emWin Display Driver GUIDRV_SPage

Supported hardware

Controllers

The driver works with the following display controller:

- Epson S1D15E05, S1D15E06, S1D15605, S1D15606, S1D15607, S1D15608, S1D15705, S1D15710, S1D15714, S1D15719, S1D15721
- Hitachi HD61202
- Integrated Solutions Technology IST3020
- New Japan Radio Company NJU6676
- Novatek NT7502, NT7534, NT7538, NT75451
- OriseTech SPLC502B
- Samsung KS0108, S6B0108, S6B0713, S6B0719, S6B0724, S6B1713
- Sino Wealth SH1101A
- Sitronix ST7522, ST7565, ST7567, ST7591
- Solomon SSD1303, SSD1305, SSD1306, SSD1805, SSD1815, SSD1821
- Sunplus SPLC501C
- UltraChip UC1601, UC1606, UC1608, UC1611, UC1701

Bits per pixel

The driver supports 1, 2 and 4 bpp resolutions.

Interfaces

The driver supports the indirect interface (8 bit) of the display controller. Parallel, 4-pin SPI or I2C bus can be used.

Color depth and display orientation

The driver can be used with different orientations and color depths. The following table shows the configuration macros which can be used to create and link the driver during the initialization:

Identifier	Color depth	Cache	Orientation
GUIDRV_SPAGE_1C0	1bpp	No	default
GUIDRV_SPAGE_OY_1C0	1bpp	No	Y axis mirrored
GUIDRV_SPAGE_OX_1C0	1bpp	No	X axis mirrored
GUIDRV_SPAGE_OXY_1C0	1bpp	No	X and Y axis mirrored
GUIDRV_SPAGE_OS_1C0	1bpp	No	X and Y swapped
GUIDRV_SPAGE_OSY_1C0	1bpp	No	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_1C0	1bpp	No	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_1C0	1bpp	No	X and Y swapped, X and Y axis mirrored
GUIDRV_SPAGE_1C1	1bpp	Yes	default

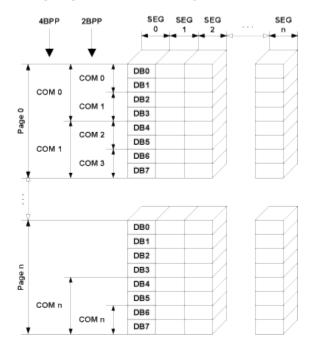|			
GUIDRV_SPAGE_OY_1C1	1bpp	Yes	Y axis mirrored
GUIDRV_SPAGE_OX_1C1	1bpp	Yes	X axis mirrored
GUIDRV_SPAGE_OXY_1C1	1bpp	Yes	X and Y axis mirrored
GUIDRV_SPAGE_OS_1C1	1bpp	Yes	X and Y swapped
GUIDRV_SPAGE_OSY_1C1	1bpp	Yes	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_1C1	1bpp	Yes	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_1C1	1bpp	Yes	X and Y swapped, X and Y axis mirrored
GUIDRV_SPAGE_2C0	2bpp	No	default
GUIDRV_SPAGE_OY_2C0	2bpp	No	Y axis mirrored
GUIDRV_SPAGE_OX_2C0	2bpp	No	X axis mirrored
GUIDRV_SPAGE_OXY_2C0	2bpp	No	X and Y axis mirrored
GUIDRV_SPAGE_OS_2C0	2bpp	No	X and Y swapped
GUIDRV_SPAGE_OSY_2C0	2bpp	No	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_2C0	2bpp	No	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_2C0	2bpp	No	X and Y swapped, X and Y axis mirrored
GUIDRV_SPAGE_2C1	2bpp	Yes	default
GUIDRV_SPAGE_OY_2C1	2bpp	Yes	Y axis mirrored
GUIDRV_SPAGE_OX_2C1	2bpp	Yes	X axis mirrored
GUIDRV_SPAGE_OXY_2C1	2bpp	Yes	X and Y axis mirrored
GUIDRV_SPAGE_OS_2C1	2bpp	Yes	X and Y swapped
GUIDRV_SPAGE_OSY_2C1	2bpp	Yes	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_2C1	2bpp	Yes	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_2C1	2bpp	Yes	X and Y swapped, X and Y axis mirrored
GUIDRV_SPAGE_4C0	4bpp	No	default
GUIDRV_SPAGE_OY_4C0	4bpp	No	Y axis mirrored
GUIDRV_SPAGE_OX_4C0	4bpp	No	X axis mirrored
GUIDRV_SPAGE_OXY_4C0	4bpp	No	X and Y axis mirrored
GUIDRV_SPAGE_OS_4C0	4bpp	No	X and Y swapped
GUIDRV_SPAGE_OSY_4C0	4bpp	No	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_4C0	4bpp	No	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_4C0	4bpp	No	X and Y swapped, X and Y axis mirrored
GUIDRV_SPAGE_4C1	4bpp	Yes	default
GUIDRV_SPAGE_OY_4C1	4bpp	Yes	Y axis mirrored
GUIDRV_SPAGE_OX_4C1	4bpp	Yes	X axis mirrored
GUIDRV_SPAGE_OXY_4C1	4bpp	Yes	X and Y axis mirrored
GUIDRV_SPAGE_OS_4C1	4bpp	Yes	X and Y swapped

GUIDRV_SPAGE_OSY_4C1	4bpp	Yes	X and Y swapped, Y axis mirrored
GUIDRV_SPAGE_OSX_4C1	4bpp	Yes	X and Y swapped, X axis mirrored
GUIDRV_SPAGE_OSXY_4C1	4bpp	Yes	X and Y swapped, X and Y axis mirrored

Driver selection

To use GUIDRV_SPage for the given display, the following call may be used in the function
LCD_X_Config:

```
pDevice = GUI_DEVICE_CreateAndLink(GUIDRV_SPAGE_4C0, GUICC_4, 0,
0);
```

Display data RAM organization

The picture above shows the relation between the display memory and the SEG and COM lines of
the display.

RAM requirements

This display driver can be used with or without a display data cache. The data cache contains a
complete copy of the LCD data RAM. If no cache is used, there are no additional RAM requirements.

It is highly recommended to use this driver with a data cache for faster LCD-access. Not using a
cache degrades the performance of this driver seriously. The amount of memory used by the cache
may be calculated as follows:

```
Size of RAM (in bytes) = (LCD_YSIZE + (8 / LCD_BITSPERPIXEL – 1))
/ 8 * LCD_BITSPERPIXEL * LCD_XSIZE
```

Run-time configuration

The table below shows the available run-time configuration routines for this driver:

Routine	Description
GUIDRV_SPage_Config	Passes a pointer to a CONFIG_SPAGE structure.

GUIDRV_SPage_SetBus8	Tells the driver to use the 8 bit indirect interface and passes pointer to a GUI_PORT_API structure to the driver.
GUIDRV_SPage_Set1502	Set up the driver to use one of the following controllers: • Hitachi HD61202 • S6B0108 (KS0108)
GUIDRV_SPage_Set1510	Set up the driver to use one of the following controllers: • Epson S1D15605, S1D15606, S1D15607, S1D15608, S1D15705, S1D15710, S1D15714 • Integrated Solutions Technology IST3020 • New Japan Radio Company NJU6676 • Novatek NT7502, NT7534, NT7538, NT75451 • OriseTech SPLC502B • Samsung S6B0713, S6B0719, S6B0724, S6B1713 • Sino Wealth SH1101A • Sitronix ST7522, ST7565, ST7567 • Solomon SSD1303, SSD1305, SSD1805, SSD1815, SSD1821 • Sunplus SPLC501C • UltraChip UC1601, UC1606, UC1608, UC1701
GUIDRV_SPage_Set1512	Set up the driver to use one of the following controllers: • Epson S1D15E05, S1D15E06, S1D15719, S1D15721
GUIDRV_SPage_SetST7591	Set up the driver to use the following controller: • Sitronix ST7591
GUIDRV_SPage_SetUC1611	Set up the driver to use the following controller: • UltraChip UC1611

Configuration Example

```
void LCD_X_Config(void) {
  CONFIG_SPAGE Config = {0};
  GUI_DEVICE * pDevice;
  GUI_PORT_API PortAPI = {0};

  //
  // Set display driver and color conversion for 1st layer
  //
  pDevice = GUI_DEVICE_CreateAndLink(DISPLAY_DRIVER,
COLOR_CONVERSION, 0, 0);
  //
  // Display size configuration
  //
  if (LCD_GetSwapXY()) {
    LCD_SetSizeEx (0, YSIZE_PHYS, XSIZE_PHYS);
```

```
      LCD_SetVSizeEx(0, VYSIZE_PHYS, VXSIZE_PHYS);
    } else {
      LCD_SetSizeEx (0, XSIZE_PHYS, YSIZE_PHYS);
      LCD_SetVSizeEx(0, VXSIZE_PHYS, VYSIZE_PHYS);
    }
    //
    // Driver configuration
    //
    Config.FirstSEG = 0;//256 - 224;
    GUIDRV_SPage_Config(pDevice, &Config);
    //
    // Configure hardware routines
    //
    PortAPI.pfWrite8_A0  = _Write8_A0;
    PortAPI.pfWrite8_A1  = _Write8_A1;
    PortAPI.pfWriteM8_A1 = _WriteM8_A1;
    PortAPI.pfReadM8_A1  = LCD_X_8080_8_ReadM01;
    GUIDRV_SPage_SetBus8(pDevice, &PortAPI);
    //
    // Controller configuration
    //
    GUIDRV_SPage_SetUC1611(pDevice);
}
```