

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.

Objective

This code example demonstrates gesture detection in PSoC® 4 with a CapSense® touchpad using the PSoC Creator™ CapSense Component.

Requirements

Tool: PSoC Creator 4.3

Programming Language: C (Arm® GCC 5.4.1)

Associated Parts: All PSoC 4 family devices that can support a CapSense Component with gesture detection

Related Hardware: [CY8CKIT-041-41XX 4100S](#), [CY8CKIT-041-40XX 4000S](#)

Overview

CapSense touchpads are used for controls requiring gradual adjustments. With the gesture detection capability, CapSense touchpads can detect a touch, position movement, and gesture. This allows the implementation of advanced user interfaces based on capacitive sensing.

This code example includes two projects that demonstrate the implementation of gesture detection on a CapSense touchpad using the CapSense Component. The first uses Self-Capacitance (CSD), while the second project uses Mutual-Capacitance (CSX). For more information, see [Related Documents](#).

To demonstrate the gesture functionality, PSoC devices control three LEDs whose brightness is controlled based on the position detected on the touchpad. These are turned ON or OFF depending on the detected gestures. The CapSense tuner can also be used for real-time tuning and gesture detection.

Hardware Setup

No hardware setup required.

Software Setup

No software setup required.

Operation

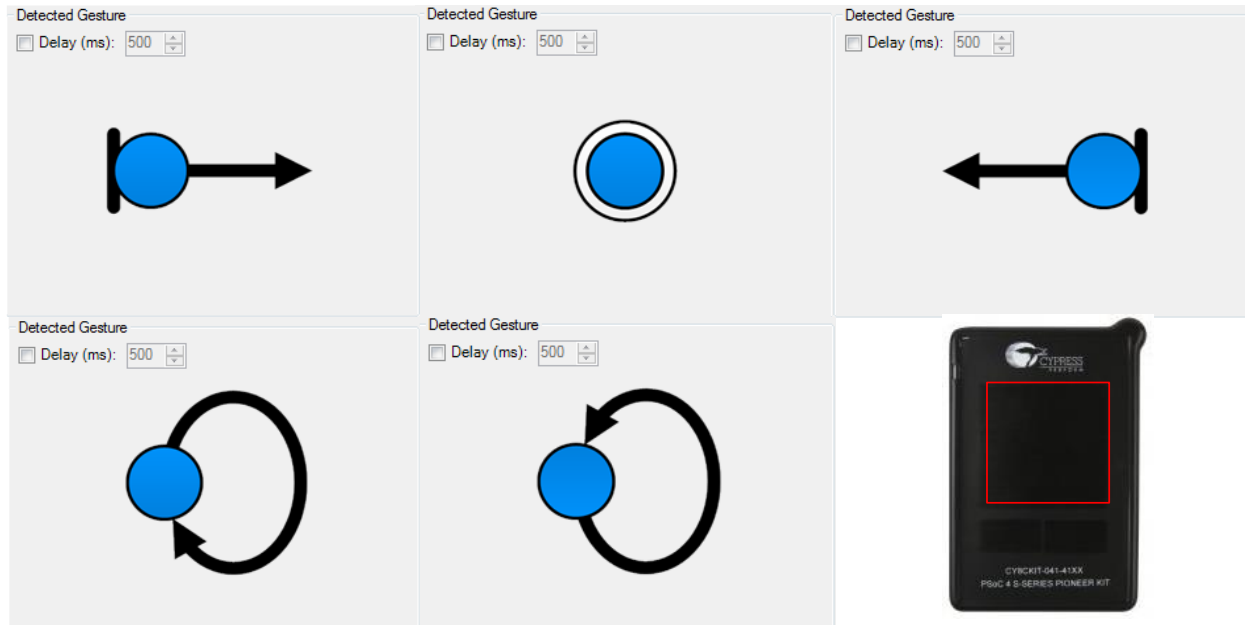
1. Connect CY8KIT-041-41XX to your computer using a USB cable. By default, the projects are configured for CY8KIT-041-41XX. To use the CY8CKIT-041-40XX, see [Reusing this Example](#) first.
2. Build the project and program it into the PSoC 4 device. Choose **Debug > Program**. For more information on device programming, see *PSoC Creator Help*.
3. Tap on the touchpad once and confirm that the blue and green LEDs turn ON.
4. Slide your finger slowly on the touchpad. Confirm that when your finger slides up and down, the green LED changes brightness. Confirm that when you slide your finger left and right, the blue LED changes brightness.
5. Place your finger on the left edge of the touchpad and swipe right. Confirm that the blue LED turns OFF; doing this again turns the LED back ON.
6. Place your finger on the right edge of the touchpad and swipe left. Confirm that the green LED turns OFF; doing this again turns the LED back ON.
7. Rotate your finger clockwise on the touchpad in a circular motion. Confirm that the red LED turns ON.
8. Rotate your finger counterclockwise on the touchpad in a circular motion. Confirm that the red LED turns OFF.

9. Right-click on the CapSense Component and select **Launch Tuner**. Click **Connect**, select **I2C**, and then click **Start**. Ensure that the data rate is set to **1 MHz**. Go to the **Gesture View** tab and select **Synchronized**. As each gesture is performed, confirm that the gesture shows up under the **Detected Gesture** portion of the page. For more information, check the CapSense datasheet under [Related Documents](#).

Design and Implementation

Each gesture is captured and is used to control different LEDs. The tuner also shows different gestures that the slider has detected and shows a picture on screen as shown in [Figure 1](#). All CapSense gesture settings can be manipulated as seen in [Components and Settings](#).

Figure 1. Edge Swipe Right, Click, Edge Swipe Left, Rotate Clockwise, Rotate Counterclockwise, Touchpad Area



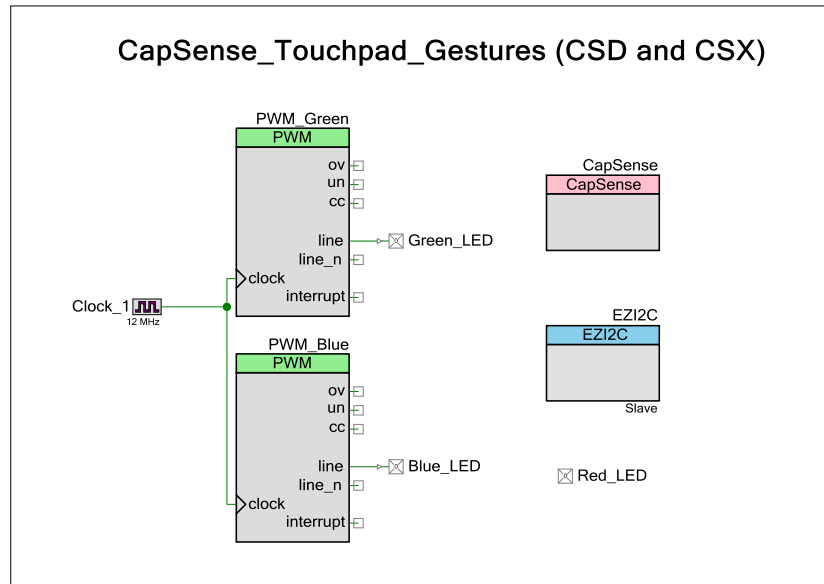
CapSense Touchpad and Gestures

In the CapSense_Touchpad_Gestures_CSD and CSX examples, the following functions are performed:

1. Initialize and start all hardware.
2. Link the communication buffer for EZI2C to the CapSense data structure.
3. Set up a timestamp method to control gestures.
4. Initial scan of all CapSense widgets.
5. Check whether the scan was completed, and then process all data.
6. Store the gesture and use it in a switch statement to control the LEDs.
7. Send all data to the tuner.
8. Scan the CapSense widget and return to Step 5.

Figure 2 shows the top-design of the CSX and CSD PSoC Creator project:

Figure 2. CapSense_Touchpad_Gestures (CSD and CSX) Top Design Schematic



Components and Settings

Table 1 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1. PSoC Creator Components

| Component | Instance Name | Purpose | Non-default Settings |
|----------------------|-----------------------|---|---|
| CapSense | CapSense | Gather and process all data from the selected sensors | CSD |
| | | | For Touchpad settings, see Figure 3 . For CSD Settings, see Figure 4 . For Widget Details, see Figure 5 . |
| | | | CSX |
| | | | For Touchpad settings, see Figure 3 , but change the Sense Mode to CSX. For CSX settings, see Figure 6 . For Widget Details, see Figure 7 . |
| EZI2C | EZI2C | Transmits data from the kit to the tuner | CSD and CSX have the same gestures setup. For Click gesture settings, see Figure 8 . For Edge Swipe gesture settings, see Figure 9 . The Rotate gesture has no setting changes. |
| | | | Change the Data rate to 1000 (kbps) and the Sub-address size to 16 . |
| PWM | PWM_Green PWM_Blue | Controls the duty cycle of the green and blue LEDs | Both PWMs have the same hardware setup. Under the PWM tab, deselect all interrupts, change the Period to 10001 , and the Compare value to 10001 . |
| Design Wide Resource | Clocks | Settings for all clocks | Click on IMO and change the Frequency to 48 MHz . |

Figure 3. CSD CapSense Touchpad Settings

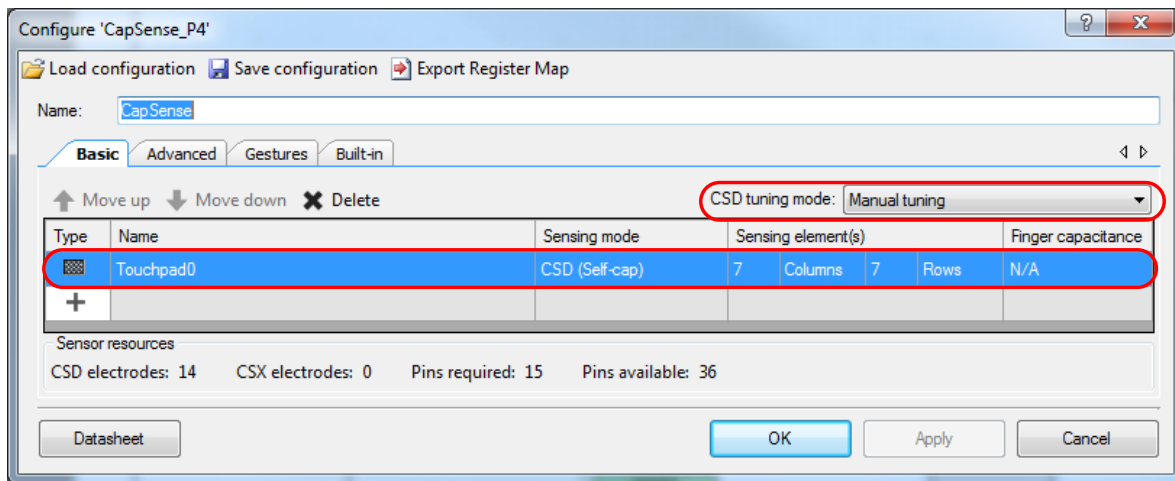
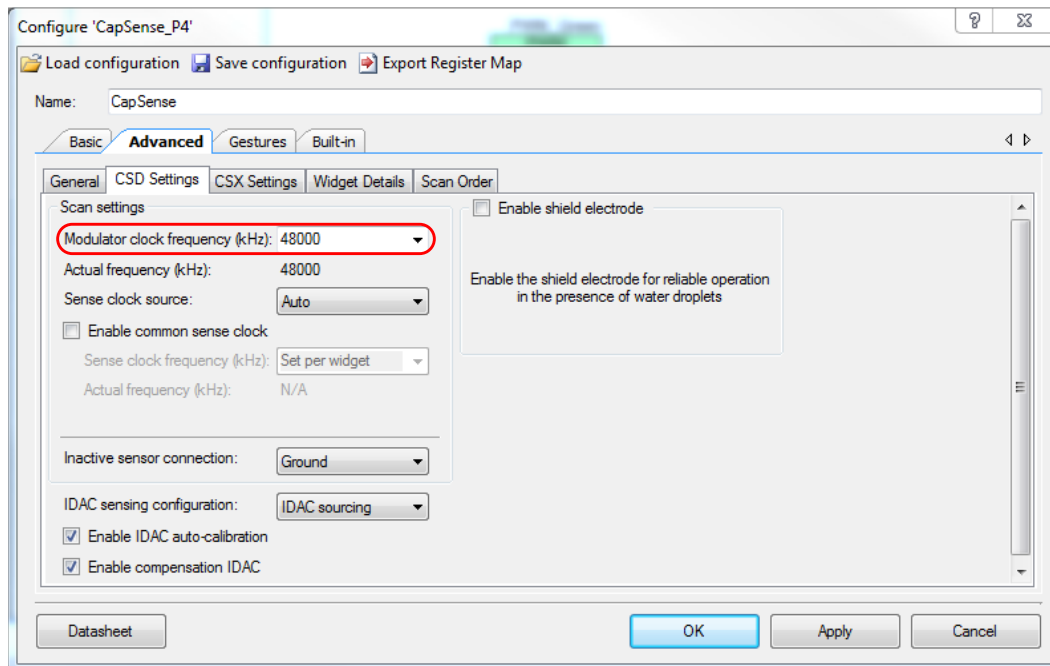


Figure 4. CSD Settings



Note: The Modulator Clock Frequency can be changed to 48,000 kHz only after the IMO clock is changed to 48 MHz. Under Design Wide Resources, click **Clocks** > **System IMO** > Select the drop down on the IMO Clock > **48 MHz**.

Figure 5. Widget Details

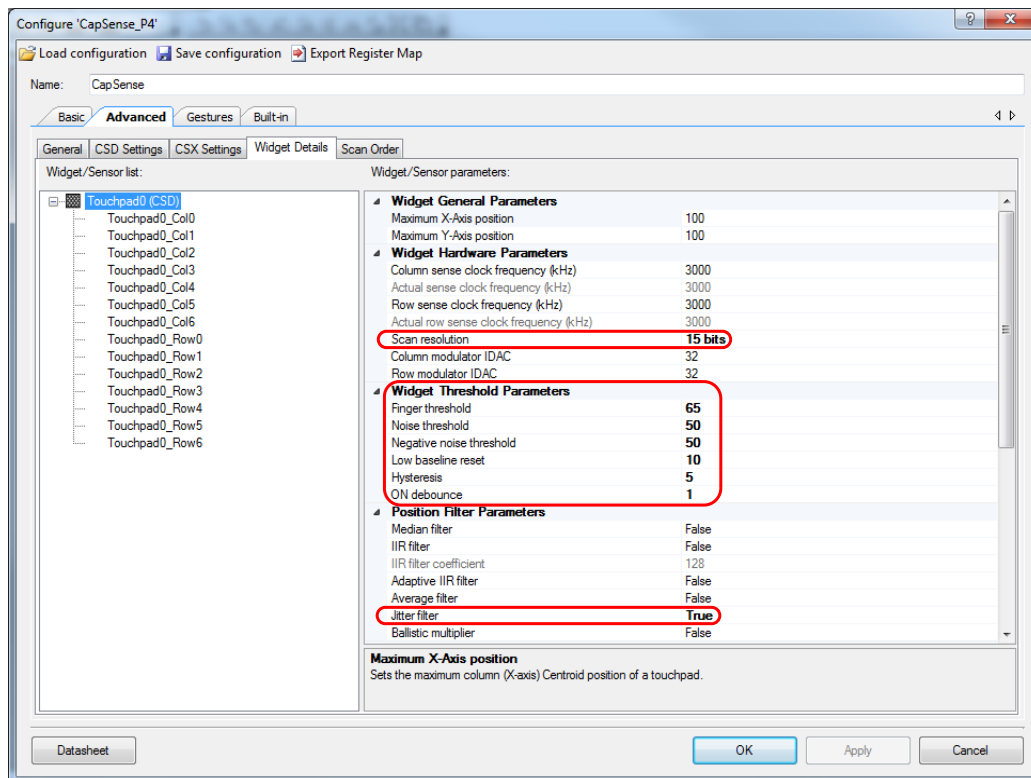
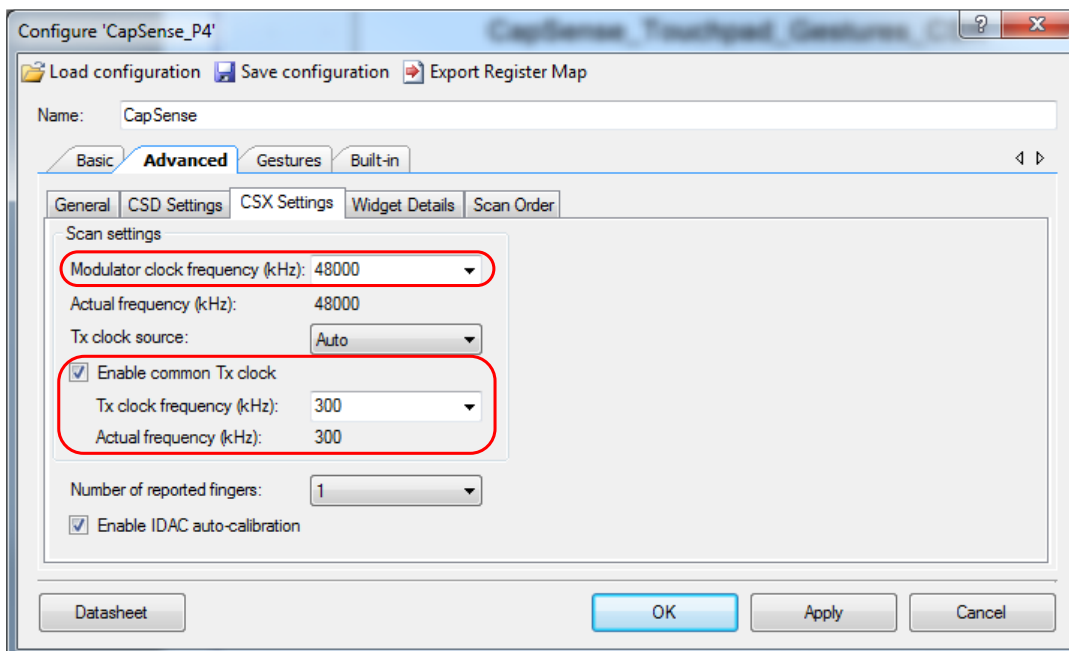


Figure 6. CSX Settings



Note: The Modulator Clock Frequency can be changed to 48,000 kHz only after the IMO clock is changed to 48 MHz. Under Design Wide Resources, click **Clocks** > **System IMO** > Select the drop down on the IMO Clock > **48 MHz**.

Figure 7. Widget Details

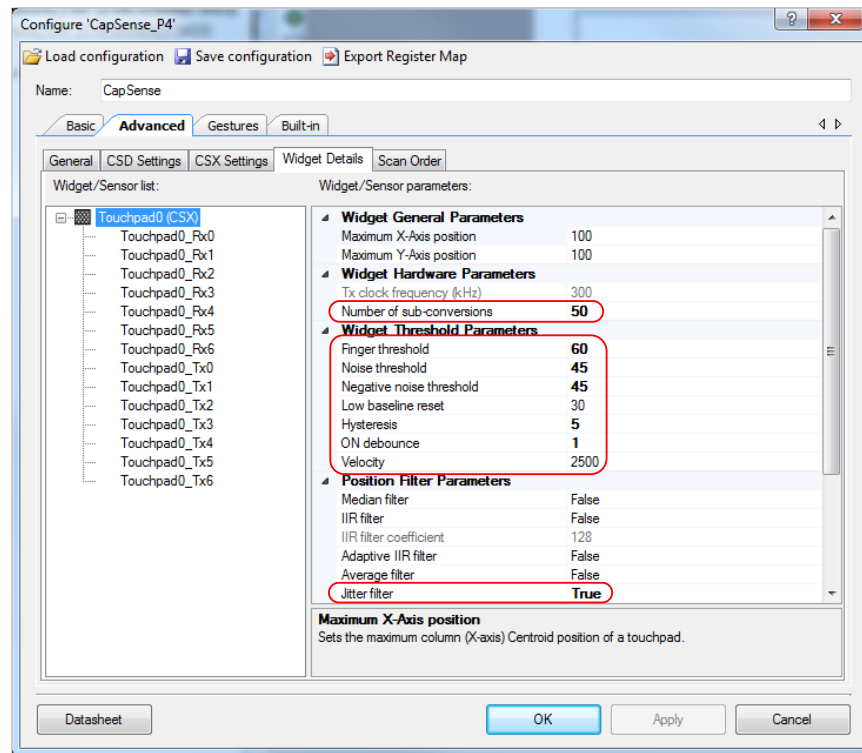


Figure 8. Click Gesture Settings

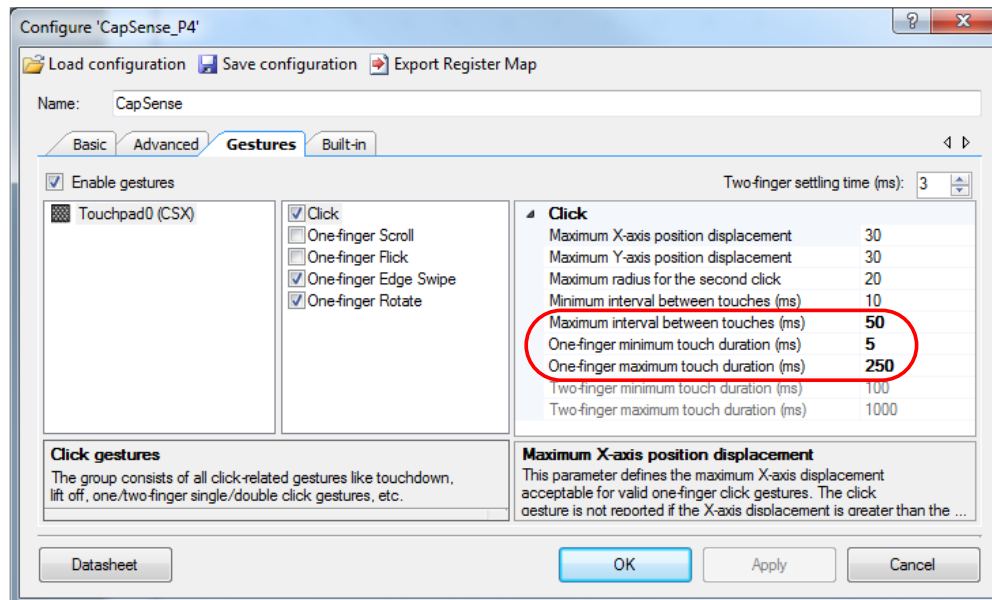
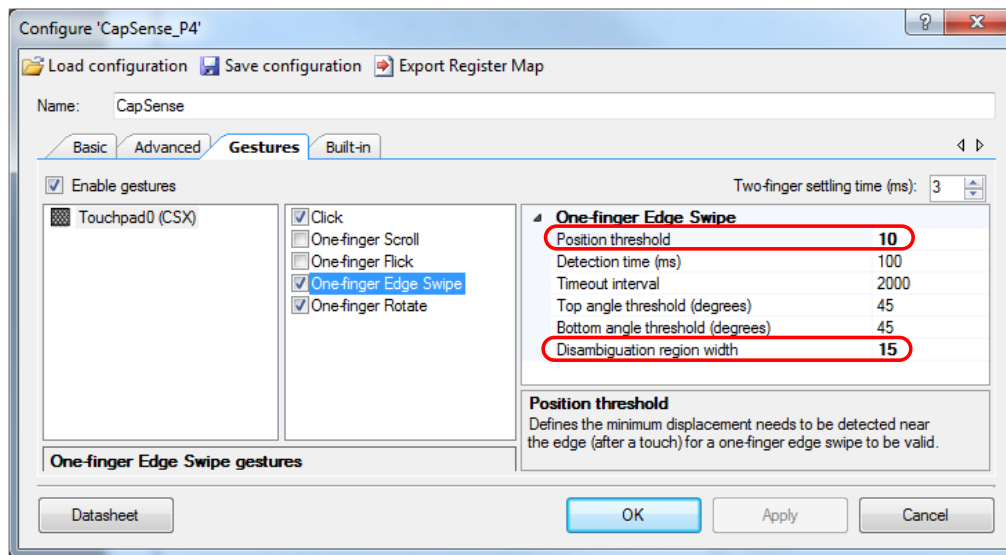


Figure 9. Edge Swipe Gesture Settings



For information on the hardware resources used by a Component, see the Component datasheet.

Reusing this Example

This code example can be used with all PSoC 4 family devices that can support a CapSense Component with gesture detection. By default, the projects are set up to work with CY8KIT-041-41XX (MPN CY8C4146AZI-S433). If CY8CKIT-041-40XX (MPN CY8C4045AZI-S413) is used, follow the directions below. Ensure the following before porting this code example:

1. One of the two kits listed is used.
2. All the pins are unlocked in the Design Wide Resources.

To port the code to a new device, in PSoC Creator, select **Project > Device Selector** and change to the target device.

Before porting this example to another device, note the following:

1. Not all PSoC 4 devices have CapSense Components with gesture detection.
2. Pinouts change from device to device. Some pins may need to be moved. See the **Pin Layout** tab in PSoC Creator

In some cases, a resource used by a code example (for example, a Universal Digital Block) is not supported on another device. In that case, the example does not work. If you build the code targeted at such a device, errors occur. See the device datasheet for information on what a particular device supports.

Related Documents

For a comprehensive list of PSoC 3, PSoC 4, and PSoC 5LP resources, see [KBA86521](#) in the Cypress community.

| Application Notes | |
|---|---|
| AN79953 – Getting Started with PSoC® 4 | Describes PSoC 4 devices and how to build your first PSoC Creator project |
| AN85951 – PSoC 4 and PSoC 6 MCU CapSense Design Guide | Describes how to tune and use the CapSense Component |
| AN64846 – Getting Started with CapSense | Describes how to set up CapSense for beginners |
| Code Examples | |
| CE224820 – CapSense Slider and Gestures | Shows how to use Gestures on a slider |
| PSoC Creator Component Datasheets | |
| CapSense | CapSense Component datasheet for more information |
| TCPWM | TCPWM Component datasheet for more information |
| Device Documentation | |
| PSoC 4 Datasheets | PSoC 4 Technical Reference Manuals |
| Development Kit Documentation | |
| CY8CKIT-041-41XX PSoC 4100S CapSense Pioneer Kit | |
| CY8CKIT-041-40XX PSoC 4000S Pioneer Kit | |
| PSoC 4 Kits | |
| Tool Documentation | |
| PSoC Creator | Look in the Downloads tab for Quick Start and User Guides |

Document History

Document Title: CE224821 – PSoC 4 CapSense Touchpad with Gestures

Document Number: 002-24821

| Revision | ECN | Submission Date | Description of Change |
|----------|---------|-----------------|--|
| ** | 6385725 | 05/06/2019 | New code example |
| *A | 6951387 | 08/26/2020 | Minor updates to links. Update documentation to show CY8KIT-041-41XX as the default device. |

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](#) | [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
An Infineon Technologies Company
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2019-2020. This document is the property of Cypress Semiconductor Corporation and its subsidiaries ("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. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, "Security Breach"). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. 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. "High-Risk Device" means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. "Critical Component" means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, 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 any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress's published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

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.