CE224820 – PSoC 4 CapSense Slider with Gestures

Objective

This code example demonstrates the implementation of gesture detection in PSoC® 4 with a CapSense® linear slider using the PSoC Creator™ CapSense Component.

Requirements

Tool: PSoC Creator 4.2

Programming Language: C (Arm® GCC 5.4.1)

Associated Parts: PSoC 4000S and PSoC 4100S devices

Related Hardware: CY8CKIT-149 PSoC 4100S Plus Prototyping Kit

Overview

CapSense Sliders are used for controls requiring gradual adjustments. Examples include a lighting control, volume control, graphic equalizer, and speed control. With gesture detection capability on the CapSense slider, a slider can detect a touch, position movement as well as gestures, which allows the implementation of advanced and sophisticated capacitive sensing-based user interfaces.

This code example demonstrates the implementation of gesture detection on CapSense linear slider using the CapSense Component. To demonstrate the gesture functionality, PSoC 4 controls three LEDs, whose brightness is controlled based on the position detected on the slider and turned ON and OFF based on the detected gestures. Alternatively, the CapSense software tuner can be used for real-time tuning and monitoring of detected gestures.

The CapSense Component requires a time stamp implemented in application firmware for the gesture detection algorithm. This code example also demonstrates three different ways to implement the time stamp for the CapSense Component.

Hardware Setup

This code example is set up to use CY8CKIT-149 by default, but this code example can be used with other PSoC 4 devices. To see how to migrate this code example to a new device, see Reusing this Example.

Software Setup

No software setup required.

Operation

1. Connect CY8CKIT-149 to your computer using a USB cable.
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 the slider once and confirm that the LED connected to P5[5] turns OFF; tap it again and confirm that the LED turns back ON.
4. Slide your finger slowly up and down the slider and confirm that the LED connected to P5[5] changes brightness.
5. Flick your finger across the slider starting on the left-hand side and slide right quickly, confirm that the LED connected to P1[0] turns OFF; doing this again will turn the LED back ON.
6. Flick your finger across the slider starting on the right-hand side and slide left quickly, confirm that the LED connected to P2[2] turns OFF; doing this again will turn the LED back ON.

7. In PSoC Creator, 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 400 kbps. 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. Note that when the tuner is used, LEDs will only respond correctly to a gesture when using `USING_SYS_TICK_CALLBACK`. For directions on how to use this and for more information see Timestamp.

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 the screen as shown in Figure 1. All CapSense Gesture settings can be manipulated as seen in Components and Settings.

Figure 1. Flick Left, Click, Flick Right Gestures



Timestamp

CapSense gestures use a timestamp to determine what gesture was detected. This code example describes three ways to accomplish this. This can be changed by changing the `#define TIMESTAMP_METHOD` to `USING_SYS_TICK_CALLBACK`, `USING_MAIN_LOOP`, or `USING_APP_TIMESTAMP`. Note that only `USING_SYS_TICK_CALLBACK` will properly control the LEDs while using the tuner. This is because the callback will update the timestamp based on the system clock. The other two methods update only when their functions are called, which means that if the time between function calls changes, for example taking time to send data to the tuner, the timestamp controlling the gestures is inaccurate.

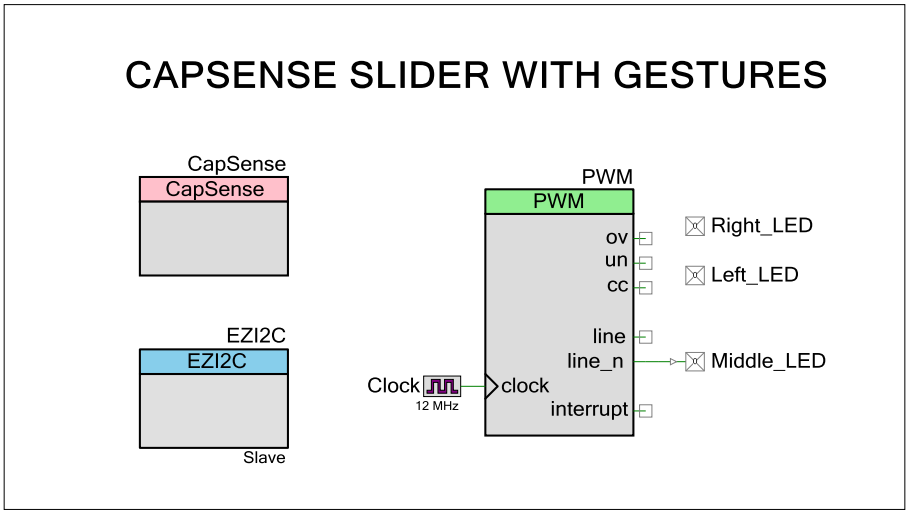
CapSense Slider and Gestures

In the CE224820_CapSense_Slider_Gestures example, the following functions are performed:

1. Initialize and start all hardware components.
2. Link the communication buffer for the EZI2C Component to the CapSense data structure.
3. Set up one of the three different timestamp methods depending on the value of `#define TIMESTAMP_METHOD:`
   - `USING_SYS_TICK_CALLBACK` increments the timestamp through a callback function using the system clock.
   - `USING_MAIN_LOOP` increments the timestamp every time `CapSense_IncrementGestureTimestamp` is called in the main loop.
   - `USING_APP_TIMESTAMP` increments the timestamp by keeping a counter, which is then passed into the `CapSense_SetGestureTimestamp` function.
4. The CapSense Component scans all widgets, and when complete, processes all data.
5. Store the gesture and use it in a switch statement to control the LEDs.
6. Send all data to the tuner.
7. Scan CapSense widgets before restarting the loop.

Figure 2 shows the top-level design of the PSoC Creator project:

Figure 2. CE224820_CapSense_Slider_Gestures 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	Processes all data from the sensors	For Linear slider, see Figure 3. For Click Gesture settings, see Figure 4. For Swipe Gesture settings, see Figure 5.
EZI2C	EZI2C	Transmits data from the kit to the tuner	Change the Data rate to **400 (kbps)** and the Sub-address size to **16.**
PWM	PWM	Controls the duty cycle of the middle light	Under the PWM tab deselect all interrupts, change the period to **10000**, and the Compare value to **5000**.

Figure 3. CapSense Slider Settings

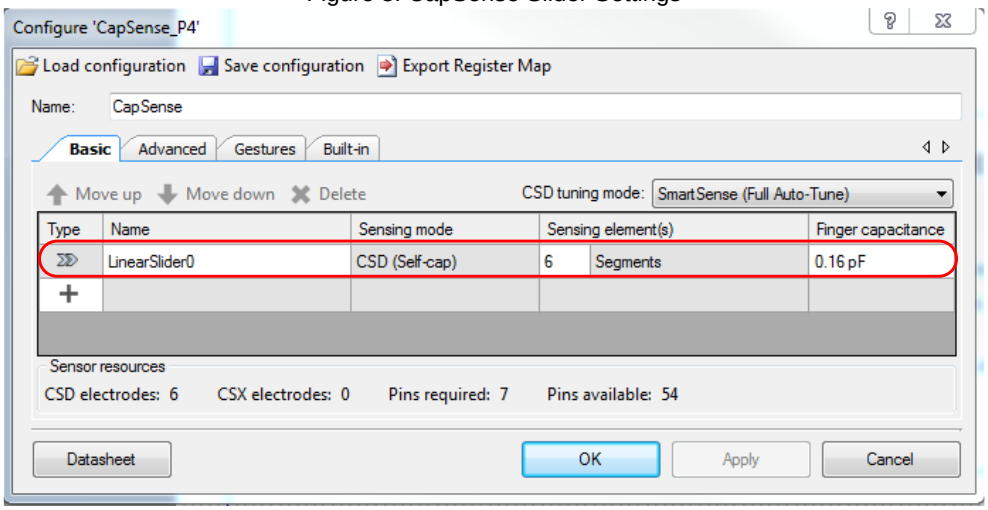


Figure 4. CapSense Click Gestures Tab Configuration

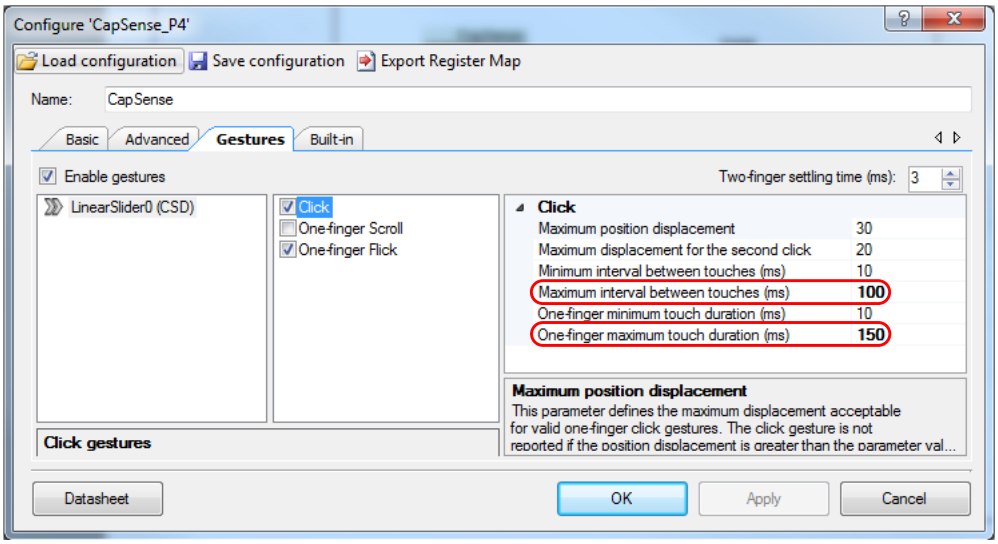
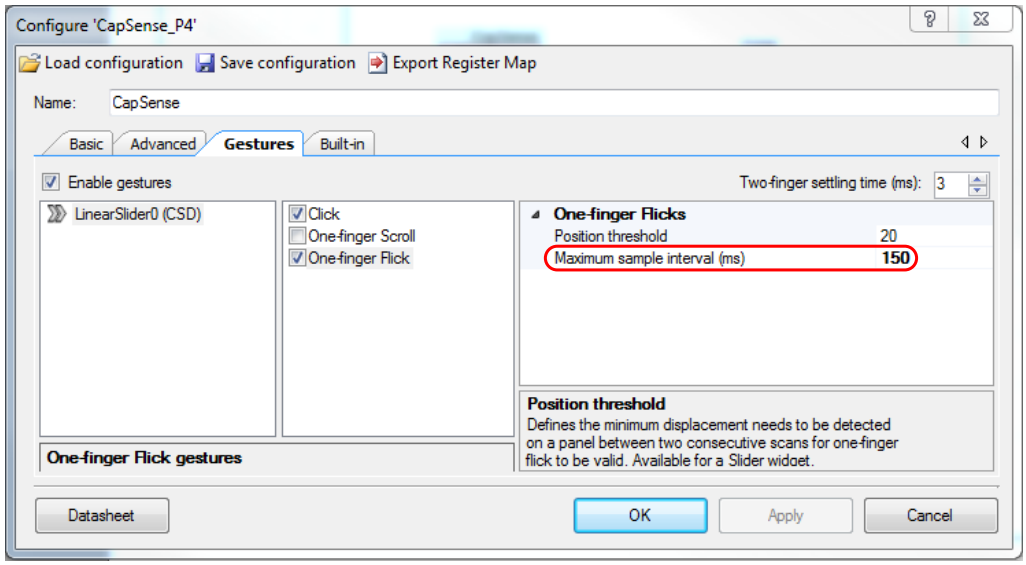


Figure 5. CapSense Swipe Gestures Tab Configuration



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

Reusing this Example

This code example can be used with two kits, CY8CKIT-149 or CY8CKIT-145-40xx. Ensure the following before porting this code example:

1. All the pins are unlocked in Design Wide Resources.
2. If using CY8CKIT-149, ensure that the number of elements is six under Basic in the CapSense Component configuration. If using CY8CKIT-145-40xx, ensure that the number of elements is five. Figure 3 shows how to change the number from **six** to **five** under Sensing elements.

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

In some cases, a resource used by a code example (for example, an IP 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
Code Examples	
CE224821 – CapSense Touchpad and Gestures	Shows how to use Gestures on a touchpad
PSoC Creator Component Datasheets	
CapSense	CapSense Component datasheet for more information
Device Documentation	
PSoC 4 Datasheets	PSoC 4 Technical Reference Manual
Development Kit Documentation	
CY8CKIT-149 PSoC 4100S Plus Prototyping Kit	
CY8CKIT-145-40xx PSoC 4000S Prototyping Kit	
PSoC 4 Kits	
Tool Documentation	
PSoC Creator	Look in the **Downloads** tab for Quick Start and User Guides

Document History

Document Title: CE224820 – PSoC 4 CapSense Slider with Gestures

Document Number: 002-24820

Revision	ECN	Submission Date	Description of Change
**	6290563	08/31/2018	New code example
*A	6673695	09/16/2019	Update document for clarity, update links.

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
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2018-2019. 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.