CapSense_CSD Tuner Design example project

1.10

Features

- Sensing elements: 2 Buttons and Linear Slider
- Tuner GUI scan results visualization
- Tuning parameters using Tuner GUI

General Description

This example project demonstrates the CapSense CSD component configured with 2 buttons and a linear slider. The Tuner GUI displays the scanning results.

Development kit configuration

This project is written for 2 Buttons and a Linear Slider as available on the Cypress kit CY8CKIT-001. The modulator capacitor Cmod is on board at port P2[7] (CY8C38 Family Processor Module CY8CKIT-009).

Note To use this project with the CY8C55 Family Processor Module CY8CKIT-010, please re-assign the Cmod to port P15[5].

Project configuration

The example project consists of the EzI2C and CapSense CSD components. The top design schematic is shown in **Figure 1**. The EzI2C component is used to establish communication between the Tuner GUI and the PSoC.

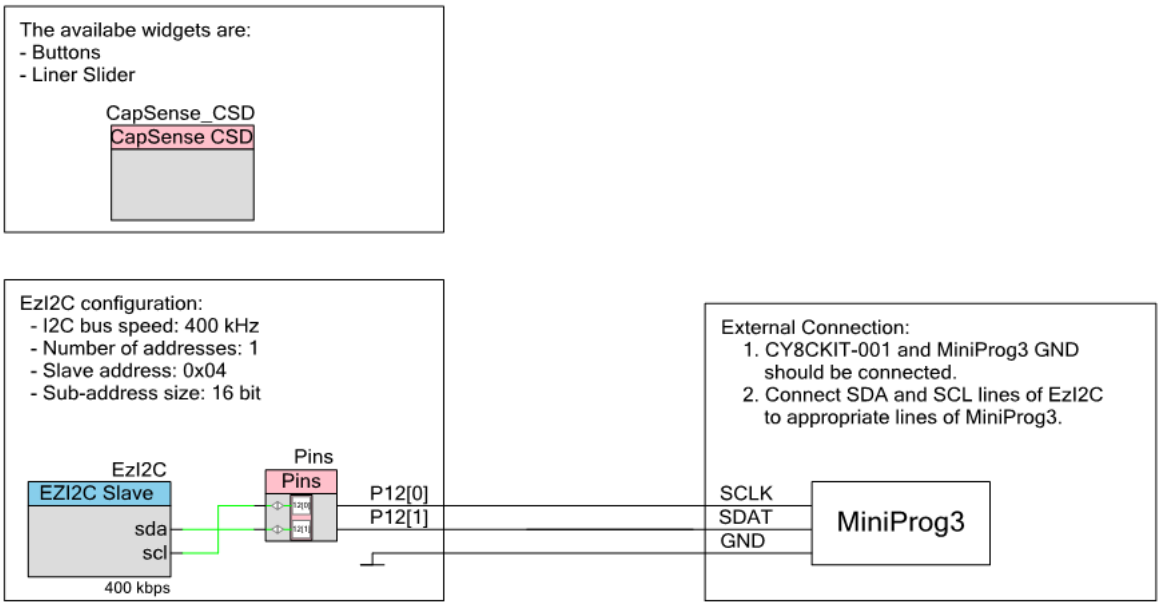

Figure 1 Top design schematic

The EzI2C component is configured: **I2C bus speed**: 400 kHz, **Number of addresses**: 1, **Slave address**: 0x04, **Sub-address size**: 16 bit.

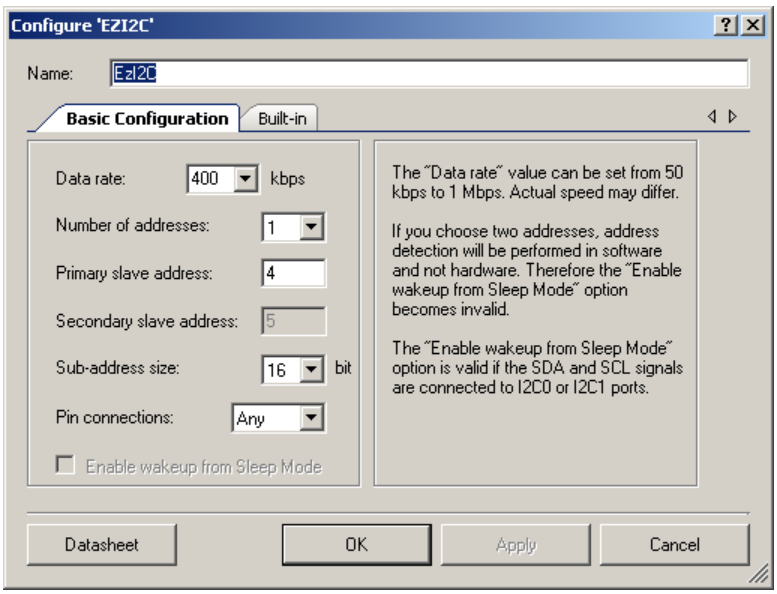

Figure 2 The “EzI2C Basic Configuration Tab”

The CapSense_CSD component is configured with 2 buttons and linear slider.

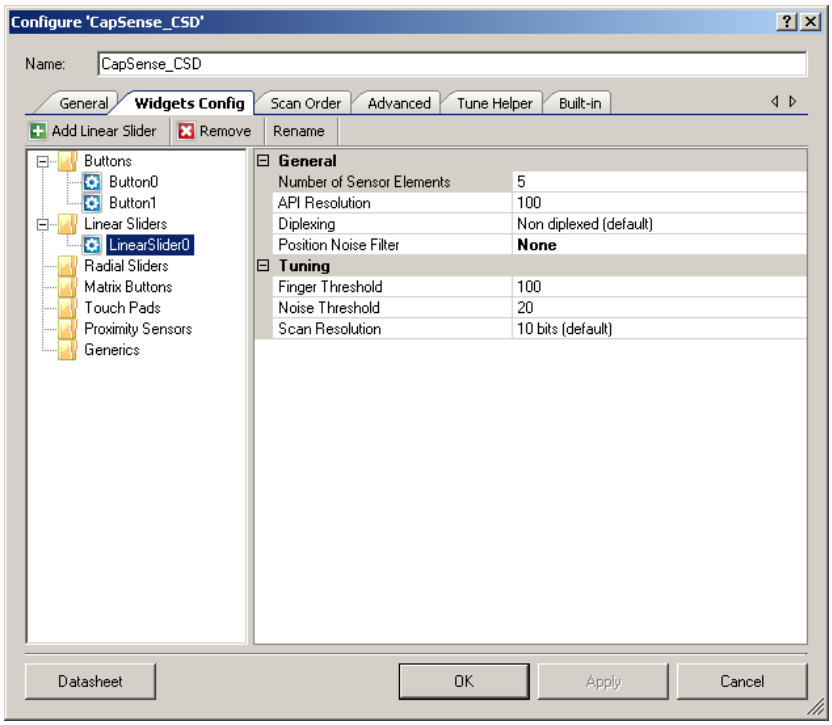

Figure 3 The “CapSense CSD Widget Config Tab”

The clock system configuration is shown in **Figure 4**.

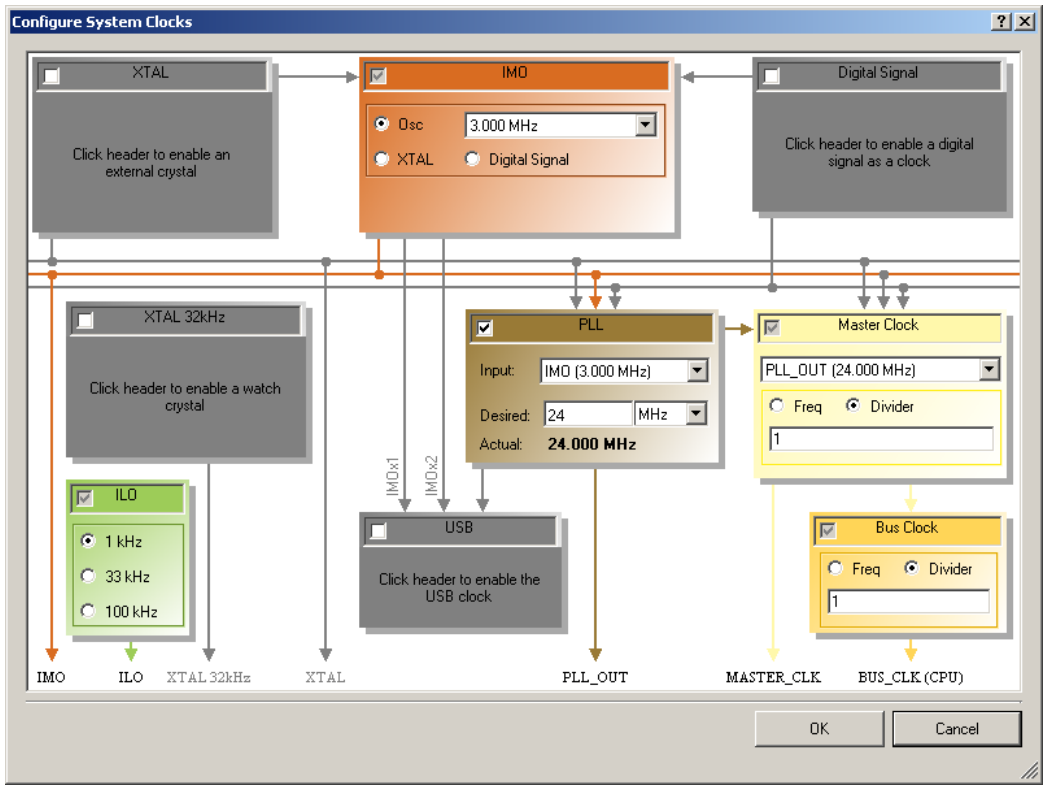

Figure 4 The “Configure System Clocks” window

Start project step-by-step

Build the design and program the PSoC device

Refer to PSoC Creator Help as needed.

Launch Tuner GUI

Right-click and select **Launch Tuner** from the CapSense_CSD instance context menu.

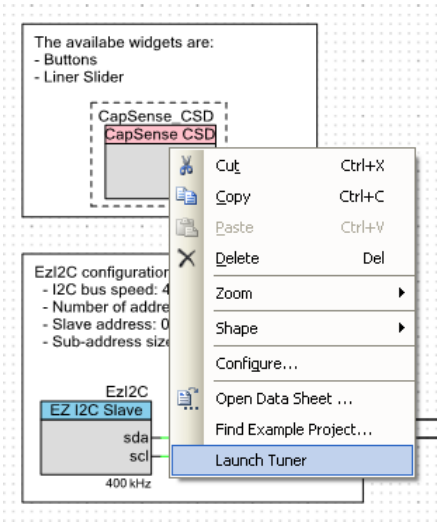

Figure 5 The CapSense CSD instance context menu

The Tuner GUI application opens. The 2 buttons and linear slider widgets are shown.

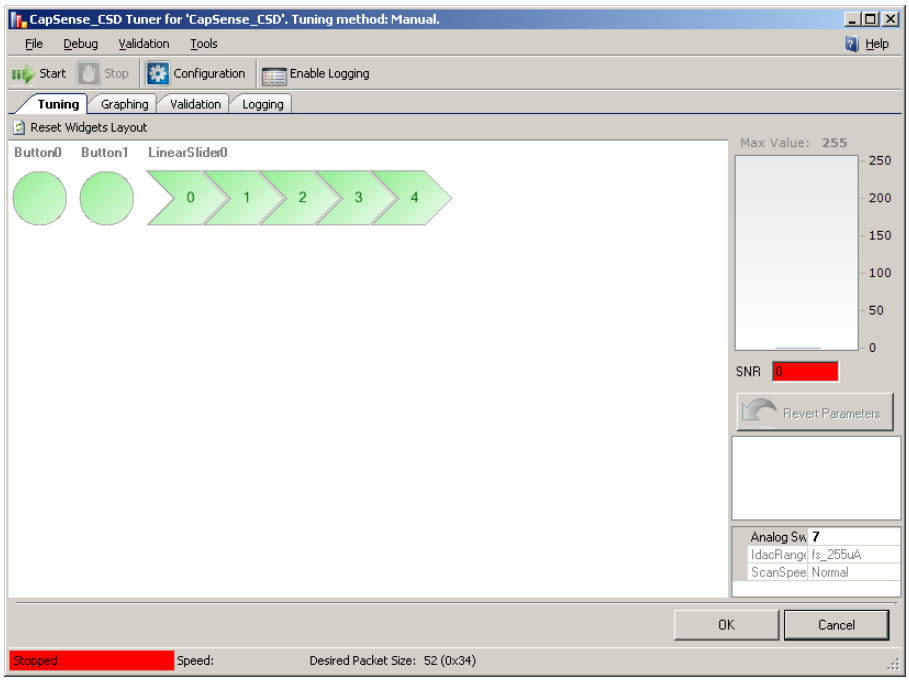

Figure 6 Tuner main window

Power cycle the device

External Connection:

1. CY8CKIT-001 and MiniProg3 GND should be connected.
2. Connect SDA and SCL lines of EzI2C to appropriate lines of MiniProg3. In current design SCL is assign to P12[0] pin and SDA to P12[1] pin.

Configure communication parameters

1. Click Configuration to open the Tuner Communication dialog.

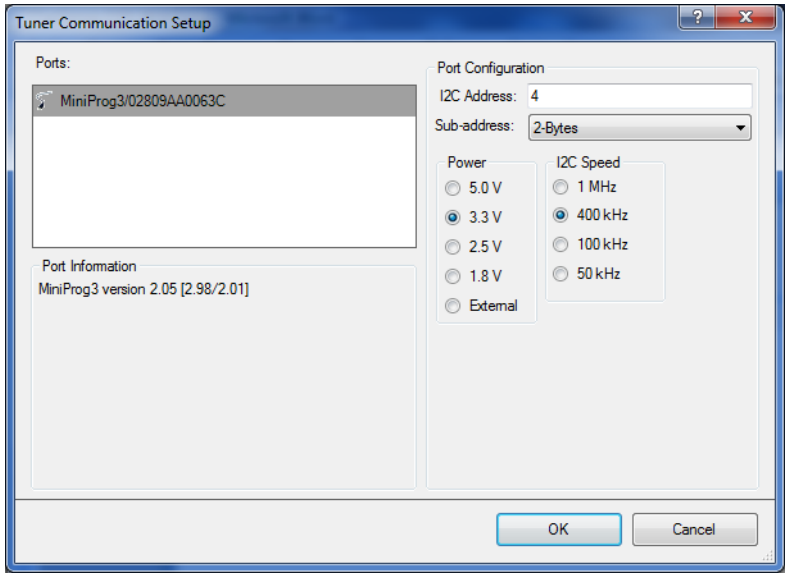

Figure 7 Tuner communication setup dialog

2. Set the communication parameters.

Important: These properties must be identical to EzI2C component: **I2C Bus Speed**, **I2C Address**, **Sub-address.**

Start tuning

Click **Start** on the Tuning GUI. The scanning results will be shown for all sensing elements.

Expected results

The scanning results are showing for all sensing elements.

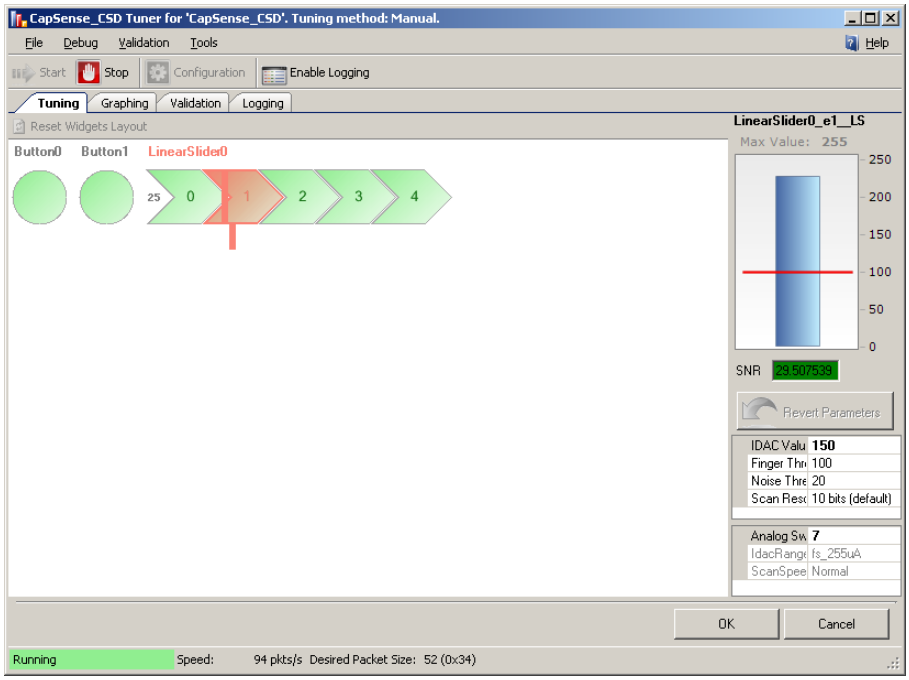

Figure 8 Scanning results

Use Graphing Tab to display detail information for scanning results of selected sensors.

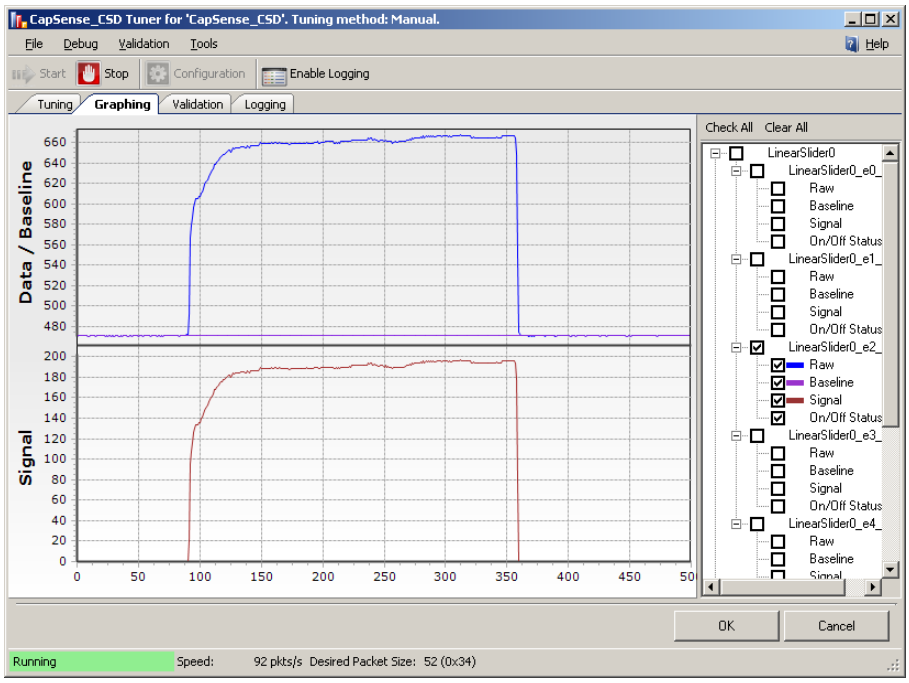

Figure 9 Graphing

© Cypress Semiconductor Corporation, 2009-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.