CE218136 – PSoC 6 MCU E-INK Display with CapSense (RTOS)

Objective

This code example shows how to create a user-interface solution using an E-INK display and CapSense®, on the PSoC® 6 MCU, using ModusToolbox™ integrated development environment (IDE).

Requirements

Tool: ModusToolbox IDE 1.1

Programming Language: C

Associated Parts: All PSoC 6 MCU parts

Related Hardware: PSoC 6 BLE Pioneer Kit

Overview

This code example demonstrates how to create a user-interface solution using an E-INK display with a CapSense slider and buttons. E-INK (electronic ink) displays consume no power for image retention. Together with PSoC 6 MCU's CapSense touch sensing, an E-INK display can be used to create user interfaces that have "always-on" functionality.

This code example assumes that you are familiar with the PSoC 6 MCU and the ModusToolbox IDE. If you are new to PSoC 6 MCU, see the application note AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity.

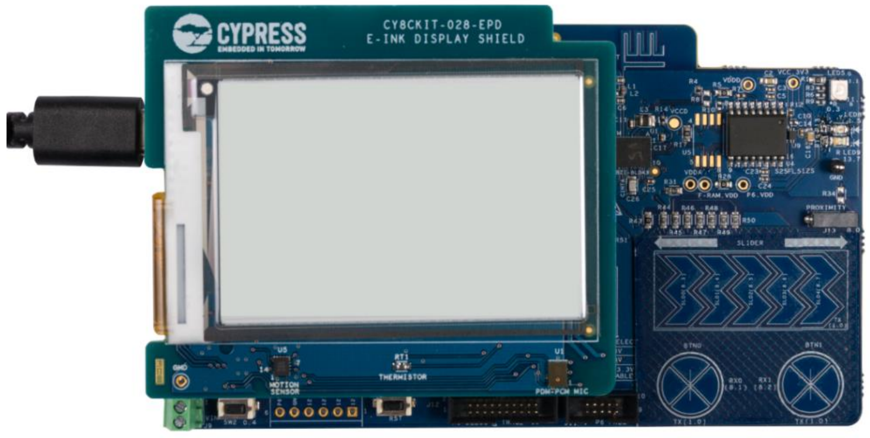
This code example uses FreeRTOS. Visit the FreeRTOS website for documentation and API references of FreeRTOS.

Segger emWin library is used as a middleware that handles the graphical user interface (GUI). Visit the emWin website for the user manual.

Hardware Setup

Plug in the E-INK display shield on to the Pioneer Board as Figure 1 shows.

Figure 1. Hardware Setup



Set the switches and jumpers on the Pioneer Board as shown in Table 1.

Table 1. Switch and Jumper Selection

Switch/Jumper	Position	Location
SW5	3.3 V	Front
SW6	PSoC 6 BLE	Back
SW7	V_{DDD} / KitProg2	Back
J8	Installed	Back

See the CY8CKIT-062-BLE Pioneer Kit documentation for more details of these jumper and switch settings.

Note: The PSoC 6 BLE Pioneer kit ships with KitProg2. ModusToolbox only works with KitProg3. Before using this code example, make sure that the kit is upgraded to KitProg3. See ModusToolbox Help > ModusToolbox IDE Documentation > User Guide; section PSoC 6 MCU KitProg Firmware Loader. If you do not upgrade, you will see an error like “unable to find CMSIS-DAP device” or “KitProg firmware is out of date”.

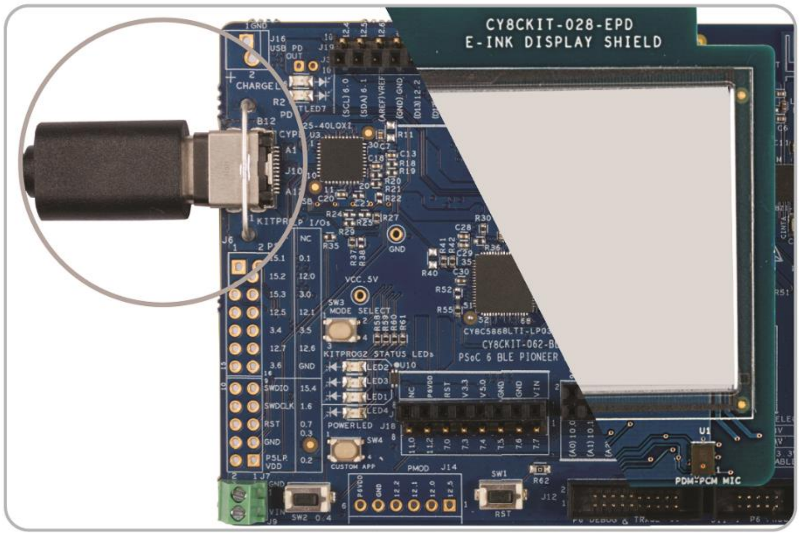
Software Setup

This example uses a terminal emulator for viewing debug messages (optional). Install one if you don’t have one. The instructions use Tera Term.

Operation

1. Connect the Pioneer Board to your PC using the provided USB cable through the USB connector (J10).

Figure 2. Connecting the USB Cable to the Pioneer Board



2. Program the Pioneer Board with the CE218136 project. The E-INK display refreshes and shows the startup screen for a second, followed by a menu that lists important information about the kit and associated software, as Figure 3 shows. LED9 (red) turns ON if the E-INK display is not detected. In this case, check the connection between the E-INK Display Shield and the Pioneer Board, and then reset the Pioneer Board.
3. Use the CapSense slider and buttons to navigate the menu, as Figure 4 shows.

   Note that the display takes about a second to refresh the display following a touch input. LED8 (orange) turns ON when the display is busy. The main menu uses partial update for faster refreshes (for details, see the `cy_eink_update_t` parameter of the `Cy_EINK_ShowFrame` function in Appendix - E-INK Display Peripheral Driver Functions, so the selection arrow may have slight ghosting.

Figure 3. Main Menu

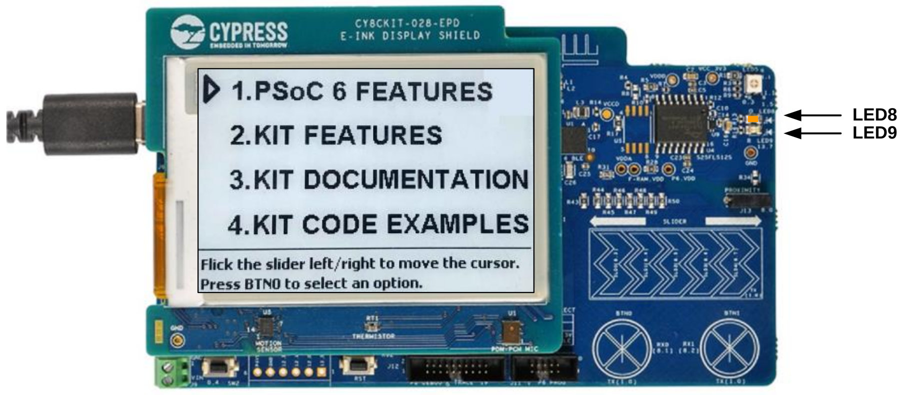
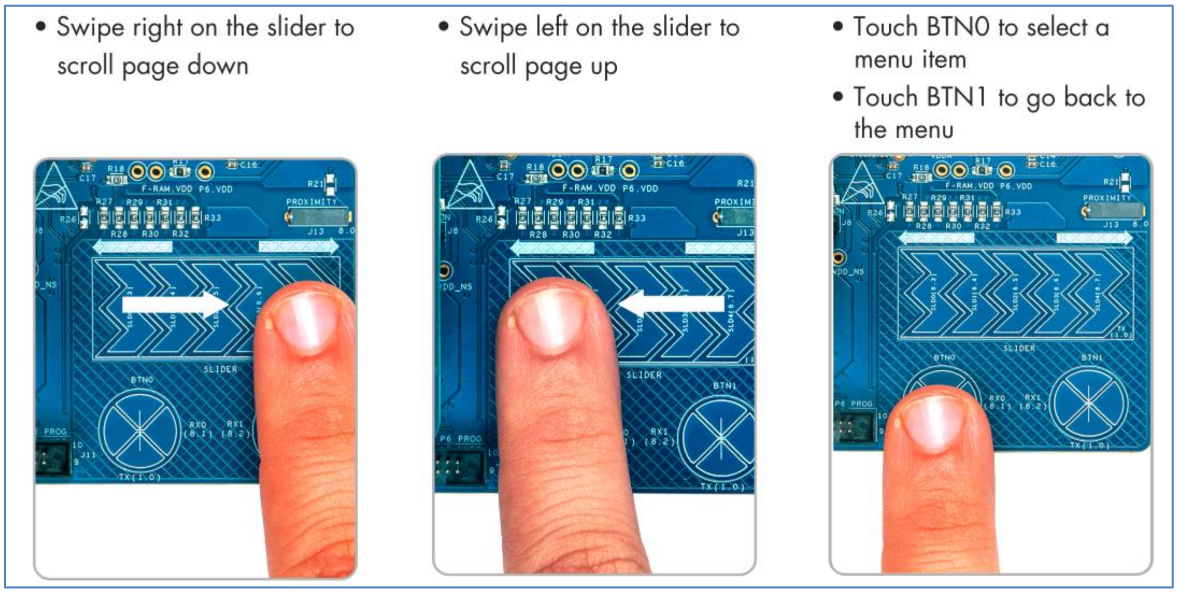


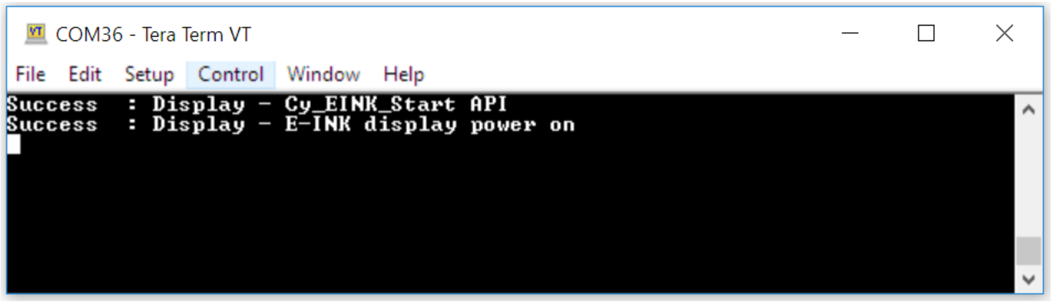
Figure 4. Menu Navigation Options


• Swipe right on the slider to scroll page down
• Swipe left on the slider to scroll page up
• Touch BTN0 to select a menu item
• Touch BTN1 to go back to the menu

Viewing Debug Messages

This code example allows you to view debug messages from various tasks and functions using a serial port terminal emulator such as Tera Term or HyperTerminal as Figure 5 shows.

Figure 5. Viewing Debug Messages using UART



```
Success   : Display - Cy_EINK_Start API
Success   : Display - E-INK display power on
```

Note: This feature is disabled by default for higher performance. To enable the UART debug feature, set `UART_DEBUG_ENABLE` macro in the *uart_debug.h* to “true”.

After re-building the projects and re-programming PSoC 6 MCU with the updated project, set up a serial port terminal emulator with these settings to view the debug information:

■ Baud rate	:	115200
■ Data size	:	8-bit
■ Parity	:	None
■ Stop	:	1-bit
■ Flow Control	:	None

Debugging

You can debug the example to step through the code. Use a **Program+Debug** configuration. If you are unfamiliar with how to start a debug session with ModusToolbox IDE, see KBA224621 in the Cypress community.

Design and Implementation

E-INK is a paper-like display technology, characterized by high contrast, wide viewing angles, and minimal standby power. Unlike conventional backlit, flat panel displays that emit light, E-INK displays reflect light like paper. This makes E-INK displays more comfortable to read, and provides a wider viewing angle than most light-emitting displays. Therefore, E-INK displays are comfortable to read even in sunlight.

This project uses a CY8CKIT-028-EPD E-INK Display Shield together with a Pioneer Board. The E-INK Shield has a 2.7-inch E-INK display with a resolution of 264×176 pixels.

For details on the Pioneer Board and E-INK Display Shield, see the Pioneer Kit Guide.

The E-INK display in CY8CKIT-028-EPD contains a basic driver IC that interfaces with the PSoC 6 MCU using a custom SPI interface. The driver converts a serial data stream into individual pixel data and generates the voltages required for the E-INK display. PSoC 6 MCU has low-level control of the E-INK display.

This code example contains the required peripheral functions for driving the E-INK display. However, the actual hardware driver functions are not covered in this document. See the E-INK display driver document for more details.

The PSoC 6 MCU controls the E-INK display’s reset, enable, discharge, and border pins. PSoC 6 MCU also reads the status of the display to determine whether the display is busy with a previous operation. A load switch on CY8CKIT-028-EPD, controlled by the PSoC 6 MCU device, can be used to turn the display ON/OFF. A voltage level translator is connected between the E-INK display and PSoC 6 MCU GPIOs so that PSoC 6 MCU can operate with variable V_{DD}. The enable input of the voltage level translator is also connected to a PSoC 6 MCU GPIO so that PSoC 6 MCU can disable the level translator to reduce power consumption when the E-INK display is not used.

In this project, PSoC 6 MCU scans a self-capacitance based, 5-element CapSense slider and two mutual capacitance CapSense buttons for user input. CapSense uses SmartSense auto-tuning and gesture recognition. See AN85951 – PSoC 4 and PSoC 6 MCU CapSense Design Guide for more details of CapSense features and usage.

Based on the user input, the E-INK display is updated to scroll through menu items to change information pages, and to move back and forth between the top-level menu and information pages as Figure 6 shows.

The project consists of the following files:

- *main.c* (CM4) contains the main function, which is the entry point and execution of the firmware application. The main function sets up user tasks and then starts the RTOS scheduler.
- *touch_task.c/h* files contain the task that initializes CapSense touch sensing and read touch and gesture data from buttons and sliders.
- *display_task.c/h* files contain the task that changes the menu displayed on the E-INK display per the gesture input.
- *menu_configuration.h* contains macros and datatypes that define the menu structure.
- *screen_contents.c/.h* and *Cypress_Logo_1bpp.c* files constitute storage for the images and text displayed on the screen.
- *uart_debug.c/h and stdio_user.c/h* files are used for retargeting I/O functions such as printf via UART to a terminal emulation program.

RTOS files:

- *FreeRTOSConfig.h* contains the FreeRTOS settings and configuration. Non-default settings are marked with in-line comments. For details of FreeRTOS configuration options, see the webpage FreeRTOS customization.

E-INK Peripheral and Driver files:

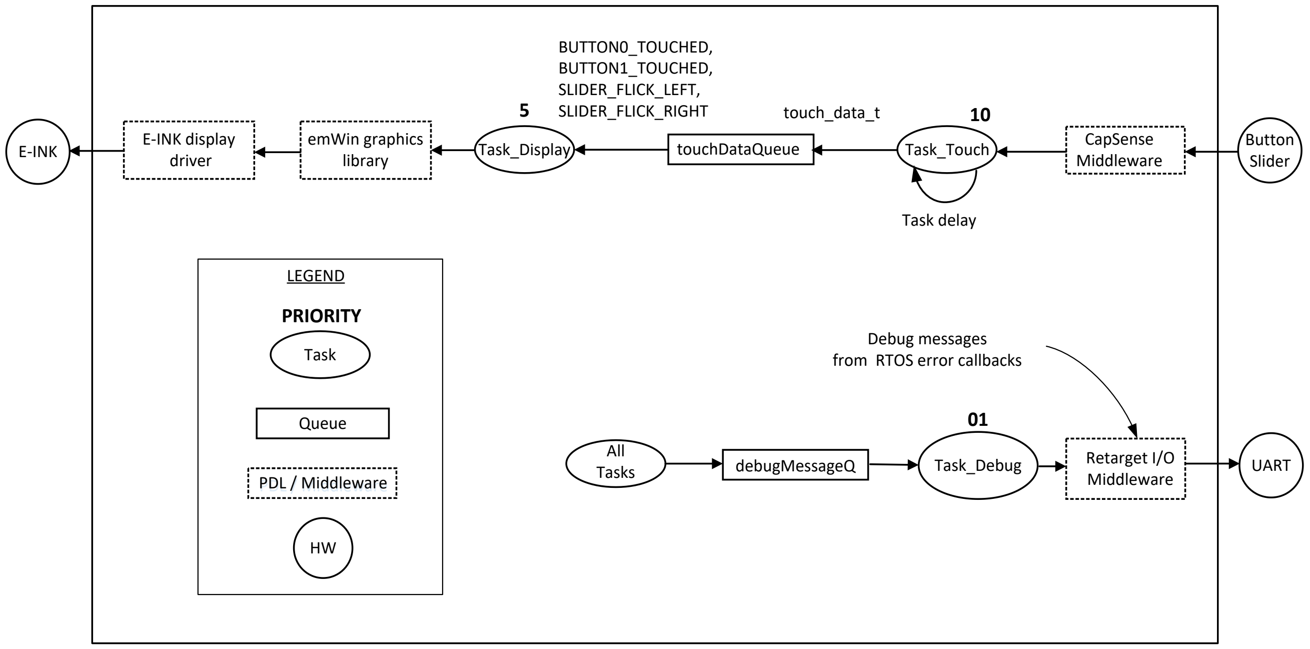
- *cy_cy8ckit_028_epd.c/.h* files contain the E-INK peripheral functions and macros for the display in CY8CKIT-028-EPD. See in Appendix - E-INK Display Peripheral Driver Functions for the E-INK API reference.
- *pervasive_eink_configuration.h* file contains display-vendor-provided definitions of register indexes and hardware parameters of the E-INK display.
- *pervasive_eink_hardware_driver.c/.h* files contain display-vendor-provided low-level display hardware driver functions.
- *cy_eink_psoc_interface.c/.h* files contain the PSoC 6 MCU PDL-level interface to the display hardware.

In addition, the project also contains emWin middleware user files *GUI_X.c, GUIConf.c/h, LCDConf.c/h* that are automatically included when the emWin middleware are enabled. For details of these files and API reference, see the user manual available at emWin website. Following changes were made to the *LCDConf.c* file to be compatible with the E-INK display on CY8CKIT-028-EPD.

1. Updated *XSIZE_PHYS* and *YSIZE_PHYS* with the horizontal and vertical resolutions of the E-INK display.
2. Updated *LCD_X_Config* function to include *GUIDRV_BitPlains_Config* function, so that emWin BitPlains library is automatically initialized when *LCD_X_Config* is called.
3. Added function *LCD_CopyDisplayBuffer* that copies BitPlains display buffer into a user memory location that can be used by the E-INK display driver.

Figure 6 shows the RTOS firmware flow of this project.

Figure 6. RTOS Firmware Flow



Resources and Settings

Table 2 lists some of the ModusToolbox resources used in the example, and how they are used in the design. The *design.modus* file contains all the configuration settings. For example, for pin usage and configuration, go to the **Pins** tab of the design file.

Table 2. ModusToolbox Resources

Resource	Alias	Purpose	Non-default Settings
SCB6	CY_EINK_SPIM	Provide communication between E-INK display and PSoC 6	Mode: Master Sub Mode: Motorola SCLK Mode: CPHA=0, CPOL=0 Data Rate: 12000 Oversample: 4
CapSense	CapSense	Reads touch and gesture information	Added a 5-element linear slider and 2 mutual cap buttons. SmartSense and flick gesture recognition has been enabled for the slider.
SCB5	Bridge_UART	Used for printing debug messages	Baud rate: 115200 Oversample: 8
SAR	KIT_ADC	Used to measure ambient temperature from a thermistor circuit, which is used for E-INK display initialization	Vref Select: Vdda Number of channels: 2 Samples averaged: 256 Channel 0 input mode: Differential Channel 1 input mode: Differential

Note: Open the **CapSense configurator** and see AN85951 – PSoC 4 and PSoC 6 MCU CapSense Design Guide to understand the additional configuration and tuning parameters.

Note: E-INK uses several software controlled pins. Open the **Pins** tab and see the E-INK display driver document for the functionality of each pin.

Following middleware are used in this project:

- CapSense, Soft FP prebuilt library
- FreeRTOS v10.0.1
- Retarget I/O
- Segger emWin BitPlains display driver v5.46
- Segger emWin core no OS, no Touch, Soft FP v5.46

To configure middleware, right-click the *CE218136_mainapp* folder in the project explorer window and select **ModusToolbox Middleware Selector**.

Related Documents

Application Notes	
AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC project
Design Guides	
AN85951 – PSoC 4 and PSoC 6 MCU CapSense Design Guide	Describes how to design CapSense applications on PSoC devices
Code Examples	
Visit the Cypress GitHub site for a comprehensive collection of code examples using ModusToolbox IDE	
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kits	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
Tool Documentation	
ModusToolbox IDE	The Cypress IDE for IoT designers

Cypress Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right device, and quickly and effectively integrate the device into your design.

For the PSoC 6 MCU devices, see KBA223067 in the Cypress community for a comprehensive list of PSoC 6 MCU resources.

Appendix - E-INK Display Peripheral Driver Functions

This section describes the functions provided by the E-INK display peripheral driver. These functions are in the *cy_cy8ckit_028_epd.c* file.

```
cy_eink_api_result Cy_EINK_Start (    int8_t                        temperature
                                      cy_eink_delay_function_t      delayFunction
                                 )
```

Initializes the E-INK display hardware and PSoC Components

Parameters

`temperature`	Ambient temperature in degree Celsius
`delayFunction`	Pointer to a function that accepts delay in milliseconds (uint32_t) and returns void

Returns

cy_eink_api_result	CY_EINK_SUCCESS – operation successful
	CY_EINK_FAILURE – operation failed

Note

After initialization of the E-INK hardware, this function turns OFF the power to the display.

```
bool Cy_EINK_Power (    bool        powerCtrl
                   )
```

Turns ON/OFF the power to the E-INK display and initializes the E-INK driver

Parameters

`powerCtrl`	false – power OFF, true – power ON

Returns

cy_eink_api_result	CY_EINK_FAILURE – driver initialization failed
	CY_EINK_SUCCESS – driver initialization was successful

Note

Display contents will be retained even after the display power has been turned OFF.

```
void    Cy_EINK_ShowFrame (    cy_eink_frame_t*       prevFrame
                               cy_eink_frame_t*       newFrame
                               cy_eink_update_t       updateType
                               bool                   powerCycle
                          )
```

Updates the display with a frame (image or pixel data stored in flash/RAM).

Parameters

`prevFrame`	Pointer to the frame that is currently displayed on the E-INK display. A frame consists of 5808 bytes (264x176/8) of data in which each bit stores the pixel information of a monochromatic (1-bit color per pixel) image.
`newFrame`	Pointer to the frame to be displayed

`updateType`	`CY_EINK_PARTIAL` – updates the display from the previous frame to the new frame without any intermediate stages. This is the fastest type of update (~0.4 seconds), however, may produce ghosting if the new frame differs considerably from the previous frame. `CY_EINK_FULL_2STAGE` – updates the display from the previous frame to the new frame with an intermediate stage that updates the display with the inverted version of the previous frame. This additional stage reduces ghosting, but increases the update time (~0.8 seconds). `CY_EINK_FULL_4STAGE` – updates the display from the previous frame to the new frame with three intermediate stages that consists inverted version of the previous frame, white frame, and inverted version of the new frame. This type of refresh produces minimal ghosting at the cost of having the longest update time (~1.6 seconds).
`powerCycle`	false – does not control the power ON/OFF automatically. true – automatically controls the power cycle. This function will turn ON the power, clear the display, and then turn the power OFF.

Returns

None

Note

If the `powerCycle` value is false, then EINK display should be powered ON (using the `Cy_EINK_Power` function) before calling this function. Otherwise, the display will not be updated.

Document History

Document Title: CE218136 – PSoC 6 MCU E-INK Display with CapSense (RTOS)

Document Number: 002-25470

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6375884	NIDH	11/05/2018	New code example
*A	6479227	NIDH	02/11/2019	Updated for ModusToolbox 1.1

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Code Examples | Projects | Videos | Blogs | Training| Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709