



PSoC® 4

CapSense® Tuning Guide

Document Number: 001-92505, Rev. *C

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408.943.2600
<http://www.cypress.com>

© Cypress Semiconductor Corporation, 2014-2017. 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.

Contents



1	Introduction	5
	Conventions	5
	References	5
	Revision History	5
2	CapSense Tuning Process	6
	Create a Design in PSoC Creator	6
	Place and Configure an EZI2C Component	6
	Place and Configure the CapSense Component	7
	Selecting Auto (SmartSense)	8
	Configure Widgets	9
	Configure Tuner Helper	9
	Add Code	10
	Build the Design and Program the PSoC Device	10
	Launch the Tuner application	11
	Configure Communication Parameters	11
	Start Tuning	12
	Edit CapSense Parameter Values	12
	Repeat as Needed	12
	Close the Tuner application	12
3	CapSense Validation Process	13
	Start Validation	13
	Stimulation Sensors	14
	Validation Displays	15
	Validation Results	16
	Manual Tuning Process	17
4	Capsense CSD User Interface	20
	General Interface	20
	Work area	20
	Menus	21
	Toolbar	21
	Tool Windows	21
	Status Bar	22

Tuning Tab	22
Sensor Properties Tool Window	23
Graphing Tab	24
Chart area	24
Graphing Properties Tool Window	24
Logging Properties Tool Window	24
Validation Tab	25
Top panel controls:	25
Validation Advanced Properties	26
Debugging Tab.....	27
Save/Load Settings Feature	28
5 CapSense Gesture User Interface	29
General Interface	29
Data Monitor.....	29
Gesture view	30
Event History.....	30
Sensor Properties	30
Gesture Properties	31
Filter Properties.....	33
Tuning Tab	34
Graphing Tab.....	35
Validation Tab.....	35
Debugging Tab	35
Image Tool Tab.....	35

1 Introduction



This PSoC 4 CapSense Tuning Guide includes instructions and information that will help you use the CapSense Tuner.

Conventions

The following table lists the conventions used throughout this guide:

Convention	Usage
Courier New	Displays file locations and source code: C:\...cd\icc\, user entered text
<i>Italics</i>	Displays file names and reference documentation: <i>sourcefile.hex</i>
[bracketed, bold]	Displays keyboard commands in procedures: [Enter] or [Ctrl] [C]
File > New Project	Represents menu paths: File > New Project > Clone
Bold	Displays commands, menu paths and selections, and icon names in procedures: Click the Debugger icon, and then click Next .
Text in gray boxes	Displays cautions or functionality unique to PSoC Creator or the PSoC device.

References

This guide is one of a set of documents pertaining to PSoC Creator and PSoC devices. Refer to the following other documents as needed:

- PSoC 4 CapSense CSD and PSoC 4 CapSense Gesture Datasheets
- PSoC Creator Help
- PSoC Technical Reference Manual (TRM)

Revision History

Document Title: PSoC® 4 CapSense® Tuning Guide		
Document Number: 001-92505		
Revision	Date	Description of Change
**	5/23/14	New document.
*A	7/23/14	Minor edits and screen capture updates to match GUI changes.
*B	11/19/14	Fixed typos and minor edits.
*C	11/30/17	Minor edits.

2 CapSense Tuning Process



The CapSense Tuner assists in tuning the CapSense component to the specific environment of the system when using the “Manual with run-time tuning” Tuning method. The tuner can also display the tuning values (read only) and performance when using the SmartSense Tuning method. No tuning is supported when the component is set to “Manual” as all parameters are stored in flash and are read only for minimum SRAM usage.

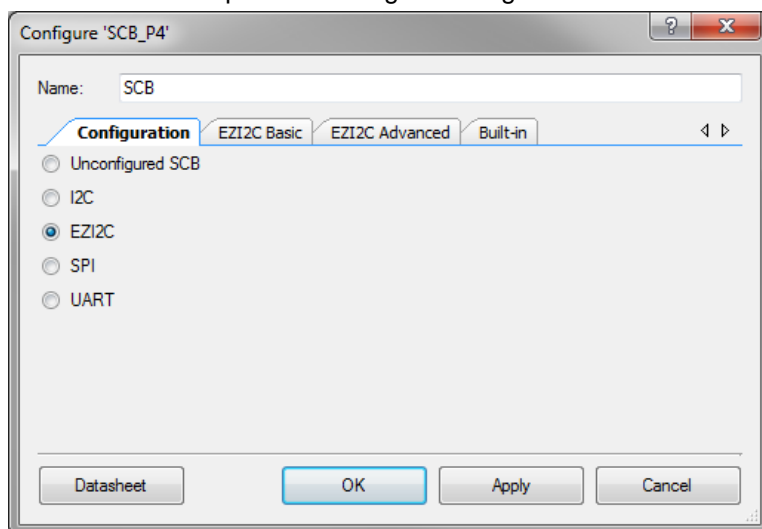
The following is the typical process for using and tuning a CapSense component:

Create a Design in PSoC Creator

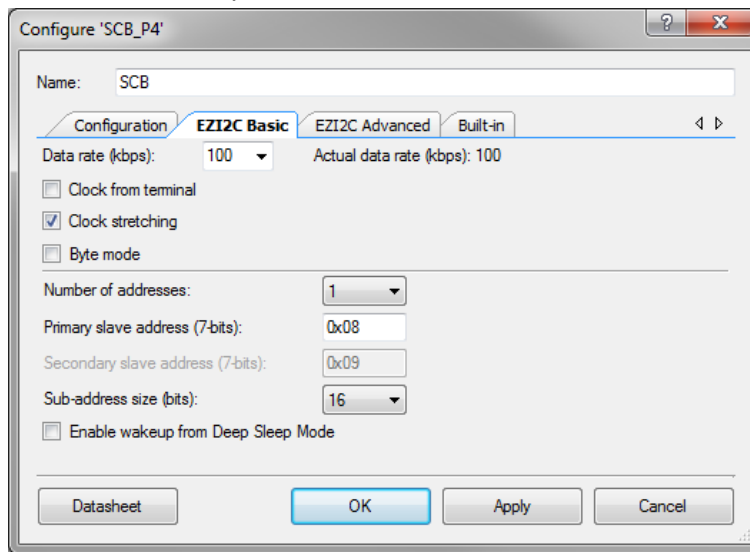
Refer to the PSoC Creator Help as needed.

Place and Configure an EZI2C Component

1. Drag a SCB component from the component catalog onto your design.
2. Double-click it to open the Configure dialog.



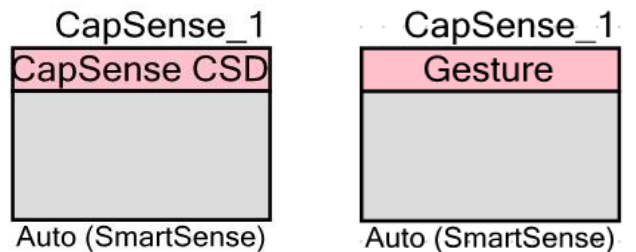
3. Select the EZI2C option and then click the **EZI2C Basic** tab.



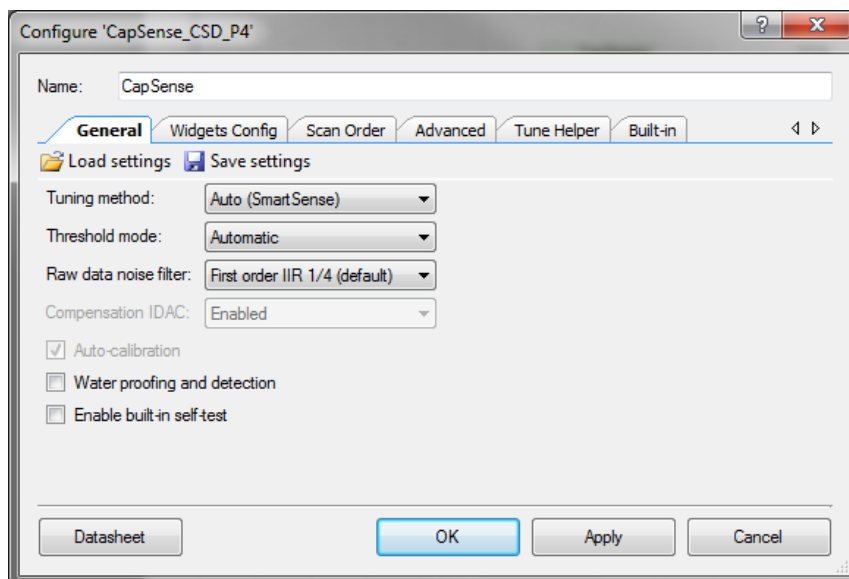
4. Change the parameters as follows:
 - ☐ Set the Sub-address size (bits) to 16.
 - ☐ Change the instance name to match the name used on the CapSense Configure dialog, under the Tuner Helper tab, for the generated APIs to function. See [Configure Tuner Helper](#).

Place and Configure the CapSense Component

1. Drag a CapSense CSD or Gesture component from the Component Catalog onto your design.



2. Double-click it to open the **Configure** dialog.



3. Change parameters as required for your application. Select **Tuning method** as **Manual with run-time tuning** or **Auto (SmartSense)**.

Selecting Auto (SmartSense)

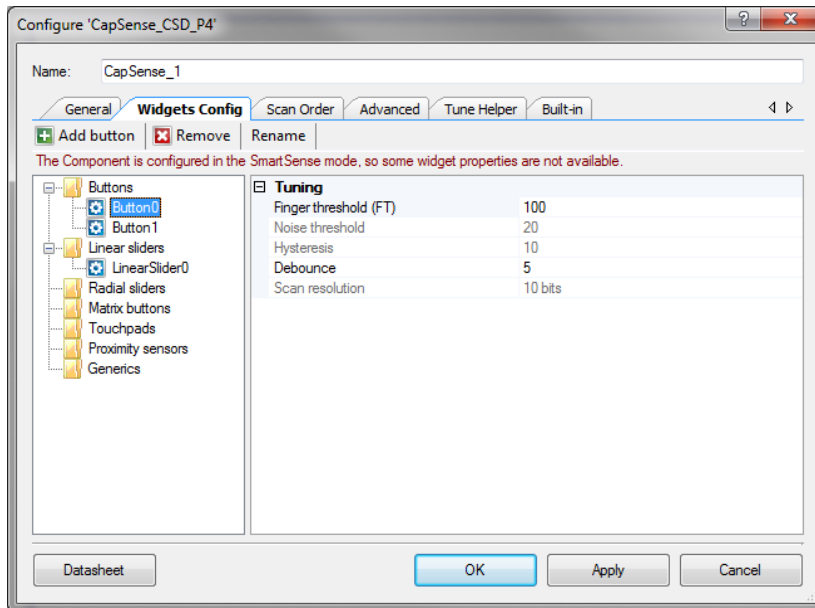
Auto (SmartSense) tunes the CapSense CSD or Gesture component to the specifics of the system automatically. CapSense CSD/Gesture parameters are computed at run time by firmware. Additional RAM and CPU time are used in this mode. Auto (SmartSense) eliminates the error-prone and repetitive process of manually tuning the CapSense CSD/Gesture component parameters to ensure proper system operation. Selecting Auto (SmartSense) tunes the following parameters:

Parameter	Calculation
Finger Threshold ^[1]	Calculated continuously during sensor scanning.
Noise Threshold ^[1]	
Compensation IDAC	Calculated once on CapSense CSD/Gesture startup.
Modulation IDAC	
Sense Clock Divider	
Modulator Clock Divider	
Negative Noise Threshold	
Low Baseline Reset	

¹ Finger Threshold and Noise Threshold are calculated continuously when Threshold mode is set to Automatic. For Flexible Threshold mode Finger Threshold is set manually and Noise Threshold = 50% of Finger Threshold.

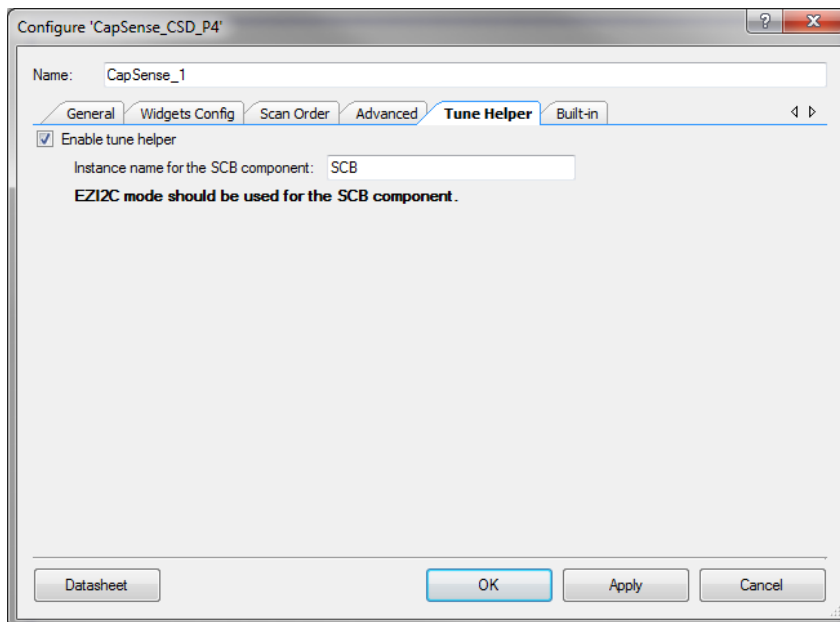
Configure Widgets

Add widgets on the **Widgets Config** tab and configure them.



Configure Tuner Helper

On the **Tune Helper** tab: The **Enable Tune Helper** check box must be selected.



Add Code

Add Tuner initialization and communication code to the projects *main.c* file. Example *main.c* file:

```
void main()
{
    CyGlobalIntEnable;
    CapSense_1_TunerStart();

    /*All widgets are enabled by default except proximity widgets.
    Proximity
    widgets must be manually enabled as their long scan time is
    incompatible
    with the fast response required of other widget types.
    */

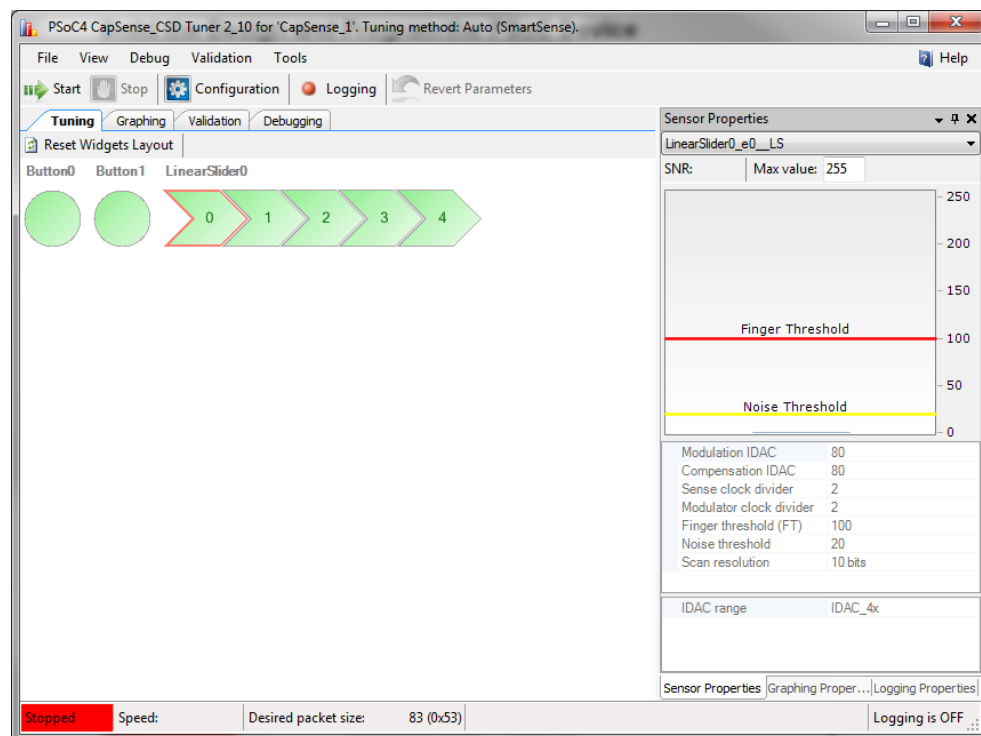
    while(1)
    {
        CapSense_1_TunerComm();
    }
}
```

Build the Design and Program the PSoC Device

Refer to PSoC Creator Help as needed.

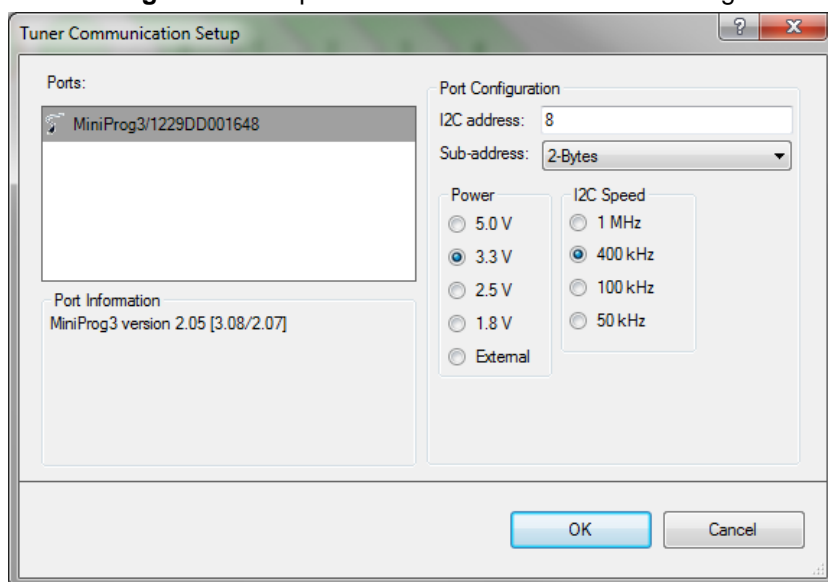
Launch the Tuner application

Right-click the CapSense CSD/Gesture component icon and select **Launch Tuner** from the context menu. The Tuner application opens.



Configure Communication Parameters

1. Click **Configuration** to open the Tuner Communication dialog.



- Set the communication parameters and click OK.

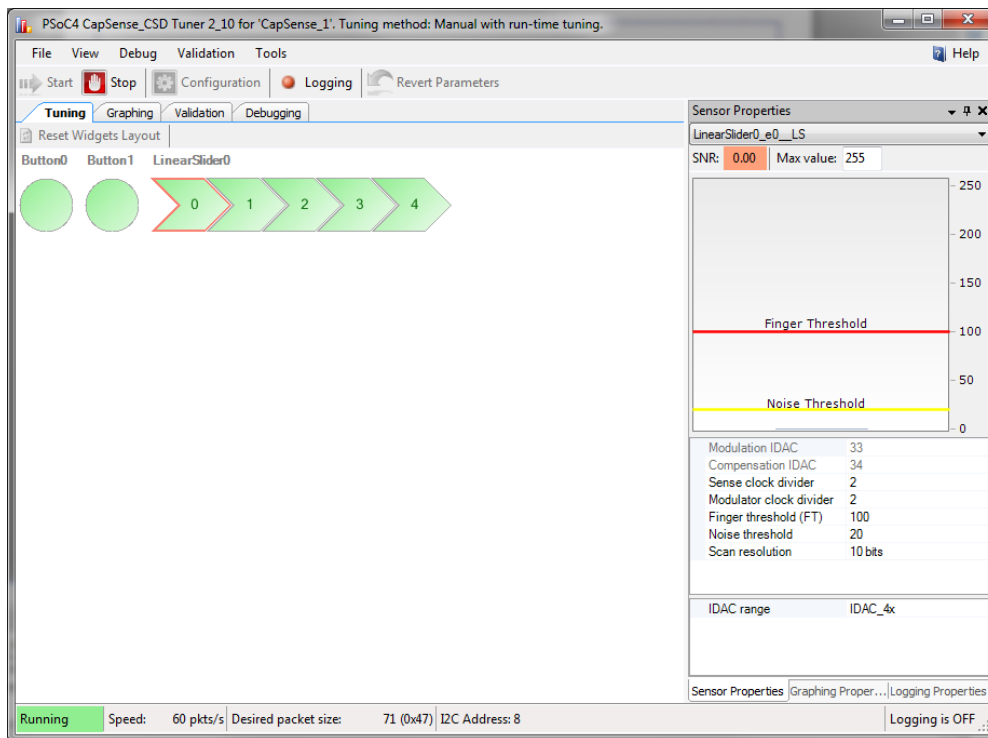
Important The fields: **I2C address**, **Sub-address**, and **I2C speed** must be identical to those in the SCB component: **Primary slave address**, **Sub-address size**, and **Data rate**, respectively. **Sub-address** must be set to 2-Bytes.

Start Tuning

Click **Start** on the tuning GUI. All of the CapSense elements start to show their values.

Edit CapSense Parameter Values

Edit a parameter value for one of the elements, and it is automatically applied after you press the **[Enter]** key or move to another option. The GUI continues to show the scanning data, but it is now altered based on the application of the updated parameter. Refer to the [Capsense CSD User Interface](#) section later in this document.



Repeat as Needed

Repeat steps as needed until tuning is complete and the CapSense component gives reliable touch sensor results.

Close the Tuner application

Click **File->Apply Changes and Close** and the parameters are written back to the CapSense CSD/Gesture instance. The Tuner application dialog closes.

3 CapSense Validation Process

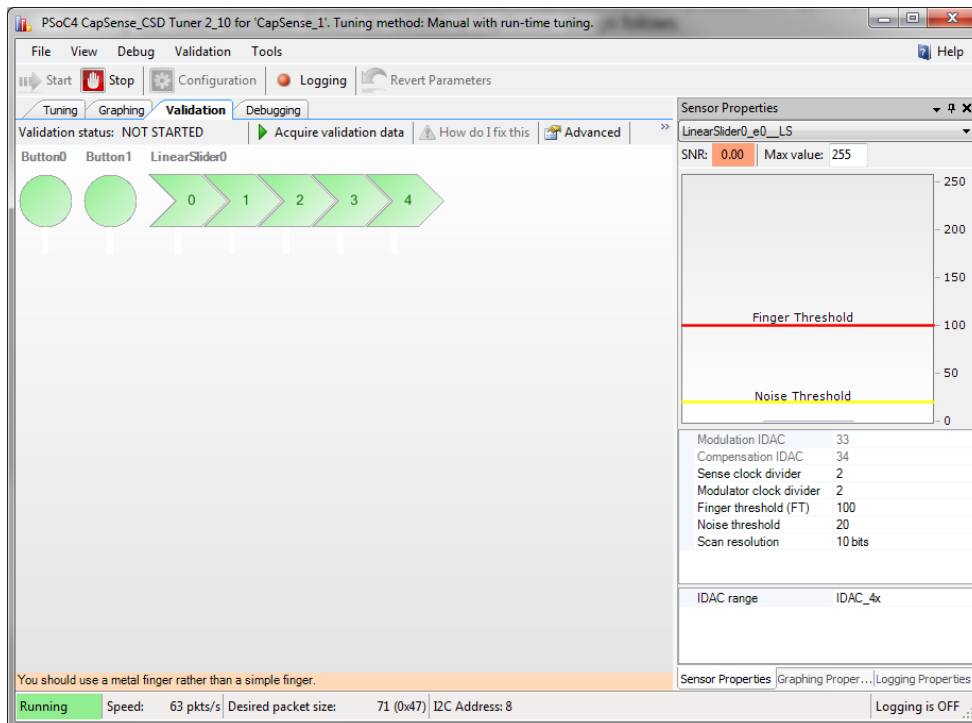


The validation mechanism determines whether the board has been sufficiently tuned. The typical process for using the Tuner Validation feature to validate a CapSense design follows.

Start Validation

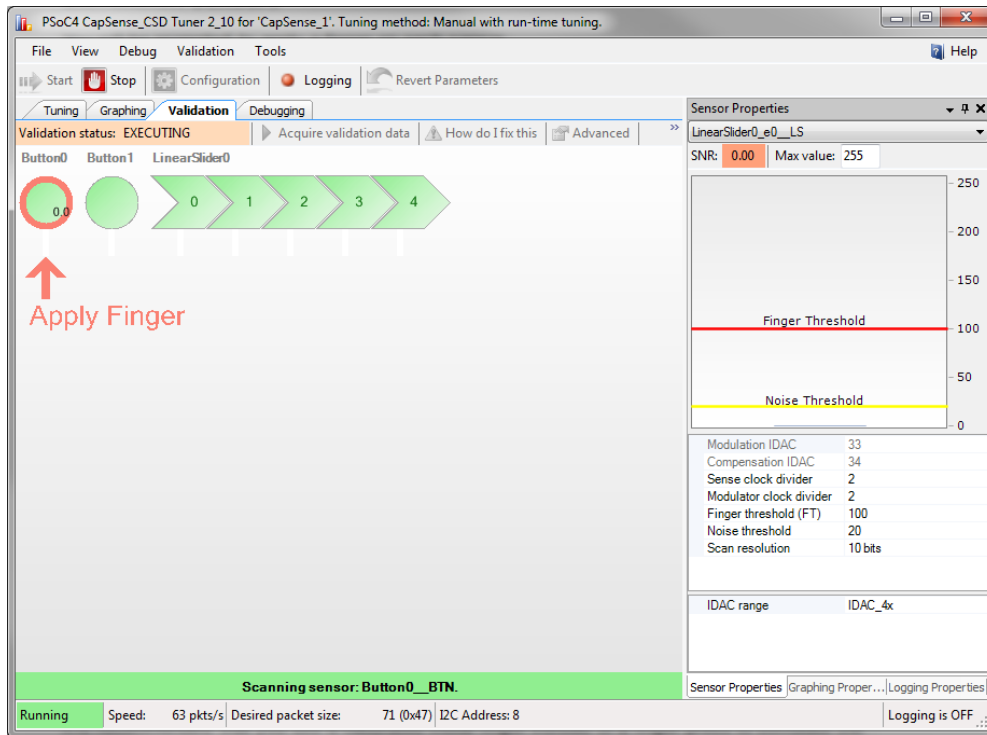
The Tuner and hardware must be ready before you start the scanning process. See the CapSense Tuning Process section to prepare the system for scanning.

On the **Validation** tab, click “Acquire validation data.” Values will begin to appear for all CapSense elements.



Stimulation Sensors

You will be prompted to apply a finger on each sensor.

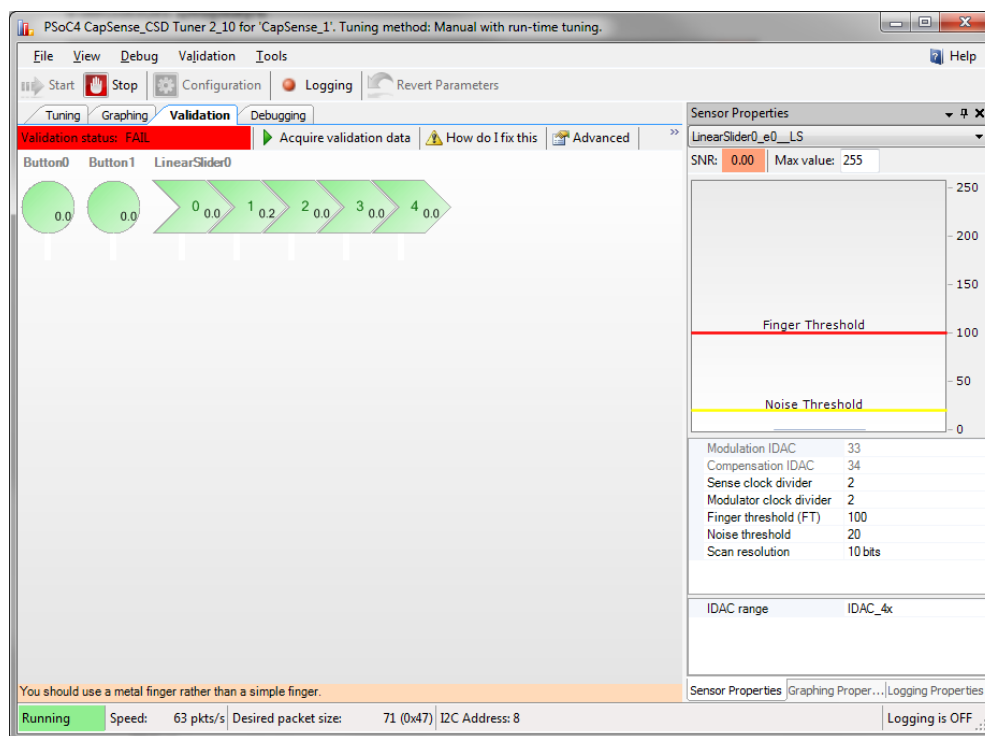


Each time you are prompted to press a CapSense element, a flashing red arrow pointing to the target appears on the layout, with the text **PRESS HERE**. Text appears beneath the Tuner that will guide you through the validation process.

To start scanning for the current sensor, press any key on the keyboard.

It is recommended that you use a calibrated metal finger instead of a finger press to stimulate the sensors.

Validation Displays



SNR warnings appear as follows:

- **Flashing red** highlights surround any CapSense sensor that has an SNR less than the **Sufficient Value** ^[2].
- **Flashing yellow** highlights surround any CapSense sensor that has an SNR between the **Sufficient** ^[2] and **Optimal Values** ^[2].
- **Solid green** highlights surround any CapSense sensor that has an SNR above the **Optimal Value**.

Crosstalk effects warnings appear as follows:

- **Individual Crosstalk Check.** During the validation process, the software monitors all elements other than the one you have been told to stimulate. If an element exhibits difference counts that exceed the **Crosstalk Threshold Percentage** (when not directly stimulated), a crosstalk warning is generated. This is displayed by a flashing line between the element that exhibits the unwanted counts and the element that was stimulated.
- **Worst Case Crosstalk Check.** As each of the individual crosstalk checks are made, the software keeps a record of each difference count measurement. At the completion of the process, worst-case crosstalk estimates are made.

For each sensor, a sum appears that is the number of the crosstalk effects equal to the **Worst Case Crosstalk Sensor Count**. The largest crosstalk value is the first element in the sum, the second largest is the second, and so on. For example: if you have the following crosstalk counts (1,5,3,2,4,1,1,0) and the

² Sufficient and Optimal Values can be defined using the **Validation** menu item **Validation Advanced Properties** (Ctrl + Alt + V).

Worst Case Crosstalk Sensor Count is 2, then the **Worst Case Crosstalk** computation will be $(5 + 4 = 9)$.

If this value exceeds the **Worst Case Crosstalk Threshold**, it is flagged with a **flashing “C” character** in the middle of the sensor display.

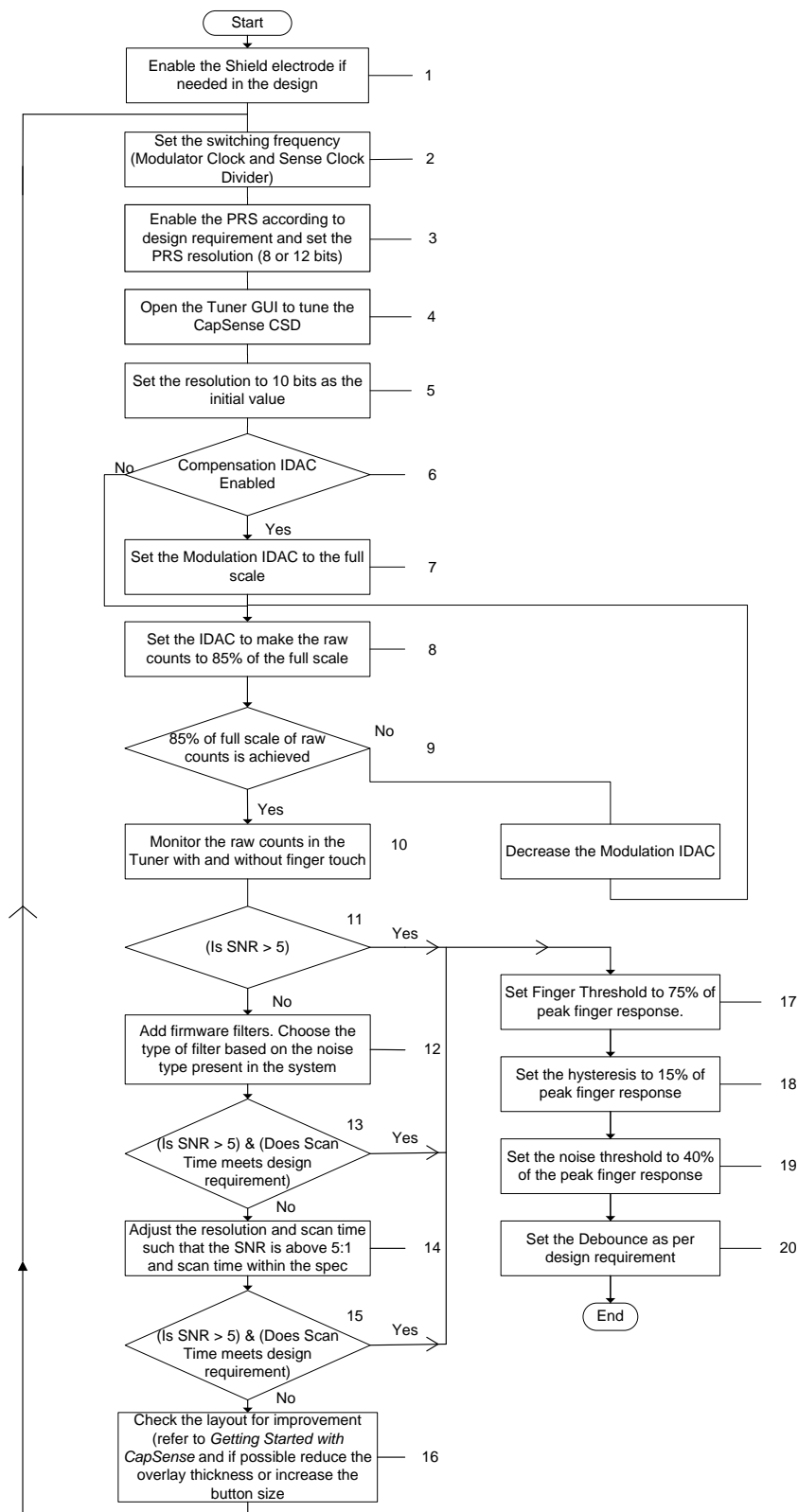
Validation Results

If the validation process uncovers failures, a **Validation Report** will be displayed. This report contains the following information:

- Any SNR values less than the **Optimal Value**
- Any SNR values less than **Sufficient Value**
- Any signals with a worst-case crosstalk failure, and, if so, the crosstalk number

You can also open the Validation Report by clicking the **How do I fix this** button on the **Validation** tab.

Manual Tuning Process



1. The Shield is enabled or disabled depending on the design requirements. A shield is useful in applications where the sensor overlay may become wet (see AN2398), to shield against EMI, or to mitigate excessively high C_p . The shield can be routed to any pin. For more information, see the Shield Electrode Section section in the Component Datasheet.
2. The switching frequency of each capacitive sensor should be set such that the sensor completely charges and discharges. **Modulator Clock Divider** and **Sense Clock Divider** determine the frequency at which the sensor capacitor is switched. **Modulator Clock** (Scan Clock) is the primary clock source for the CapSense component. In the PSoC 4100/PSoC 4200 device, the **Sense Clock Divider** (prescaler) divides the Modulator Clock to produce the switching clock used to charge and discharge each sensor. If the sensor is not charging and discharging completely, reduce the switching frequency by increasing the **Sense Clock Divider**.

If the Parasitic Capacitance of sensor is known the **Sense Clock Divider** can be calculated so that the period of the Sense Clock is greater than $10RC_p$ or Sense Clock Frequency $< 1/(RC_p)$, where R – the total resistance of Resistor, Analog MUX bus and Switch (see SW3 in the CapSense Analog System section of the CapSense CSD/Gesture component datasheet) connected in series with the Sensor Parasitic Capacitance C_p .

Note Parasitic Capacitance of the sensor can be measured using the built-in test `Capsense_GetSensorCp()` API.

To test whether sensors are charging and discharging completely, probe each sensor pin. Note that, while observing the voltage on the sensor capacitor with an oscilloscope probe, the probe capacitance is added to the sensor parasitic capacitance. Using probes in 10x mode reduces the capacitance of the probes. Use a FET input probe, if available. Make sure that the sensor is charging and discharging completely; if not, increase the **Sense Clock Divider** value in the component configuration. Because this parameter cannot be changed in the Tuner GUI, it must be set in the component configuration. Therefore, when this value is changed in the component **Configure** dialog, the project should be built again and programmed to the device.

3. The PRS is enabled by default to reduce the effects of external EMI on CapSense as well as to reduce sensor scan emissions. It is enabled in designs prone to EMI effects. The resolution of the PRS is set to 8 or 12 bits depending on the scan time. For longer scan times, use PRS 12; for shorter scan times (< 380 us), use PRS 8.
4. Open the CapSense Tuner GUI, and set the resolution to 10.
 Increasing the resolution and Sense Clock frequency give better sensitivity but they increase the scan time. Therefore, there is a tradeoff between scan time and sensitivity.
 The resolution of 10 is a good starting value in the tuning process, although lower resolutions of 8 and 9 can also be used as initial values if the design has thin overlays less than 1 mm.
5. Enable Compensation IDAC if needed.
6. If Compensation IDAC is enabled then set the Modulation IDAC to the full scale.
7. **Note** If Auto-calibration is enabled the steps 6-9 must be skipped. In this case the IDACs will be calibrated automatically to achieve the raw counts to be equal to 70 percent of the full scale value. When both IDACs are enabled they are calibrated to have equal values.
8. Change the Compensation IDAC (or Modulation IDAC if Compensation IDAC is disabled) value in the GUI until the raw counts reach 85 percent of the full scale value. The full scale value is $2^{\text{Resolution}}$. Note that decreasing the IDAC value increases the raw counts and vice versa. If it is not possible to achieve 85 percent with any of the IDAC values, then you should change the IDAC range in the component configuration. The IDAC range cannot be changed in the Tuner GUI; it should be changed in the component **Configure** dialog. When the value is changed in the **Configure** dialog, you should build the project again and program it to the device.

9. If it is not possible to achieve 85 percent with any of the Compensation IDAC values, then you should decrease the Modulation IDAC value and repeat step 8.
10. Monitor the raw counts with and without a finger present. Note the peak-to-peak noise and peak finger response. Calculate the SNR as:

$$\text{SNR} = \frac{\text{Peak Finger Response}}{\text{Peak to Peak Noise when finger is not present}}$$

11. Also, check whether the scan time requirement for the design is met. The tuned values inside the GUI are updated in the component when the **OK** button is clicked on the GUI. The scan time approximation is calculated by the component based on the parameter settings. The scan time is shown in the scan order tab. If the design has many sensors that have high resolution and low scan speed then the total scan time for all the sensors will result in a long scan interval for the sensors.
12. For a good CapSense design, the SNR should be above 5. Check whether the total scan time fits the design. If the SNR requirement is not met, add firmware filters. See the Filters section of the Component Datasheet and select the type of filter that suits the noise present in the system. For most designs, start with the First Order IIR 1/4 filter because it requires minimal SRAM and gives a fast response.

If the SNR is below 5, increase the resolution, the Modulator Clock Divider, or both. By doing this, the scan time increases. Therefore, the resolution and scan time both should be tuned to achieve the SNR above 5 and keep the scan time below design spec. Recheck the SNR and scan. If you cannot achieve the SNR of 5:1 and keep the scan time within the design spec, look for improvements in the PCB layout or overlay design. Refer to *Getting Started with CapSense* for PCB design guidelines. You can also reduce the overlay thickness or increase the button diameter, which increases the sensitivity.

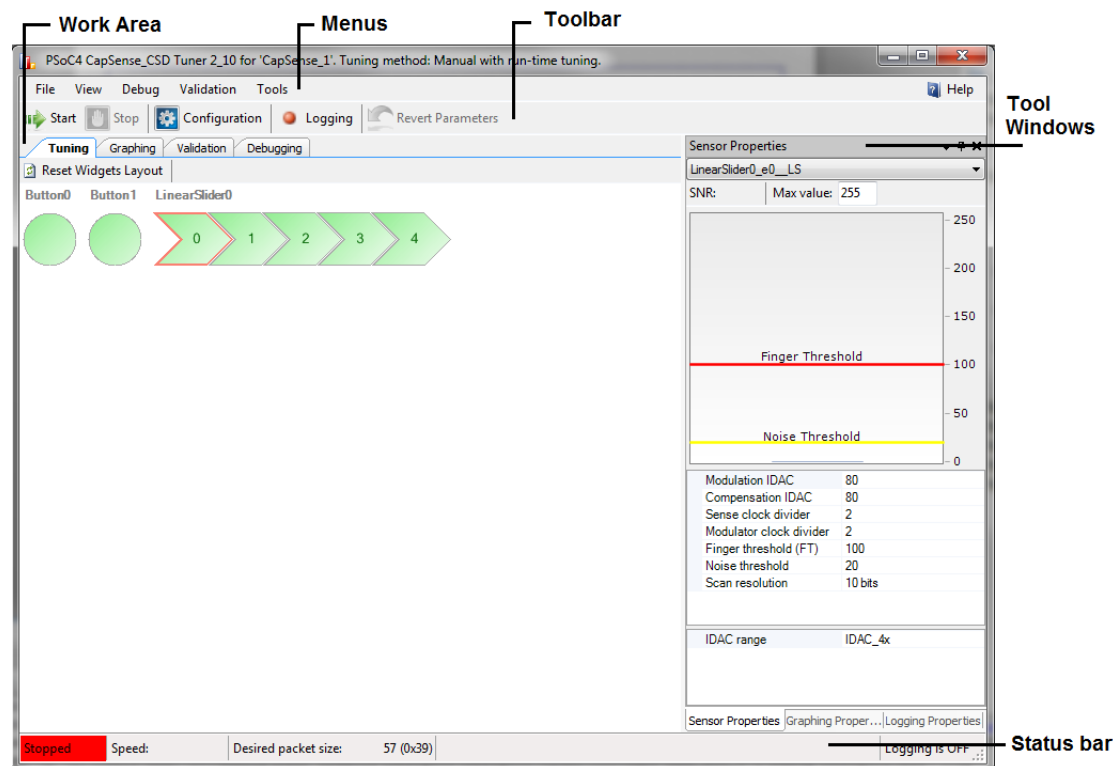
13. After SNR of 5:1 is achieved, set the following firmware parameters.
 - ☐ The Finger Threshold is the parameter the firmware uses as the threshold to determine whether the sensor is active or not. Set this parameter to 75 percent of the peak finger response.
 - ☐ Set the hysteresis to 15 percent of peak finger response.
 - ☐ Set the noise threshold to 40 percent of the finger response.
 - ☐ Debouncing ensures that high-frequency, high-amplitude noise such as an ESD event does not cause button activation. The debouncing value should be a small number such as 3, because the spike or high frequency noise that can trigger a false button touch will not be as wide as two scan lengths. In fast scanning designs, the debounce value should be set to a higher value such as 5.

4 Capsense CSD User Interface



This section describes the CapSense CSD Tuner user interface for the case where there are no Trackpad with gestures widgets selected for the component.

General Interface



Work area

The work area consists of the following tabs:

- **Tuning** – Displays all of the component widgets as configured on a workspace. This allows you to arrange the widgets similarly to the way they appear on the physical PCB or enclosure. This tab is used for tuning widget parameters and visualizing widgets data and states.
- **Graphing** – Displays detailed individual widget data on charts.
- **Validation** – Provides validation functionality.
- **Debugging** – Provides debugging functionality.

Menus

Main menu provides following commands to help control and navigate Tuner:

- **File > Settings > Load Settings from File (Ctrl + O)** – Imports settings from an XML tuning file and loads all data into the Tuner.
- **File > Apply Changes and Close (Ctrl + F4)** – Commits the current values of parameters to the CapSense component instance and exits the GUI.
- **File > Exit (Alt + F4)** – Asks to save changes if there were any, and closes the Tuner.
- **View > Sensor Properties (Alt + 1)** – Shows **Sensor Properties** tool window.
- **View > Graphing Properties (Alt + 2)** – Shows **Graphing Properties** tool window.
- **View > Logging Properties (Alt + 3)** – Shows **Logging Properties** tool window.
- **View > Reset Widgets Layout (Alt + R)** – Duplicates **Reset Widgets Layout** button from **Tuning Tab**.
- **Debug > Start (F5)** – Starts reading and displaying data from the chip. Also starts graphing and logging if configured.
- **Debug > Stop (F6)** - Stops reading and displaying data from the chip.
- **Debug > Configuration (F10)** - Opens the **Communication Configuration** dialog;
- **Validation > Acquire Validation Data (Alt + V)** – Duplicates **Acquire Validation Data** button from **Validation Tab**;
- **Validation > Validation Advanced Properties (Ctrl + Alt + V)** – Duplicates **Advanced** button from **Validation Tab**;
- **Validation > How do I fix this (Ctrl + H)** – Duplicates **How do I fix this** button from **Validation Tab**;
- **Tools > Enable Logging** - Enables logging of data received from the device to a log file.

Toolbar

Contains frequently used buttons that duplicate main menu items:

- **Start** – Duplicates **Debug > Start** menu item.
- **Stop** – Duplicates **Debug > Stop** menu item.
- **Configuration** – Duplicates **Debug > Configuration** menu item.
- **Logging** – Enables data logging into a csv file. Data for logging and logging properties can be selected in the [Logging Properties Tool Window](#).
- **Revert Parameters** – Resets the parameters to their initial values and sends those values to the chip. Initial values are what were displayed when the GUI was launched.

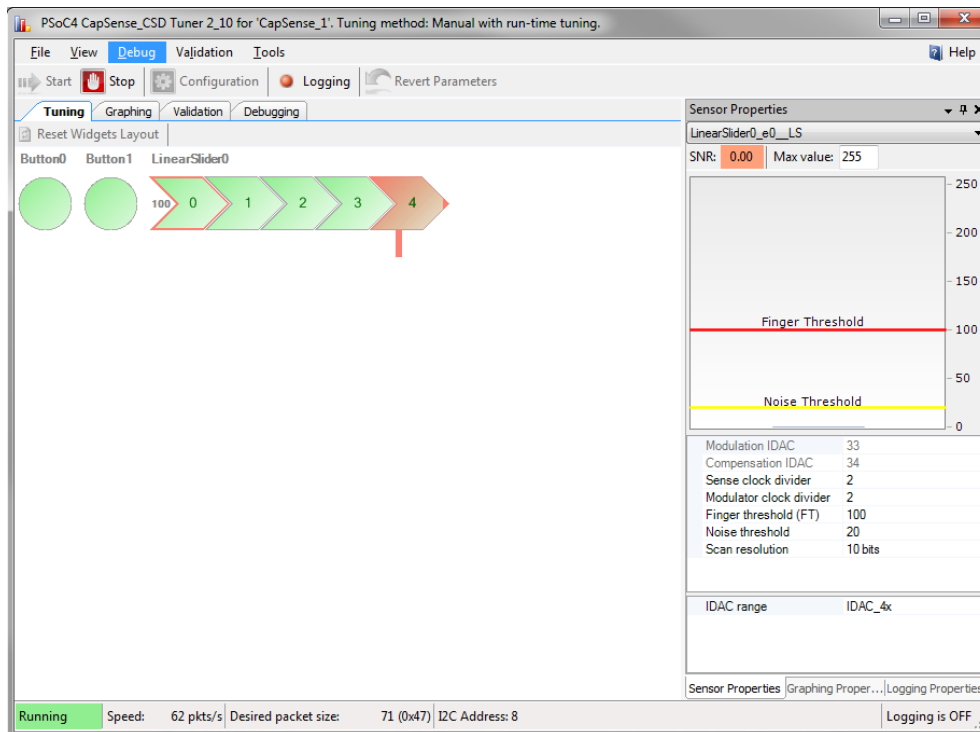
Tool Windows

Tool windows are windows that can be shown at any time not depending on tab which is selected at the moment. Also tool windows can be docked to the right, left, top or bottom side of the Tuner. Windows can be docked all together by dragging title or separately by dragging specific page in the bottom of the tool window.

Status Bar

The status bar displays the current state of communication between the Tuner and the device.

Tuning Tab



- **Widgets schematic** – Contains a graphical representation of all of the configured widgets. If a widget is composed of more than one sensor the individual sensors may be selected for detailed analysis. Every widget is movable within the schematic.
- **Reset Widgets Layout button** – Moves widgets to default positions within the schematic.
- **Widget controls context menu** (this functionality applies only to the layout of widget controls in GUI):
 - ☐ **Send To Back** – Sends widget control to the back of the view.
 - ☐ **Bring To Front** – Brings widget control to the front of the view.
 - ☐ **Rotate Clockwise 90** – Rotates widget control 90 degrees clockwise. (Only for Linear Sliders).
 - ☐ **Rotate Counter Clockwise 90** – Rotates widget control 90 degrees counter clockwise. (Only for Linear Sliders).
 - ☐ **Flip Sensors** – Reverses the order of the sensors. (Only for Linear and Radial Sliders).
 - ☐ **Flip Columns Sensors** – Reverses the order of the Columns sensors. (Only for Touchpads and Matrix Buttons).
 - ☐ **Flip Row Sensors** – Reverses the order of the Row sensors. (Only for Touchpads and Matrix Buttons).
 - ☐ **Exchange Columns and Rows** – Columns sensors become rows and rows sensors become columns. (Only for Touchpads and Matrix Buttons).

Sensor Properties Tool Window

Sensor Properties tool window displays properties of the sensor selected on **Tuning** tab and its signal values.

- **Active sensor** – drop-down list located at the top side of the tool window and displays the name of the selected sensor. Active sensor can be selected at any time not depending on currently selected tab.
- **Bar graph** – Displays signal values for the selected sensor:
 - The maximum scale of the detailed view bar graph can be adjusted by double-clicking on Max Value label. Valid range for 8 bit Widget Resolution is between 1 and 255, default is **255**. Valid range for 16 bit Widget Resolution is between 1 and 32767, default is **32767**.
 - The current finger turn on threshold is displayed as a **green line** across the bar graph.
 - The current finger turn off threshold is displayed as a **red line** across the bar graph.
 - The current noise threshold is displayed as a **yellow line** across the bar graph.
 - Thresholds and hysteresis can be set by moving lines up and down with a mouse.
- **SNR** – The signal-to-noise ratio is computed in real time for the selected sensor. SNR values below 5 are poor and colored red, 5 to 10 are marginal and yellow, and greater than 10 is good and colored green. SNR value is calculated based on previously received data.
- **Sensor properties** (property grid located below bar graph) – Displays the properties for the selected sensor based on the widget type. It is located on the right side panel.
- **General CapSense properties** (property grid located below sensor properties grid, it is read only) – Displays global properties for the CapSense CSD component that cannot be changed at run time. These are for reference only. This information is located on the bottom of the right-side panel.

Graphing Tab

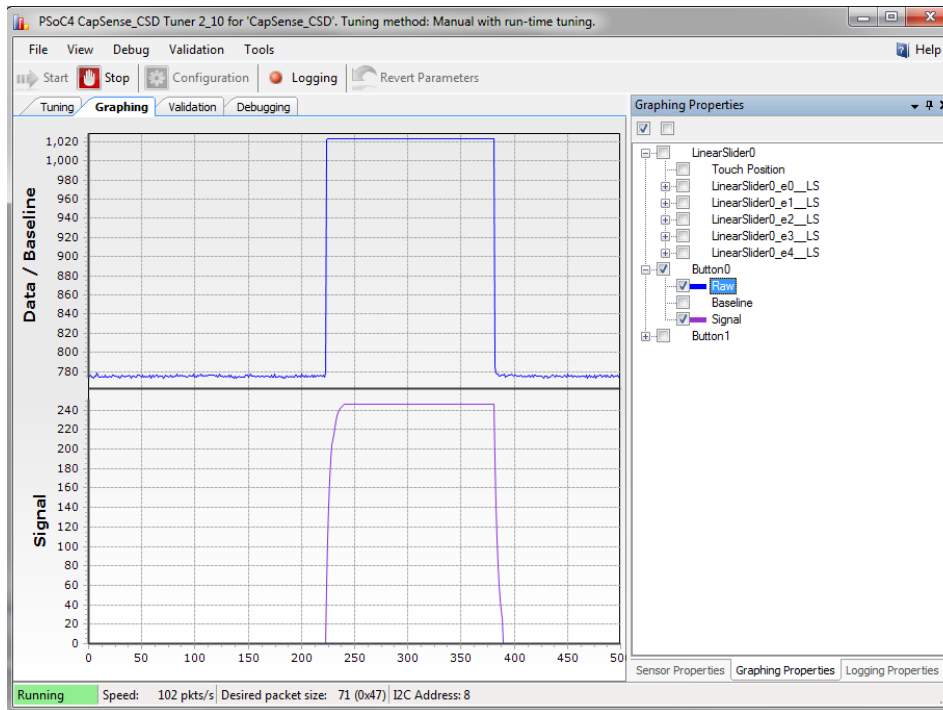


Chart area

Displays charts for selected items from the tree view. If you right-click the menu item **Export to .jpg**, you can generate a screenshot of the chart area that is saved as a .jpg file.

Graphing Properties Tool Window

Graphing Properties tool window allows selecting sensors and type of series which should be displayed on chart.

- **Tree view** – Gives all combinations of data for widgets and sensors which can be shown on the chart.

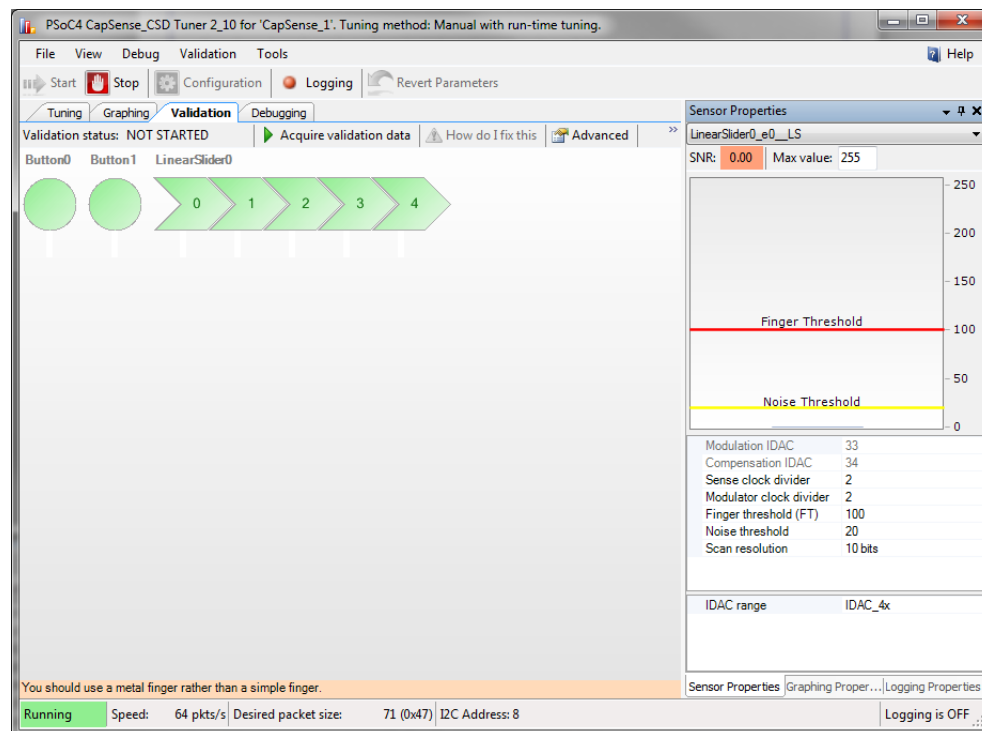
Logging Properties Tool Window

Logging Properties tool window allows selecting sensors and type of series which should be logged into a file.

- **Tree view** – Gives all combinations of data for widgets and sensors which can be logged to a file if the logging feature is enabled. The On/Off Status data value can only be logged, it cannot be shown on a chart
- **Append new data to existing file** – If selected, new data is appended to an existing file. If not selected, old data is erased from the file and replaced with the new data
- **Log duration** – Defines log duration in minutes. Default value is **10**
- **Log file name** – Defines log file path (file extension is .csv)

Validation Tab

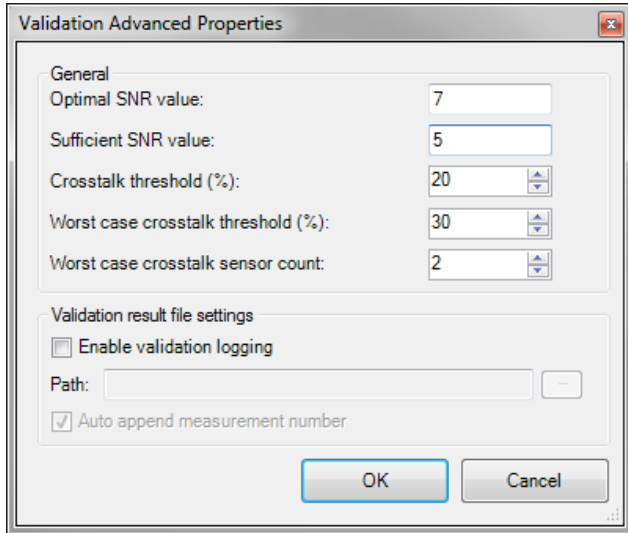
The **Validation** tab is for diagnostics only. The tab contains the widget layout view, but without the ability to edit the layout. This layout portion is used as a display only.



Top panel controls:

- **Validation Status** label – Shows validation status. It has following messages:
 - ❑ **VALIDATION NOT STARTED** – The validation process has not been run since the last time the design was changed.
 - ❑ **PASS** – The full validation process has been completed without failures.
 - ❑ **FAIL** – The validation process has uncovered failures; a validation report will be displayed.
- **Acquire validation data** (or menu item **Validation > Acquire Validation Data**) – Starts the validation process. This process guides you through a sequence of operations in which you are prompted to apply your finger to each sensor in sequence.
- **How do I fix this** – Opens a report with a list of suggested fixes for sensors that have not pass validation. This button is available only if the validation process was previously completed and design errors were found.
- **Advanced** (or menu item **Validation > Validation Advanced properties**) – Opens the properties window for validation properties (for more information, see [Validation Advanced Properties](#)).
- **SNRs** – In the widget schematic, turns the SNR display on or off (for more information, see [Validation Displays](#)).
- **Cross-talks** – In the widget schematic turns the cross-talk display on or off (for more information, see [Validation Displays](#)).

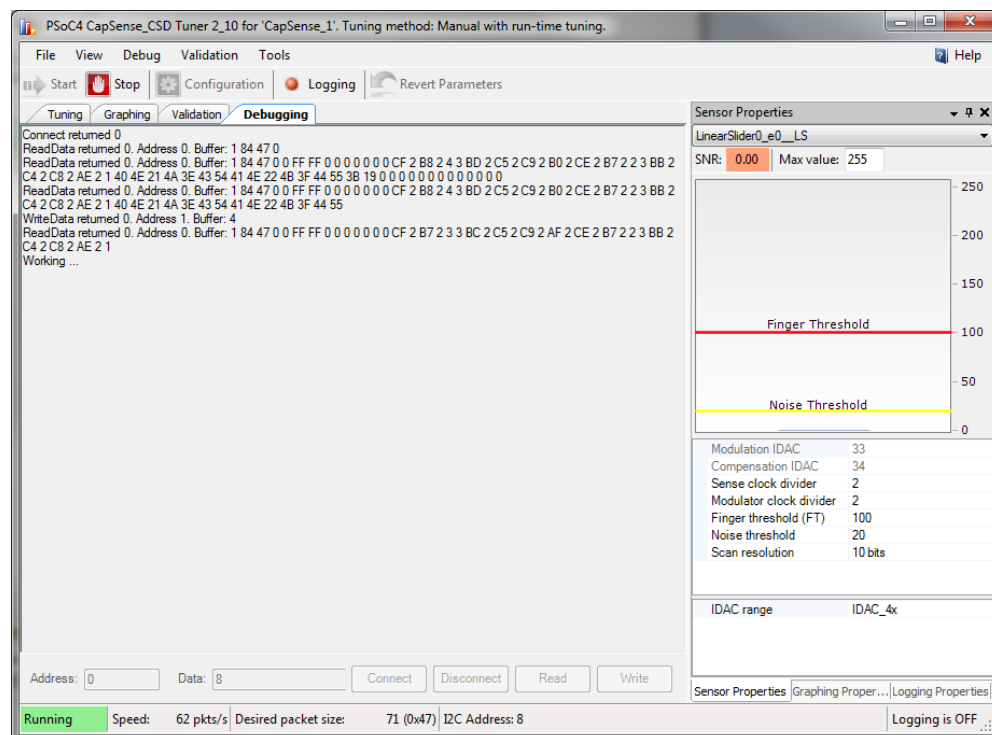
Validation Advanced Properties



The dialog box titled "Validation Advanced Properties" contains two sections. The "General" section has five input fields: "Optimal SNR value" (text box with 7), "Sufficient SNR value" (text box with 5), "Crosstalk threshold (%)" (spin box with 20), "Worst case crosstalk threshold (%)" (spin box with 30), and "Worst case crosstalk sensor count" (spin box with 2). The "Validation result file settings" section has a checkbox for "Enable validation logging" (unchecked), a "Path:" text box with a browse button, and a checked checkbox for "Auto append measurement number". At the bottom are "OK" and "Cancel" buttons.

- **Optimal SNR value** – Defines optimal SNR value. Valid range is between 0 and 100; default is **7**.
- **Sufficient SNR value** – Defines sufficient SNR value. Valid range is between 0 and 100; default is **5**.
- **Crosstalk threshold (%)**– Defines crosstalk threshold value as a percentage of the finger threshold for each sensor. Valid range is between 0 and 100 percent; default is **20**.
- **Worst case crosstalk threshold (%)** – Defines worst case crosstalk threshold value as a percentage of worst case crosstalk. Valid range is between 0 and 100 percent; default is **30**.
- **Worst case crosstalk sensor count** – Defines the number of sensors used to compute worst case crosstalk; valid range is between 0 and 100; default is **2**.
- **Enable validation logging** – Enables logging of validation data.
- **Path** – Defines log file path for validation data (file name extension is .csv).
- **Auto append measurement number** – If selected, after each start of the validation process, the log file name will be incremented (for example "validation001.csv") and data will be saved in a new file.

Debugging Tab



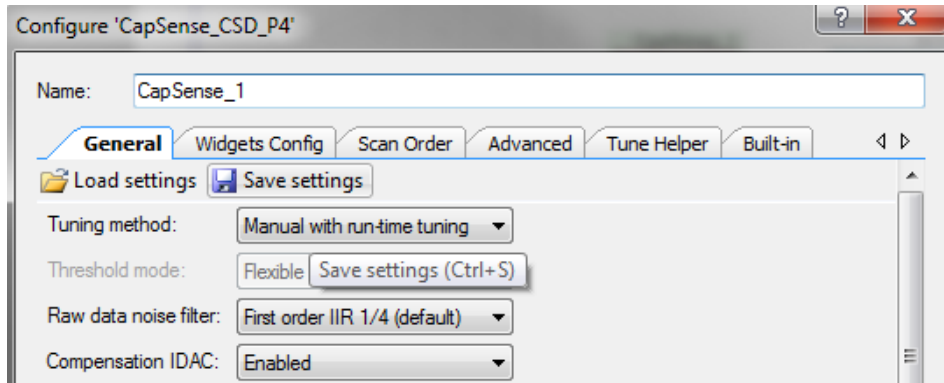
This functionality exists only for debugging purposes. It helps you investigate Tuner communication errors.

- **Debugging log window** – Displays communication commands that the Tuner executes. All communication errors are logged here. If the Tuner was successfully started, only the first few communication commands are logged.
- **Connect** – Connects to the PSoC device;
- **Disconnect** – Disconnects from the PSoC device;
- **Address** – Specifies the PSoC device address;
- **Read** – Reads data from the PSoC device. The address field defines the address in the buffer. The data field defines number of bytes to read;
- **Write** – Writes data to the PSoC device. The address field defines the address in the buffer. The data field defines the data to write.

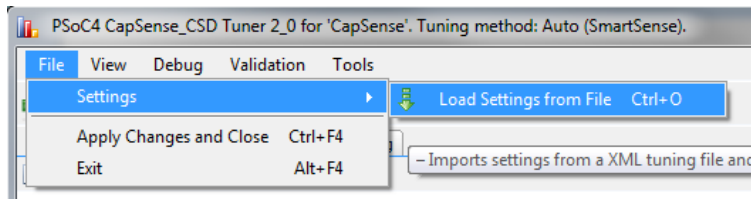
Save/Load Settings Feature

The Tuner GUI can also be opened as standalone application. In this case you must use the **Save settings** and **Load settings** features of the CapSense component.

1. Click the **Save settings** button on the Configure dialog.



2. In the **Save File** dialog box, specify name of the file and location where it will be saved.
3. Open the Tuner GUI and click **File > Settings > Load Settings from File**.



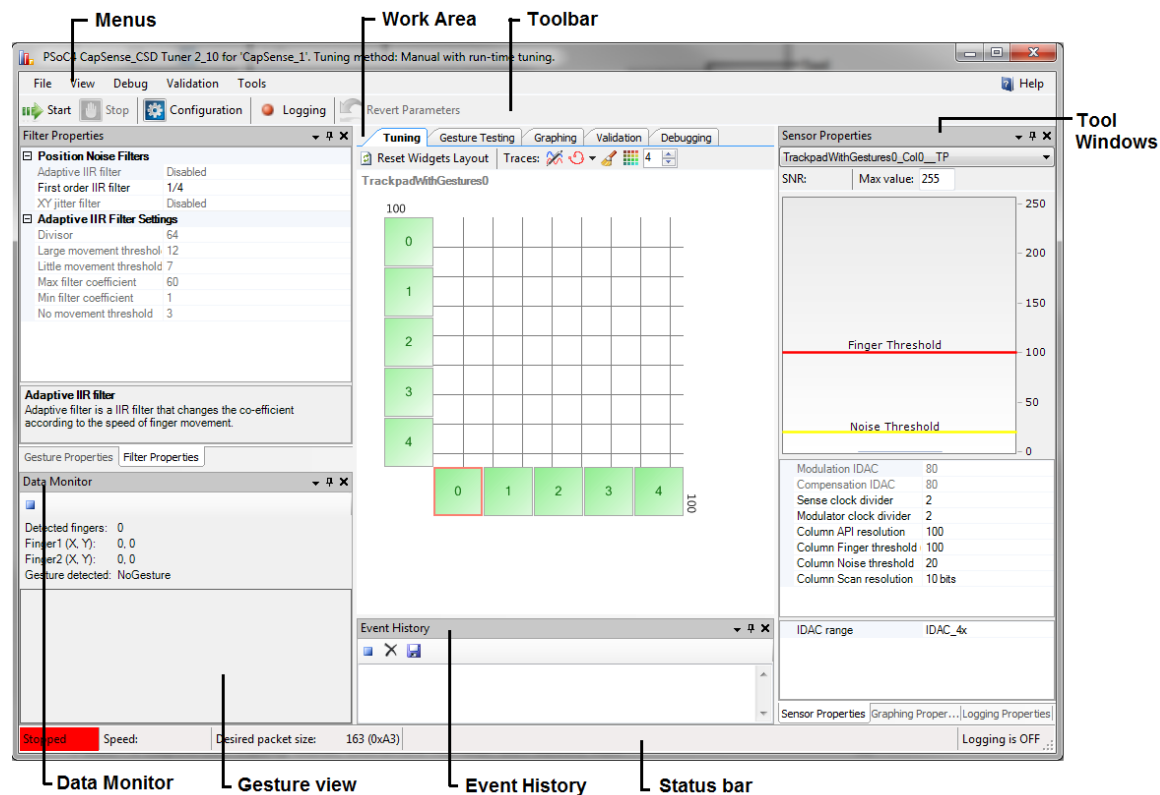
4. In the **File Open** dialog box, point to the previously saved file with the component settings. Settings will automatically load into the Tuner.

5 CapSense Gesture User Interface



This section describes the CapSense Tuner user interface for the case where a Trackpad with gestures widget is selected for the component. It is similar to the CapSense CSD user interface where there is no Trackpad with gestures widget, but there are significant differences.

General Interface



Data Monitor

Data Monitor shows information listed below:

- **Detected Fingers** – Number of finger on the panel
- **Finger 1(X,Y)** – First finger, X and Y positions on the panel
- **Finger 2(X,Y)** – Second finger, X and Y position on the panel
- **Gesture detected** – Gesture name

Data Monitor Toolbar

- **Start/Stop:** starts/ stops data monitor.

Gesture view

Gesture view shows a picture of the gesture on a successful gesture trigger event. The picture shows up for 3 secs and is replaced with user control color screen. In case a new gesture is triggered within 3 sec period, a picture for the new gesture is showed up. This is a read-only window.

Event History

Records and shows the history of gestures triggered. The user has an option of saving this history and exporting the data in .csv or .txt format. It fires when a finger is present on the trackpad (irrespective of gesture being triggered or not) and also in case of a lift-off. The event history displays data for all widgets. The following data is recorded by event history tab:

- Time stamp (updates on each scan)
- Number of fingers
- Gesture
- Finger1_X
- Finger1_Y
- Finger2_X
- Finger2_Y

Event History Toolbar:

- **Start/Stop:** starts/ stops the event logger
- **Clear :** Clears data in event logger
- **Save:** Allows to save the data in either.csv or .txt format

Sensor Properties

The following parameters can be configured at runtime.

Sensor Level:

- Compensation IDAC
- Modulation IDAC

Row/Column Level:

- Row Sense Clock Divider
- Column Sense Clock Divider
- Row Modulator Clock Divider
- Column Modulator Clock Divider
- Row Finger threshold
- Row Noise threshold

- Column Finger threshold
- Column Noise threshold
- Row Scan Resolution
- Column Scan Resolution
- Column API resolution
- Row API resolution

Component Level:

- IDAC RANGE

Gesture Properties

This tab displays the group of gestures listed in the table in the Gesture Parameter Descriptions section of the CapSense Gesture component datasheet. Clicking on a gesture group shows/hides the tuning parameters associated with that group of gestures.

General Parameters:

- Two finger settling count

Click Parameters:

- Single click min timeout
- Single click max timeout
- Double click min timeout
- Double click max timeout
- Two finger click min timeout
- Two finger click max timeout
- Double click max radius
- Click X Radius Pixels
- Click Y Radius Pixels

Edge Swipe Parameters:

- Active Edge Swipe threshold
- Top angle threshold
- Bottom angle threshold
- Width of disambiguation region
- Edge Swipe Timeout
- Edge Swipe Complete Timeout

Flick Parameters:

- Flick Active distance threshold X
- Flick Active distance threshold Y

- Flick sample time

One finger Scroll Parameters:

- One finger Scroll Threshold 1 X
- One finger Scroll Threshold 2 X
- One finger Scroll Threshold 3 X
- One finger Scroll Threshold 4 X
- One finger Scroll Step 1
- One finger Scroll Step 2
- One finger Scroll Step 3
- One finger Scroll Step 4
- One finger Scroll Debounce count
- One finger Inertial scroll Active distance threshold X
- One finger Inertial scroll Active distance threshold Y
- One finger Inertial scroll count level

Two finger Scroll Parameters:

- Two finger Scroll Threshold 1 X
- Two finger Scroll Threshold 2 X
- Two finger Scroll Threshold 3 X
- Two finger Scroll Threshold 4 X
- Two finger Scroll Step 1
- Two finger Scroll Step 2
- Two finger Scroll Step 3
- Two finger Scroll Step 4
- Two finger Scroll Debounce count
- Two finger Inertial scroll Active distance threshold X
- Two finger Inertial scroll Active distance threshold Y
- Two finger Inertial scroll count level

Zoom in/out Parameters:

- Zoom Active distance threshold X
- Zoom Active distance threshold Y
- Debounce Zoom count
- Debounce Two finger Scroll to Zoom count

Rotate Parameters:

- Rotate debounce limit

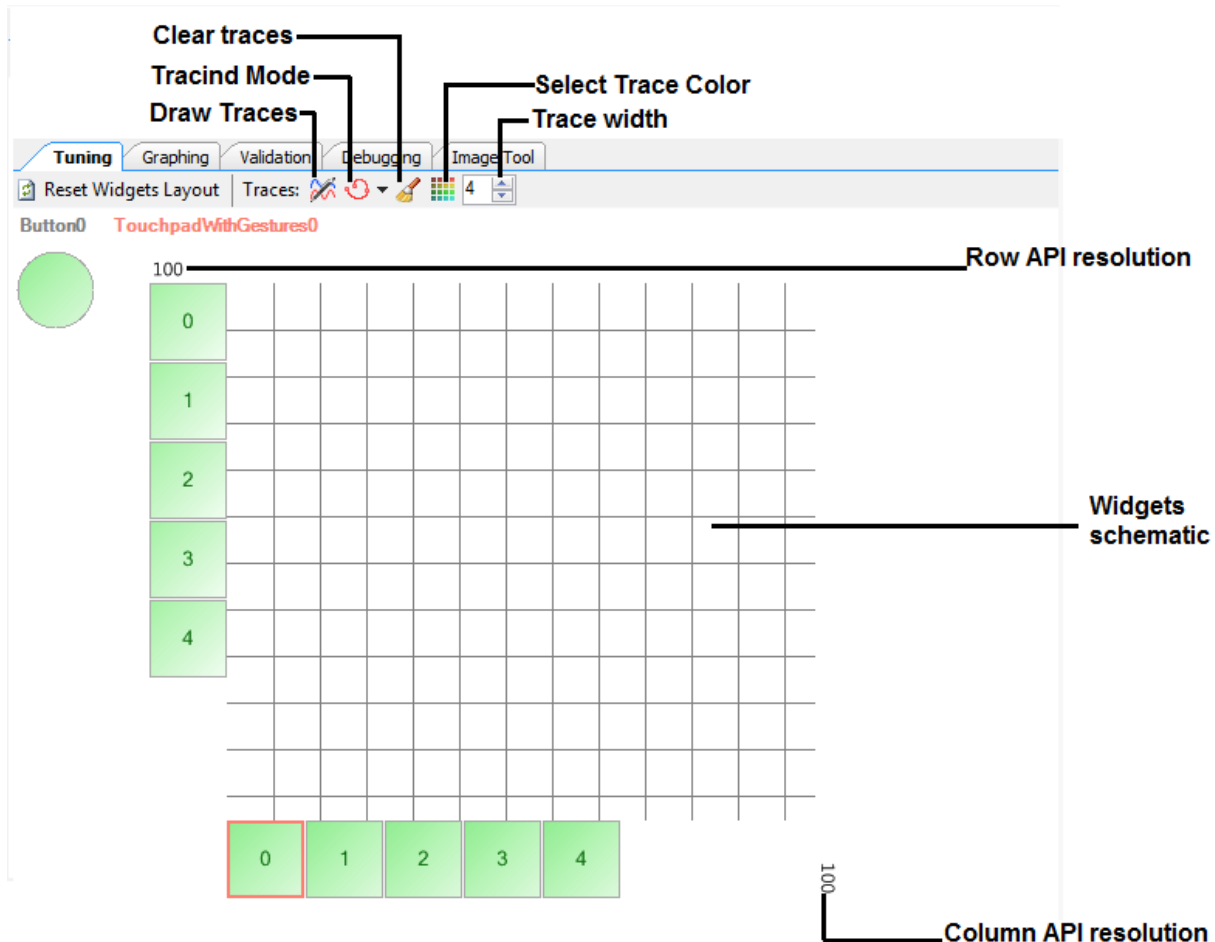
Filter Properties

Following options are provided to enable and disable X/Y IIR filters to observe improvement in linearity:

- XY jitter filter. Can be enabled and disabled
- XY IIR filter. XY IIR filter coefficient can be changed to 1/2, 1/4, 1/8 and 1/16
- Adaptive filter. The following parameters can be changed:
 - ☐ Max Filter Coefficient
 - ☐ Min Filter Coefficient
 - ☐ No Movement Threshold
 - ☐ Little Movement Threshold
 - ☐ Large Movement Threshold
 - ☐ Filter Divisor

Tuning Tab

Displays all of the component widgets as configured on a workspace. This allows you to arrange the widgets similarly to the way they appear on the physical PCB or enclosure. This tab is used for tuning widget parameters and visualizing widgets data and states.



Widgets schematic:

The schematic contains a graphical representation of all of the configured widgets. If a widget is composed of more than one sensor, the individual sensors may be selected for detailed analysis. Every widget is movable within the schematic.

Toolbar:

- **Reset Widgets Layout** – Moves widgets to default positions within the schematic.
- **Traces buttons:**
 - **Draw Traces** – enables/disables Traces Mode. In this mode the finger tracking for one finger is shown;
 - **Tracind Mode** – selects Continuous or Discontinuous mode. In Continuous mode only, the **Clear traces** button erases the lines drawn on the trackpad by a finger. In Discontinuous mode, a next touch erases the LINES DRAWN on the trackpad by a finger.

- ❑ **Clear traces** – erases lines drawn on the trackpad by a finger;
- ❑ **Select Trace Color** – gives possibility to select a trace color. In the Continuous mode you can set a color for the different traces;
- ❑ **Trace width** – sets traces width.

Widget controls context menu:

(this functionality applies only to the layout of widget controls in GUI when - right click mouse button):

- **Send To Back** – Sends widget control to the back of the view.
- **Bring To Front** – Brings widget control to the front of the view.
- **Flip Columns Sensors** – Reverses the order of the Columns sensors. (Only for Touchpads and Matrix Buttons).
- **Flip Row Sensors** – Reverses the order of the Row sensors. (Only for Touchpads and Matrix Buttons).
- **Exchange Columns and Rows** – Columns sensors become rows and rows sensors become columns. (Only for Touchpads and Matrix Buttons).

Graphing Tab

The **Graphing** tab for the Trackpad with gestures widget is the same as described in the Graphing Tab section.

Validation Tab

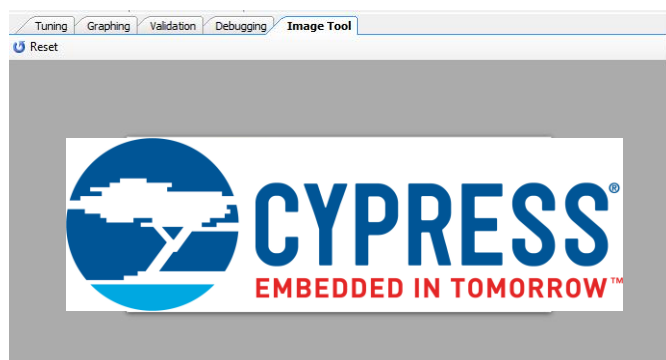
The **Validation** tab for the Trackpad with gestures widget is the same as described in the Validation Tab section.

Debugging Tab

The **Debugging** tab for the Trackpad with gestures widget is the same as described in the Debugging Tab section.

Image Tool Tab

This tab displays a sample image to test an experience in a normal use case. Each gesture has a specific action that is triggered on the picture.



The **Reset** button resets image size and position.

The following table provides a list of actions that each gesture will have on the image.

Gesture	Action
Zoom In	When this gesture is detected, the image zooms in around the mouse pointer.
Zoom Out	When this gesture is detected, the image zooms out around the mouse pointer.
Two finger Vertical Scroll	When this gesture is detected, the image scrolls in the respective direction.
Two finger Horizontal Scroll	
Two finger Inertial Vertical Scroll	When this gesture is detected, the Image scrolls inertially in the respective direction.
Two finger Inertial Horizontal Scroll	
One finger Horizontal Scroll	When this gesture is detected, the image moves in the respective direction.
One finger Vertical Scroll	
One finger Inertial Horizontal Scroll	When this gesture is detected, the Image scrolls inertially in the respective direction.
One finger Inertial Vertical Scroll	
Left Edge Swipe	No action on tool
Right Edge Swipe	
Top Edge Swipe	
Bottom Edge Swipe	