OkayaLCD/okatatest para cargar el sketch de ejemplo.

Métodos

void st7579::clear ()

Borra el display.

Solo borra el display, no la memoria de textos!

void st7579::clear_framebuffer ()

Borra la memoria de texto(framebuffer).

Llama esta método si un texto nuevo debe se pintado en la pantalla borrada.

void st7579::config_bias (uint8_t *bias*)

Adjuste el contraste del display.

Mira la hoja de datos del ST7579 para mayor información.

Un valor buena es 6.

void st7579::config_booster (uint8_t *be*, uint8_t *pc*)

Configuración del sistema de alimentación interna del ST7579

Mira la hoja de datos del ST7579 para mayor información.

void st7579::config_framerate (uint8_t *framerate*)

Configura la frecuencia en que la pantalla esta pintada.

Mira la hoja de datos del ST7579 para mayor información.

void st7579::config_startline (uint8_t *st*)

Ajusta la primera linea del display.

Mira la hoja de datos del ST7579 para mayor información.

void st7579::config_vlcd (uint8_t *vop*)

Configuración del sistema de alimentación interna del ST7579

Mira la hoja de datos del ST7579 para mayor información.

void st7579::display_allon ()

Pinta toda la pantalla en negro.

void st7579::display_invert ()

Pinta todo en inverso (blanco en fondo negro)

void st7579::display_normal ()

Reajustar el display al modo normal (negro en fondo blanco).

void st7579::display_off ()

Apaga el display.

void st7579::drawBitmap (const unsigned char * *data*, unsigned char *mx*, unsigned char *my*, char *xpos*, char *ypos*)

Pinta un imagen.

Es necesario de entregar el tamaño del imagen en pixeles con los parámetros mx y my

Parámetros:

data	Vector a los datos del imagen. (por ejemplo big_0)
mx	Ancho del imagen en pixeles.
my	Altura del imagen en pixeles.
xpos	La posición en X donde se inicio el imagen
ypos	La posición en X donde se inicio el imagen

Ejemplo para pintar la cara feliz en los coordinados 0,0 :

```
okaya.drawBitmap(Smily_32x32,32,32,64,s);
```

void st7579::go (uint8_t *x*, uint8_t *y*)

Mueva el origen para el siguiente imagen a los coordinados x,y

void st7579::init (byte *resetpin*, byte *cspin*)

Inicialisazion del display.

Parámetros:

resetpin	El pin del Arduino en donde esta conectado el señal RESET del display.
cspin	El pin del Arduino en donde esta conectado el señal CS del display.

void st7579::poweroff ()

Mete el display en un estado de ahorro de energía.

void st7579::poweron ()

Despierta el display del estado de ahorro de energía.

void st7579::puts (const char * *str*)

Escribe un texto. Se puede utilizar los siguientes combinaciones especiales:

\n para una nueva fila

\t para un espacio de 8 caracteres

\b para volver un carácter (borrar)

Parametros:

str	Vector al texto que debe ser escrito.

Ejemplo: `okaya.puts("Fila1\nFila2\tcon espacio");`

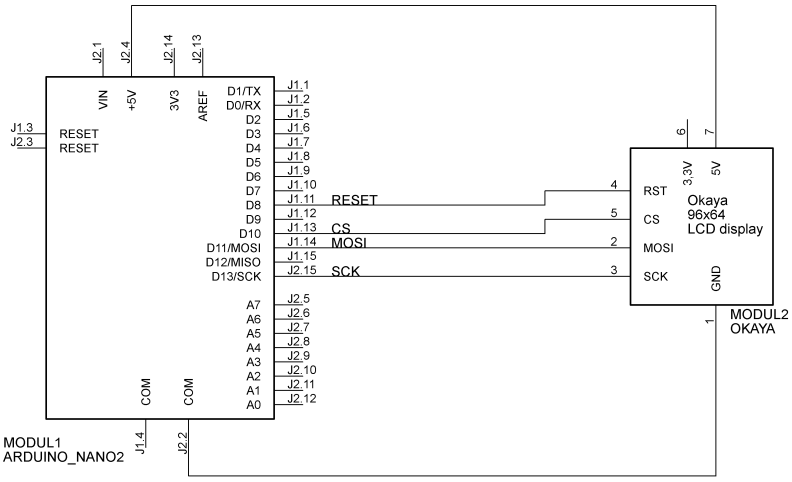
```
Fila1

Fila2     con espacio
```

void st7579::testpattern ()

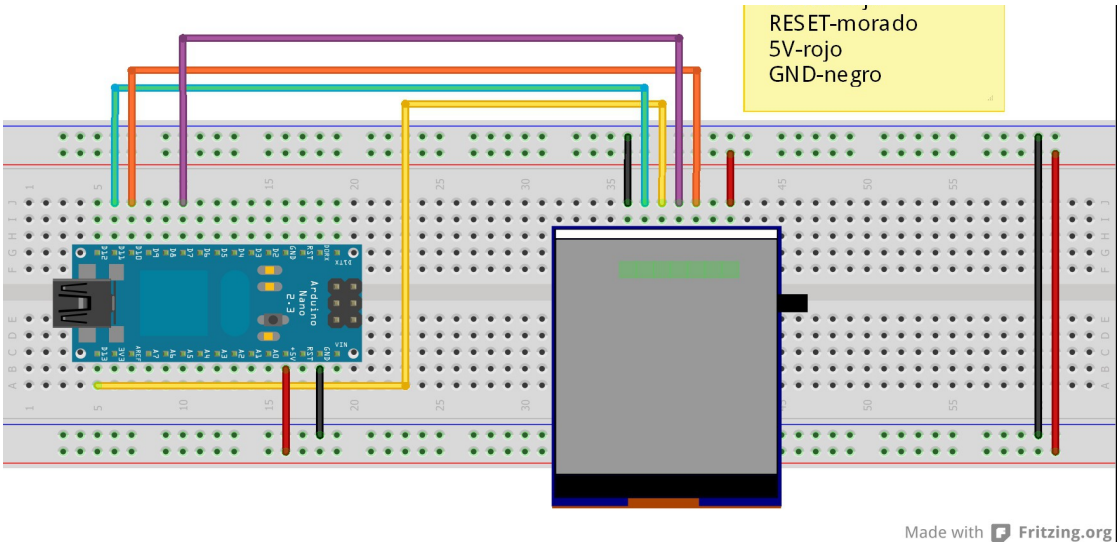
Pinta un imagen de prueba.

Esquemático



Las señales SCKy MOSI tienen que ser conectado como mostrado!

Las señales CS y RESET pueden ser conectado a otros pines del Arduino.



Referencias

- ST7579 http://www.tianma.com/web/uploads/controller/20080316012510_ST7579_V0.9a.pdf
- Okaya display http://www.graftec.com/images/files/re9664wrf-004-i02.pdf
- Arduino nano http://www.arduino.cc/en/Main/ArduinoBoardNano