Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

SEGGER emWin Graphic Library (emWinGraphics) 5.46

Features

- The package integrates emWin 8051 Graphic Library for PSoC3 and full-featured emWin Graphic Library v5.46 for PSoC 4 and PSoC 5LP.
- The library can be used with the Keil_PK51, GCC and Keil MDK toolchains.
- CompactColor_16, FlexColor, and BitPlains display drivers are available for PSoC 4 and PSoC 5LP.

General Description

The SEGGER emWin is an embedded graphic library and graphical user interface (GUI) designed to provide an efficient, processor- and LCD controller-independent GUI for any application that operates with a graphical display. It is compatible with single-task and multitask environments. Developed by SEGGER Microcontroller, emWin is extremely popular in the embedded industry. Cypress has licensed the emWin library from SEGGER and offers a full-featured graphic library free to customers.

When to Use a emWinGraphics Library

This package provides access to and the functionality of the SEGGER emWin graphic library. The emWinGraphics is a complete, easy to use, software package for the PSoC 3, PSoC 4, and PSoC 5LP devices. The package consists of the emWin GUI provided in library form, the required header files, and the source files specific to the PSoC implementation.

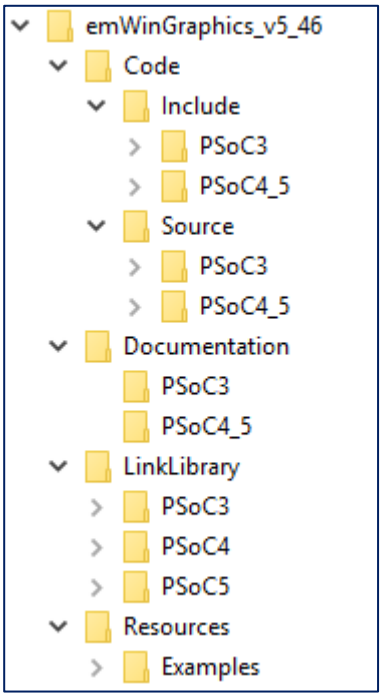
The emWin Graphics Library component allows you fast and efficient GUI development for any application that operates with a graphical LCD.

Library Contents

The emWinGraphics library is supplied as a zip file. You can get the latest library version from the following location:

http://www.cypress.com/go/comp_emWin

The following shows the directory structure for the unzipped library.



The following table describes the contents of all major emWinGraphics Library directories.

Directory	Contents
Code\Include	The GUI header files.
Code\Source	The source files that you need to add to your project
Documentation	The emWin User Guides
LinkLibrary	The emWin libraries for each of the supported toolchains and devices
Resources	emWin sample files and windows programs from SEGGER to work with bitmaps

The *Code\Include*, *Code\Source*, and *Documentation* directories are divided into PSoC3 and PSoC4_5 subdirectories, because the implementations are different:

- PSoC 3 supports the limited library version (emWin 8051).
- PSoC 4 and PSoC 5LP supports the full-featured version of the standard emWin library.

The detailed functionality provided by each library version is documented in the SEGGER emWin documentation, located in the *Documentation\PSoC3* and *Documentation\PSoC4_5* directories. For details about the different areas of functionality for PSoC 3, PSoC 4 and PSoC 5LP, see the Functional Description section of this document.

Using the emWinGraphics Library

To use the emWinGraphics library, follow these steps. All steps are explained further in subsequent sections.

1. Integrate the emWinGraphics Library into your PSoC Creator project.
2. Add a Component to communicate with a graphics LCD panel to your project.

   The emWinGraphics is a software only library. The communication to the panel is done using the appropriate hardware component for the panel. You can use the Graphics LCD Interface (parallel 8/16 bit indirect interface, I8080), Graphics LCD Controller (direct 16-bit RGB interface), SPI, or I2C Components. For more details on supported interfaces for the selected display driver, refer to the “Detailed display driver description” section of the *emWin User Manual* of selected device family.
3. Configure emWin; specify your display settings and display access functions.
4. Compile, link, and test using the provided sample code.

   The emWinGraphics library comes with sample code for both single- and multi-task environments. These examples can be used to learn the usage methodology of the library. These examples are found in the Resources directory.
5. For multi-tasking applications, adapt the library to your operation system (OS).

   If multiple tasks will have access to the display simultaneously, then OS interface routines must be defined. For details and sample applications, refer to the "Execution Model: Single Task/Multitask" section of the *emWin User Manual*.
6. Write your own application using emWin.

   At this point you should be ready to start programming using emWin. Refer to the *emWin User Manual* for the details of each capability the library provides.

Integrating the Library into PSoC Creator

To integrate the emWinGraphics library into your PSoC Creator project, follow these steps. This example uses the GCC toolchain and PSoC 4200 BLE device. Other toolchains and devices are supported in a similar fashion.

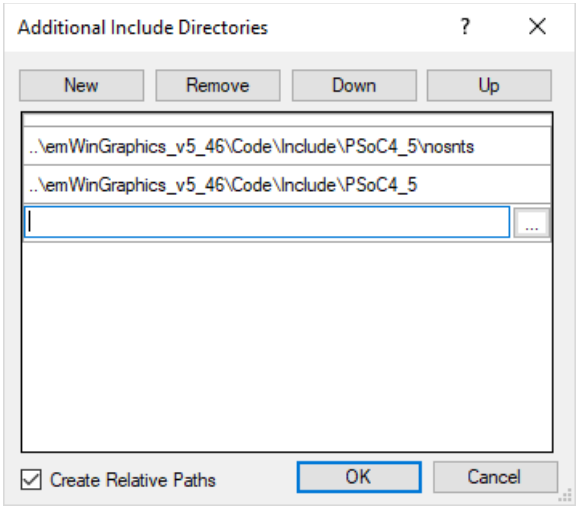
1. Decide which library you need.

   The decision is based on whether you are using an RTOS (os), a touchscreen (ts), or both. The emWin libraries are named according to the supported options. In this example, the library name is *libemWin_nosnts_cm0_gcc.a*, which is a library without OS (nos) and touchscreen (nts) support.
2. Add the necessary include files.

   The emWinGraphics *include* directory contains include files that are general to the entire application and specific include files in subdirectories based on the options chosen.

3. Open the PSoC Creator Build settings dialog and navigate to the **Additional Include Directories** field**: Project > Build Settings > Compiler > General > Additional Include Directories**.

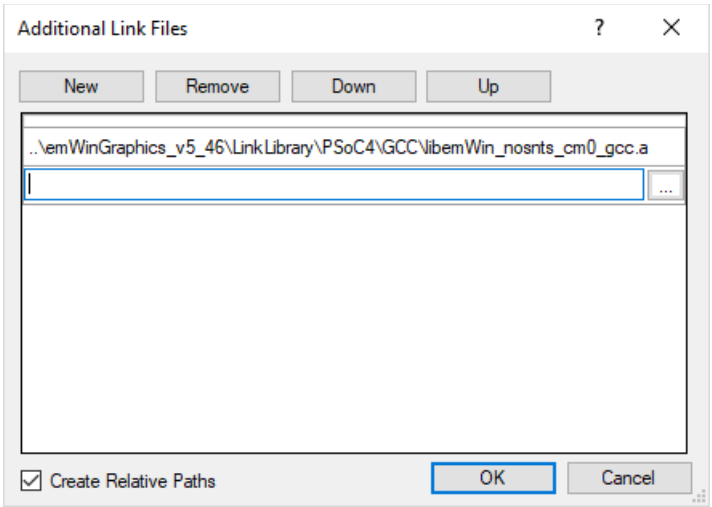
   The Compiler options should contain an entry that addresses the path to the general and library-specific include files. Specify the include path, as shown.



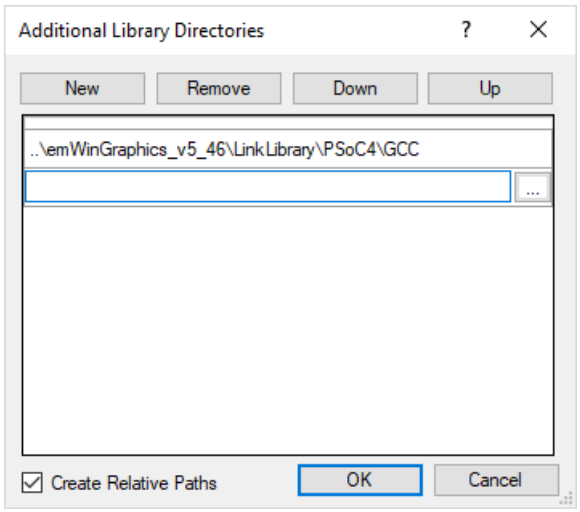
4. Add the link library file.

   The emWin Library can be used with the Keil_PK51, GNU CC, or KEIL MDK toolchain, so select the library according to the toolchain you choose.

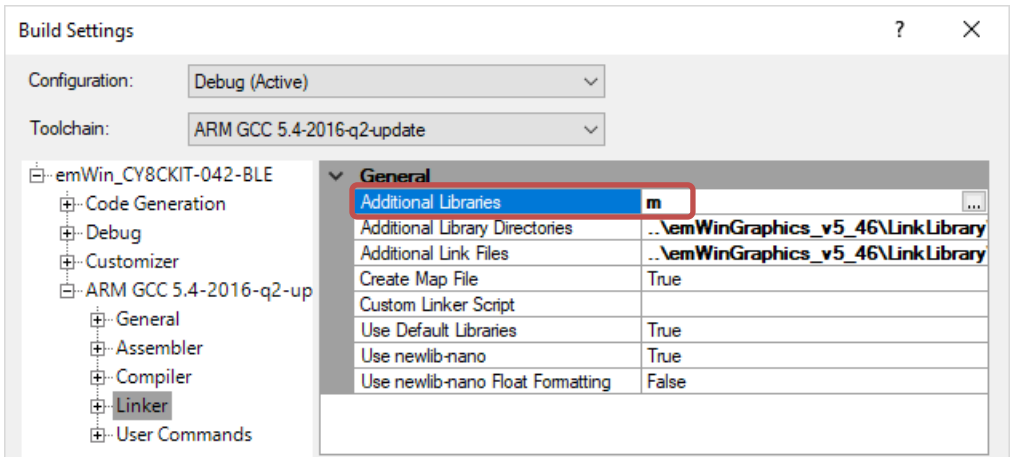
   On the Build Settings dialog, click **Additional Link Files**, and select the library file shown.



Note The process of adding the library to the linker is different for the Keil PK51 toolchain. In this case, you not need to specify additional library directory.

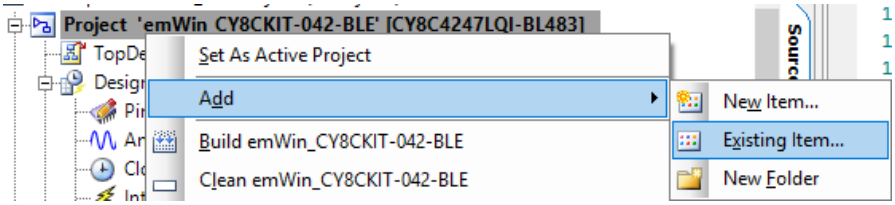


Note Some emWin features, such as arcs, require the *math.h* library. To include this library, add `m` to the **Additional Libraries** field.



5. Add the required source files to the project for the emWin display driver chosen. Select one of CompactColor_16, FlexColor, BitPlains, or Control drivers. Based on that selection, the files are available in the *Source\PSoC4_5\[DriverName]* directory.

   These files are typically edited for the specific project, so Cypress recommends that you copy the files into the root folder of your project and then add them to your project. Add the files by right-clicking on the project and selecting **Add > Existing Item**.

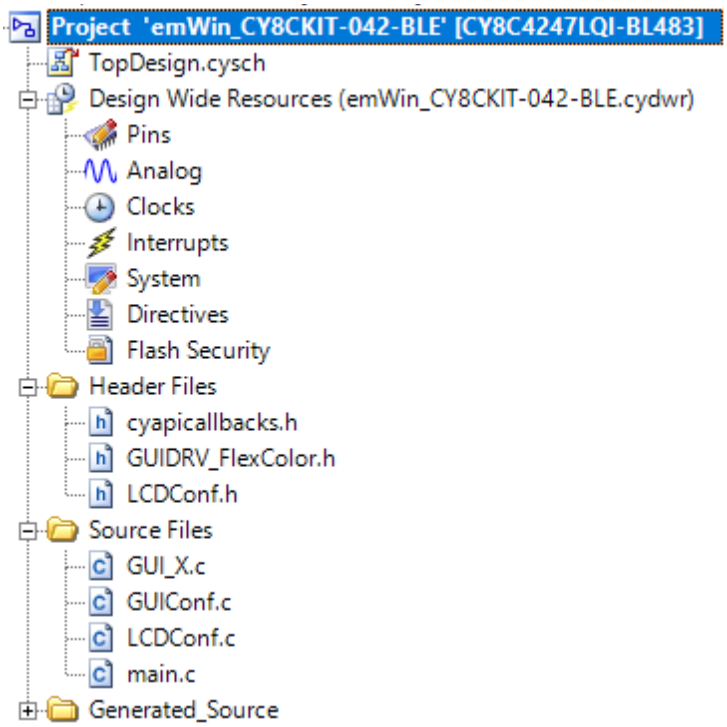


   If the include files are not placed in the root of the project directory, then the path to the include files must also be added to the **Additional Include Directories**.

   Depending on RTOS usage, add to project one of GUI_X files:

   - GUI_X.c – project without RTOS;
   - GUI_X_FreeRTOS.c, GUI_X_embOS.c, GUI_X_uCOS.c – project with FreeRTOS, embOS or uCOS.

The project as shown includes all needed files from the *Source\PSoC4_5\FlexColor* directory for a project without an RTOS.



emWin Basic Configuration

The emWin source files, located at Source directory contains two type of files display driver files and configuration files. The following files are configuration files:

- GUI_X.c – contains the configuration of the emWin timing routines, by default starts the SysTick timer for emWin timing. Use this file when RTOS is not used.
- GUI_X_FreeRTOS.c, GUI_X_embOS.c, GUI_X_uCOS.c - contains the configuration of the emWin timing and multitask routines specific to selected RTOS.
- GUIConf.c – contains configuration of the memory assignment of emWin. This assignment does not change the memory allocation dynamically.
- LCDConf.h – contains the display driver constants.
- LCDConf.c – contains configuration of the display size, orientation and display driver initialization code.
- LCDConf_CompactColor_16.h – contains macros for display hardvare acces. Available only for CompactColor_16 diriver.

To configure emWin:

1. Specify the display resolution, color conversion, and other required display settings in the *LCDConf.c* file.

   Note For Control driver, additional settings are located in the *LCDConf.h* file.

2. Modify the emWin config files with your display interface access functions:
   - **CompactColor_16 driver:** LCD_WRITExxx and LCD_READxxx macros of the *LCDConf_CompactColor_16.h* file. The following is an example of the initialization for a GraphicLCDIntf Component, 8-bit mode:

```
#define LCD_WRITEM_A1(p, Num) GraphicLCDIntf_WriteM8_A1(p, Num)
#define LCD_WRITEM_A0(p, Num) GraphicLCDIntf_WriteM8_A0(p, Num)
#define LCD_READM_A1(p, Num) GraphicLCDIntf_ReadM8_A1(p, Num)
#define LCD_WRITE_A0(Data) GraphicLCDIntf_Write8_A0(Data)
#define LCD_WRITE_A1(Data) GraphicLCDIntf_Write8_A1(Data)
```

- **FlexColor driver:** The LCD interface access functions are in the *LCDConfig.c* file. Modify the GUI_PORT_API PortApi structure passed as the pointer to GUIDRV_FlexColor_SetFunc(). The following is an example of the initialization for a GraphicLCDIntf Component, 8-bit mode:

```
GUI_PORT_API PortAPI = {0};
. . .
PortAPI.pfWrite8_A0 = GraphicLCDIntf_Write8_A0;
PortAPI.pfWrite8_A1 = GraphicLCDIntf_Write8_A1;
PortAPI.pfWriteM8_A1 = GraphicLCDIntf_WriteM8_A1;
PortAPI.pfRead8_A1 = GraphicLCDIntf_Read8_A1;
PortAPI.pfReadM8_A1 = GraphicLCDIntf_ReadM8_A1;
GUIDRV_FlexColor_SetFunc(pDevice, &PortAPI, LCD_CONTROLLER,
GUIDRV_FLEXCOLOR_M16C0B8);
```

- **BitPlains driver:** This driver only manages display buffer and don't have any display specific code. Implement your own display access and update routines.
- **Control driver:** No modification is needed.

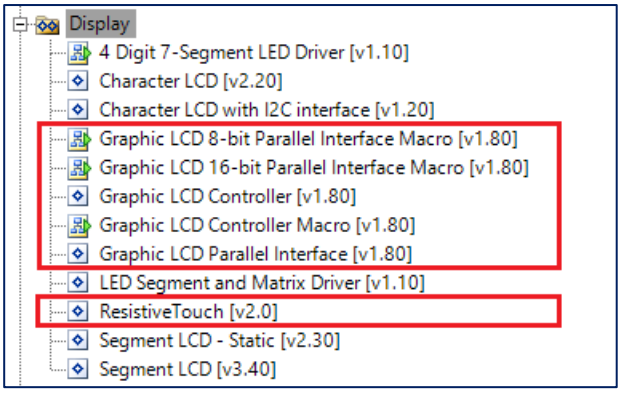
3. Insert the initialization code for the display interface and display the display driver IC in the _InitController() function of the LCDConf.c configuration file.

For details about emWin configuration, refer to the "Configuration" section of the *emWin User Manual*.

Using PSoC Creator Components with emWin

The emWin is a software library. It does not include any hardware to drive a display. A display interface Component (Graphics LCD Interface, Graphics LCD Controller, SPI, or I^2C) is required to build a design using the emWin library. The Graphics LCD Interface and Graphics LCD Controller Components provide parallel 8/16-bit I8080 and direct 16-bit RGB interfaces.

These components are located in the **Display** folder of the Component Catalog.



For PSoC 3 and PSoC 5LP, there is also a ResistiveTouch Component that can be used with emWin to implement a touch user interface.

Graphics LCD Interface (GraphicLCDIntf) Component

The GraphicLCDIntf Component is used to interface to a panel that has a graphic LCD controller and a driver device integrated into the LCD panel. This type of panel also includes a frame buffer that is managed by the LCD controller integrated into the panel. The Component performs read and write transactions to this controller.

To create a GraphicLCDIntf application:

1. Place the Graphic LCD Parallel Interface Macro onto the schematic. Select an 8-bit or 16-bit interface, based on the display controller used.
2. Configure the GraphicLCDIntf component to satisfy the requirements for write and read transactions to the LCD controller. For details about the configuration refer to the GraphicLCDIntf datasheet.
3. Make sure that either CompactColor_16 or FlexColor display driver files are included in your project. The files are located in the *Code\Source\PSoC4_5* directory.

Note For PSoC 3, the GraphicLCDIntf is only supported by the LCD667XX driver, which is located in the *Code\Source\PSoC3\Graphics LCD Interface Parallel* directory.

Graphics LCD Controller (GraphicLCDCtrl) Component

The GraphicLCDCtrl Component provides the interface to an LCD panel that has an LCD driver, but not an LCD controller. This type of panel does not include a frame buffer. The frame buffer must be provided externally. This Component directly drives the control signals and manages the frame buffer in an external SRAM.

To create a GraphicLCDCtrl application:

1. Place the Graphic LCD Controller Macro onto the schematic.
2. Configure the GraphicLCDCtrl component to satisfy the screen refresh requirements and data transaction timing parameters of your LCD panel. For details about the configuration, refer to the GraphicLCDCtrl datasheet.
3. Make sure that all needed Control driver files are included in the project. The files are located in the *Code\Source\PSoC4_5\Control* directory.

Note For PSoC 3, the Control driver files are located in the *Code\Source\PSoC4_5\Graphics LCD Controller* directory.

Available Libraries

The emWin GUI can be used with the Keil_PK51, GCC, or Keil MDK toolchain. For every toolchain, except Keil_PK51, the libraries are available for two options: OS (os) support and touchscreen (ts) support. The OS support provides the necessary hooks during initialization of the Graphics library and hooks for locking and unlocking resources within the library by the OS. The following table shows the options supported by the specific library names:

Option	OS support	TS support
nosnts	NO	NO
nosts	NO	YES
osnts	YES	NO
osts	YES	YES

Application Programming Interface

The emWin Graphics does not include any API functions not present in the standard emWin library. Refer to the *emWin User Manual* for a detailed description of available emWin routines.

emWin Sample Files

The emWin Library is shipped with many emWin sample files available for PSoC 3, PSoC 4, and PSoC 5LP. They are placed in the PSoC 3 or PSoC4_5 subdirectories of the *Resources\Samples* directory. If you are not familiar with emWin, Cypress recommends that you compile, link, and test sample programs before creating your own applications.

To get a sample file running:

1. Add one of the demo sample files to your project.
2. Edit the main source file (*main.c*) to include *GUI.h*. Enable interrupts and then call MainTask() as shown. All of the emWin examples use MainTask() as the entry point.

```
#include <project.h>
#include "GUI.h"

int main()
{
    CyGlobalIntEnable;
    MainTask();

    for(;;)
    {

    }
}
```

3. Build and run the project.

API Memory Usage

The memory usage for the emWin library varies significantly, depending primarily on the application and features used. For more details, refer to the “Performance and Resource Usage” section of the *emWin User Manual*.

MISRA Compliance

The emWin Library is not checked for MISRA compliance.

Placement

The implementation is entirely software.

Performance

The actual performance and resource usage depends on many factors (CPU, compiler, memory model, optimization, configuration, interface to LCD controller, and so on). This section contains performance values of image drawing operations and information about memory resource usage for a typical configuration.

Device	CPU Speed (MHz)	Display Interface Component	bpp	filling	Small fonts	Large fonts	Bitmap 1 bpp	Bitmap 8 bpp	Bitmap 16 bpp
PSoC 3	24	Graphics LCD Interface (16-bit)	16	112K	24K	30K	36K	14K	NA
PSoC 3	60	Graphics LCD Interface (16-bit)	16	279K	56K	70K	82K	33K	NA
PSoC 3	24	Graphics LCD Interface (8-bit)	16	63K	20K	24K	28K	13K	NA
PSoC 3	60	Graphics LCD Interface (8-bit)	16	156K	46K	56K	65K	30K	NA
PSoC 3	24	Graphics LCD Controller	16	34K	21K	23K	26K	12K	NA
PSoC 3	60	Graphics LCD Controller	16	75K	48K	54K	61K	29K	NA
PSoC 4	24	Graphics LCD Interface (16-bit)	16	317K	48K	74K	162K	81K	267K
PSoC 4	48	Graphics LCD Interface (16-bit)	16	588K	86K	132K	294K	141K	493K
PSoC 4	24	Graphics LCD Interface (8-bit)	16	159K	33K	53K	65K	47K	114K
PSoC 4	48	Graphics LCD Interface (8-bit)	16	347K	61K	96K	122K	84K	238K
PSoC 4	24	Graphics LCD Controller	16	65K	134K	153K	60K	45K	47K

Device	CPU Speed (MHz)	Display Interface Component	bpp	filling	Small fonts	Large fonts	Bitmap 1 bpp	Bitmap 8 bpp	Bitmap 16 bpp
PSoC 4	48	Graphics LCD Controller	16	91K	203K	222K	84K	68K	92K
PSoC 5LP	24	Graphics LCD Interface (16-bit)	16	558K	170K	231K	292K	152K	443K
PSoC 5LP	60	Graphics LCD Interface (16-bit)	16	1.09M	371K	521K	655K	347K	874K
PSoC 5LP	24	Graphics LCD Interface (8-bit)	16	280K	88K	102K	114K	82K	183K
PSoC 5LP	60	Graphics LCD Interface (8-bit)	16	592K	195K	231K	260K	190K	401K
PSoC 5LP	24	Graphics LCD Controller	16	92K	174K	187K	82K	70K	90K
PSoC 5LP	60	Graphics LCD Controller	16	162K	310K	368K	150K	132K	167K

Functional Description

The detailed functionality is documented in the SEGGER emWin documentation. This section identifies each of the major categories of SEGGER functionality with respect to PSoC 3, PSoC 4, and PSoC 5LP devices.

Library feature	PSoC 3	PSoC 4 and PSoC 5LP
emWin version	4.0	5.46
2-D graphics library	Full support	Full support
Displaying bitmap files	No support for formats other than compiled bitmap format. No support for 16-bit bitmaps. Eight and smaller bit-count bitmaps are fully supported.	Support for all compiled bitmap formats Other formats (JPEG, GIF, PNG) may be supportable, but may be impractical because of RAM requirements
Fonts	Support for proportional fonts only	Support for all font types
Bitmap converter	Compiled windows application shipped as a binary.	
Color support	Full support	Full support
Memory devices	No support	Full support
Multitask (RTOS)	No support	Full support
Window manager	No support	Full support
Window objects (widgets)	No support	Full support
Virtual screens/Virtual pages	Full support	Full support
Multilayer/Multidisplay	No support	No support
Pointer input devices	Full support of touchscreens only	Full support of touchscreens only

Library feature	PSoC 3	PSoC 4 and PSoC 5LP
Sprites and cursors	No support	Full support
Antialiasing	No support	Full support
Foreign language	No support	No support
VNC	No support	No support
Drivers	LCD667XX Cypress custom driver for GraphicLCDCtrl Component	CompactColor_16 FlexColor BitPlains Cypress custom driver for GraphicLCDCtrl Component

Library Changes

For a list of emWin library changes, refer to the *emWin User Manual.*

SEGGER emWinLibrary Datasheet Changes

This section lists the major changes from the previous version.

Version	Description of Changes	Reason for Changes / Impact
5.46	Updated emWin version from 5.00 to 5.46. Added FlexColor and BitPlains drivers for PSoC 4 and PSoC 5LP devices.	The SEGGER emWin library was updated to version 5.46. Updated this datasheet to match. To support additional devices.
1.0	Initial Version	

© Cypress Semiconductor Corporation, 2018. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.