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.

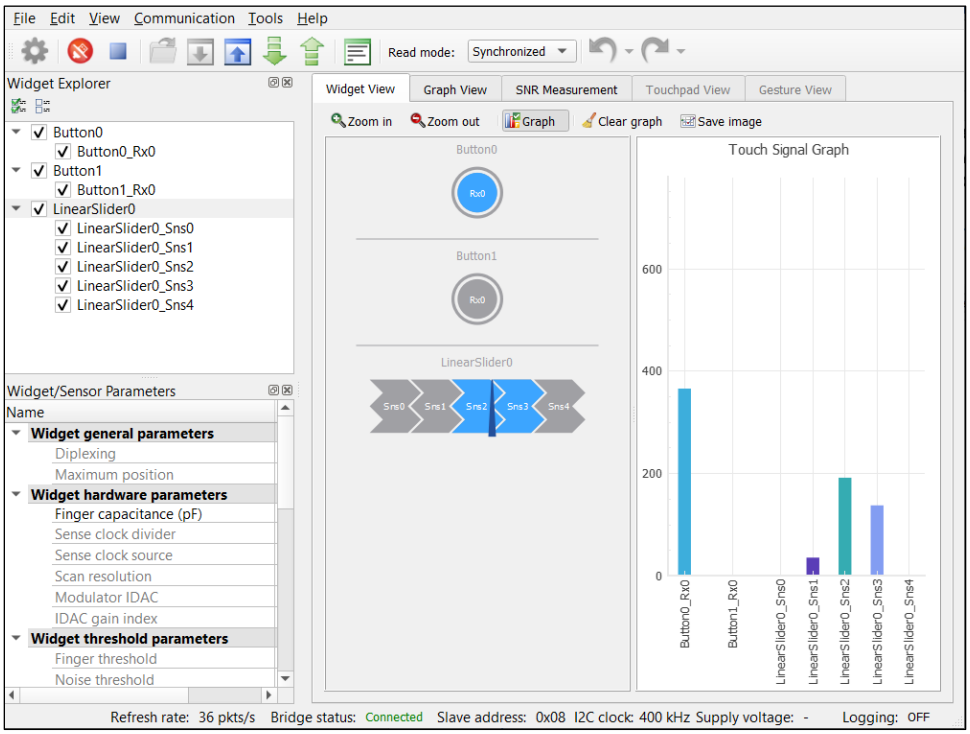
ModusToolbox® CapSense® Tuner Guide

Version 3.11

Overview

The CapSense Tuner is a stand-alone tool included with the ModusToolbox software. The tool is used to tune CapSense applications.

Prior to using the CapSense Tuner, create a CapSense application and program it into the device. Refer to the CapSense Configurator User Guide and the CapSense Middleware Documentation for help. Your application must contain the CapSense Middleware and a communication interface. The Tuner works with I2C and UART communication interfaces.



Launch the CapSense Tuner

You can launch the CapSense Tuner various ways: as a stand-alone tool, from the Device Configurator, from the Eclipse IDE for ModusToolbox or from the command line. Then, you can either use the generated source with an Eclipse IDE application, or use it in any software environment you choose.

As a Stand-Alone Tool

You can launch the CapSense Tuner as a stand-alone tool. By default, it is installed here:

<install_dir>/ModusToolbox/tools_<version>/capsense-configurator<version>

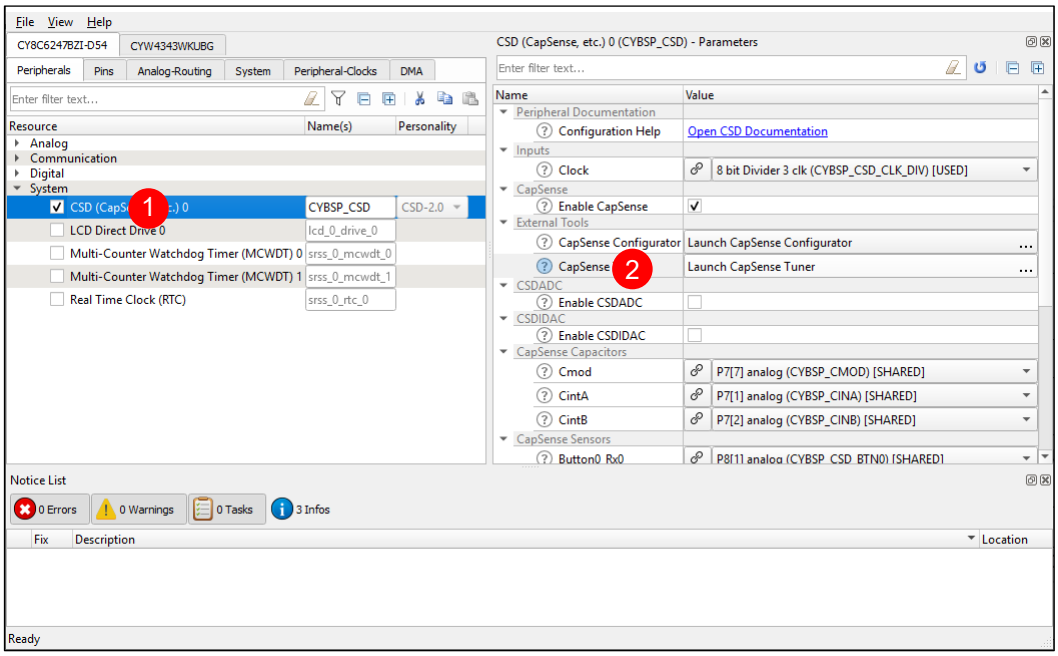
On Windows, launch the tool from the **Start** menu. For other operating systems, navigate to the install location and run the executable.

When run independently, the application opens without any information. You need to open a specific CapSense application configuration file that was created by the CapSense Configurator for the Tuner to work. See Menus for more information about opening a configuration file.

From the Device Configurator

You can also launch the CapSense Tuner from the Device Configurator. Refer to the Device Configurator Guide for more details.

1. On the **Peripherals** tab, select the **CSD (CapSense)** resource.
2. On the **Parameters** pane, click the **Launch CapSense Tuner** button.



From the Eclipse IDE

Note These steps are the minimum required to launch the CapSense Tuner. However, the Tuner will not work without a properly configured CapSense application programmed into the device.

If your Eclipse IDE application already includes a *design.cycapsense* file in the Board Support Package (BSP), right-click on a project in the IDE Project Explorer, and select **ModusToolbox > CapSense Tuner** to open the tool.

If your Eclipse IDE application does **not** include a *design.cycapsense* file, then you do not have a properly configured CapSense application. Refer to the CapSense Configurator Guide to create such a design.

From the Command Line

You can run the Tuner from the command line. However, there are only a few reasons to do this in practice.

For information about the command-line options, run the Tuner using the `-h` option.

Quick Start

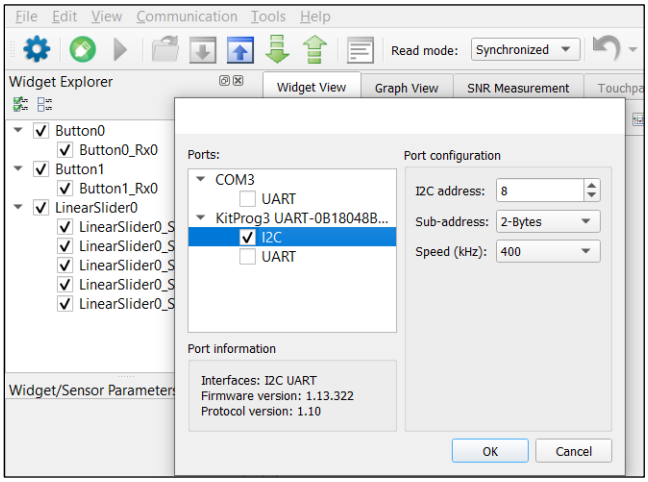
This section provides a simple workflow for how to use the CapSense Tuner.

1. Create a CapSense application with a tuner communication interface, and program the application into the device. Refer to the *Eclipse IDE for ModusToolbox User Guide* and the *CapSense Configurator Guide* for more details.

   Note You can also use the CapSense Slider starter application configured to work on PSoC 6 MCU kits.

2. Launch the CapSense Tuner.

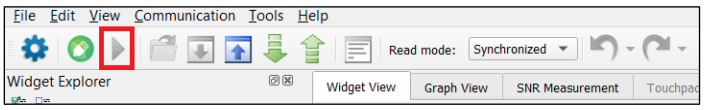
3. Start Communication. Click **Connect** and select the protocol. In this case “I2C.”



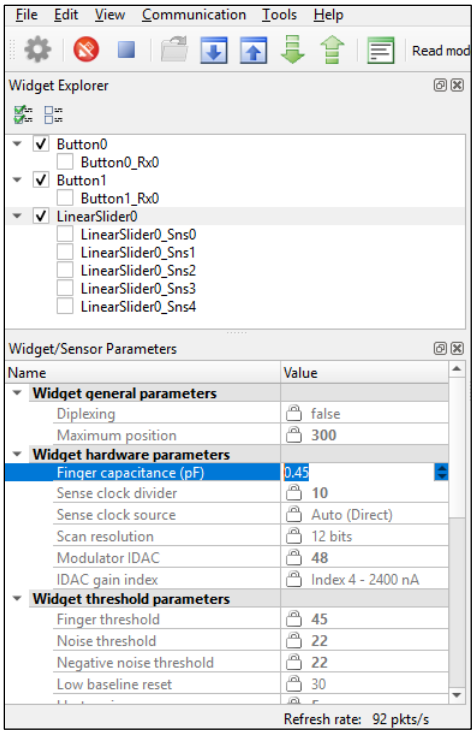
Notes

- Refer to the KitProg User Guide for supported configurations and modes. For this example, I2C address, sub-address, and speed must match the configuration.
- For the I2C communication, the Tuner only supports the 2-Bytes Sub-address.

4. Click **Start** to extract data.



5. Touch the sensors on the hardware and notice the change in the sensor/widget status on Widget View Tab.
6. Open the **Graph View** tab. Check the sensors in the Widget Explorer Pane to observe sensor signals in the graph. Touch the sensors and notice the signal change on the Graph View Tab.
7. Change widget/sensor parameter values as needed. Then, apply the new settings to the device using the **Apply to Device** button.



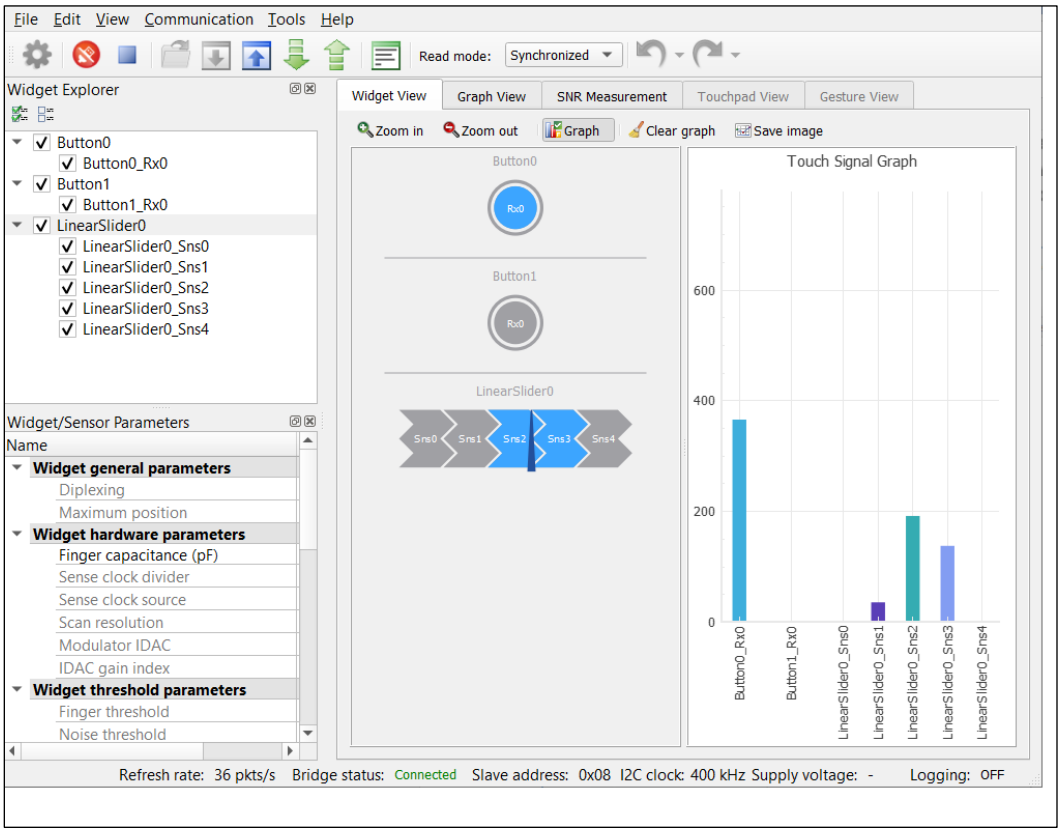
You can do this when using Manual or SmartSense (Hardware parameters only) modes for tuning:

- To edit the threshold parameters, use SmartSense (Hardware parameters only) mode.
- To edit all the parameters, use Manual mode.
- When SmartSense (Full Auto-Tune) is selected for CSD tuning mode, parameters are read-only (except the Finger Capacitance parameter).

8. Save the Tuner parameters. Click the **Apply to Project** button.
9. Exit the Tuner application.

GUI Description

The CapSense Tuner application contains menus, a toolbar, panes, tabs, and a status bar, all used to tune a CapSense application.



Menus

File

- **Open** – Opens a *<file_name>.cycapsense* configuration file. This command is visible only when running the Tuner independently from the Eclipse IDE.
- **Apply to Device** – Commits current values of a widget/sensor parameter to the device. This command becomes active if a value of any configuration parameter from the Tuner application changes (that is, if the parameter values in the Tuner and the device are different). This is an indication to apply the changed parameter values to the device.
- **Apply to Project** – Commits current values of a widget / sensor parameter to the CapSense project.
- **Import –** Imports a specified configuration file.
- **Export –** Exports the current configuration file into a specified file.
- **Exit** – Asks to save changes if there are any and closes the Tuner. Changes are saved to the configuration file.

Edit

- **Undo** – Undoes the last action or sequence of actions.
- **Redo** – Redoes the last undone action or sequence of undone actions.

View

- **Widget Explorer –** Hides or shows the Widget Explorer pane where widgets and sensors tree display.
- **Widget/Sensor Parameters –** Hides or shows the Widget/Sensor Parameters pane.
- **Toolbar –** Hides or shows the Toolbar.
- **Reset View –** Resets the view to the default.

Communication

- **Connect** – Connects to the device via a communication channel selected in the Tuner Communication Setup dialog. If the channel was not previously selected, the Tuner Communication dialog is shown.
- **Disconnect** – Closes the communication channel with the connected device.
- **Start** – Starts reading data from the device.
- **Stop** – Stops reading data from the device.

Tools

- **Tuner Communication Setup…** – Opens the configuration dialog to set up a communication channel with the device.
- **Options…** – Opens the configuration dialog to set up different tuner preferences.

Help

- **View Help** – Opens the CapSense Tuner User Guide (this document).
- **About CapSense Tuner** – Opens the About box to display the version information.

Toolbar

Contains frequently used buttons that duplicate the main menu items:

-  – Opens the configuration dialog to set up a communication channel with the device. The same as the **Tools > Tuner Communication Setup** menu command.
-  – Connects to the device via a communication channel selected in the Tuner Communication Setup dialog. The same as the **Communication > Connect** menu command.
-  – Closes the communication channel with the connected device. The same as the **Communication > Disconnect** menu command
-  – Starts reading data from the device. The same as the **Communication > Start** menu command.
-  – Stops reading data from the device. The same as the **Communication > Stop** menu command.
-  – Opens a configuration file. The same as the **File > Open** menu command.

-  – Commits current values of a widget/sensor parameter to the device. The same as the **File > Apply to Device** menu command.
-  – Commits current values of a widget / sensor parameter to the CapSense project. The same as the **File > Apply to Project** menu command.
-  – Imports a specified configuration file. The same as the **Import** menu command.
-  – Exports the current configuration file into a specified file. The same as the **Export** menu command.
-  – Starts or stops data logging into a specified file.
- **Read mode** – Selects the Tuner communication mode with a device. The following options are available (under the I2C communication interface):
  - **Asynchronized** – When selected, the Tuner reads data asynchronously with sensor scanning and data processing. Due to this, data coherency may be corrupted. So, the Tuner may read only partially updated sensor data. For example, the device completed scanning of only the first sensor in a row. At this moment, the Tuner reads data of the latest scan from the first sensor and data of previous scans from the remaining sensor. This can occur to all sensors, when the Tuner reads data prior to the completion of data processing tasks, such as baseline update and filter execution. SNR and noise measurements are less accurate due to non-coherent data reading and only a limited set of tuning parameters can be edited in real time in this mode.
  - **Synchronized** – When selected, application firmware periodically calls a corresponding Tuner function: Cy_CapSense_RunTuner(). The Tuner synchronizes data reading and firmware execution to preserve data coherency. CapSense middleware waits for the Tuner read/write operation to be completed prior to execution of sensor scan and processing tasks. The Tuner does not read/write data until CapSense middleware completes scanning and data processing tasks for all widgets in the application. The SNR and noise measurements are most accurate, and most tuning parameters can be edited in real time updating in this mode.
-  – Undoes the last action or sequence of actions. The same as the **Edit > Undo** menu command.
-  – Redoes the undone action or sequence of undone actions. The same as the **Edit > Redo** menu command.

Widget Explorer Pane

The Widget Explorer pane contains a tree of widgets and sensors used in the CapSense application. You can expand/collapse the Widget nodes to show/hide widget's sensor nodes. You can check/uncheck individual widgets and sensors in the Widget Explorer pane. The widget checked status affects its visibility in the *Widget View*, while the sensor checked status controls the visibility of the sensor raw count / baseline / signal / status graph series in the Graph View and signals in the *Touch Signal Graph* on the *Widget View*.

Selecting a widget or sensor in the Widget Explorer pane updates the selection in the Widget/Sensor Parameters pane.

Note For CSX widgets, the sensor tree displays individual nodes (Rx0_Tx0, Rx0_Tx1 …) contrary to the Configurator where the CSX electrodes are displayed (Rx0, Rx1 … Tx0, Tx1 ...).

Widget/Sensor Parameters Pane

The Widget/Sensor parameters pane displays the parameters of a widget or sensor selected in the Widget Explorer tree. The grid is similar to the grid on the Widget Details tab in the CapSense Configurator. The main difference is that some parameters are available for modification in the Configurator, but not in the Tuner. This pane includes the following parameters:

- **Widget General Parameters** – Cannot be modified from the Tuner because corresponding parameter values reside in the application flash widget structures that cannot be modified at runtime.
- **Widget Hardware Parameters** – Cannot be modified for the CSD widgets when CSD tuning mode is set to SmartSense (Full Auto-Tune) or SmartSense (Hardware parameters only) in the CapSense Configurator. In Manual tuning mode (for both CSD and CSX widgets), any change to Widget Hardware Parameters requires hardware re-initialization performed only if the Tuner communicates with the device in Synchronized mode.
- **Widget Threshold Parameters** – Cannot be modified for the CSD widgets when CSD tuning mode is set to SmartSense (Full Auto-Tune) in the Configurator. In Manual tuning mode (for both CSD and CSX widgets), threshold parameters are always writable (Synchronized mode is not required). The exception is the ON debounce parameter that also requires hardware re-initialization (in the same way as the hardware parameters).
- **Sensor Parameters** – Sensor-specific parameters. The Tuner application displays only IDAC values or/and compensation IDAC value. The parameter is not present for the CSD widget when Enable Compensation IDAC is disabled on the Configurator CSD Settings tab. When CSD Enable IDAC auto-calibration or/and CSX Enable IDAC auto-calibration is enabled, the parameter is Read-only and displays the IDAC value as calibrated by the firmware. When auto-calibration is disabled, the IDAC value entered in the CapSense Configurator appears, and the parameter is writable in Synchronized mode.
- **Filter Parameters** and **Centroid Parameters** – Cannot be modified at runtime from the Tuner because, unlike the other parameters, these parameter values reside in the application flash widget structures that cannot be modified at runtime.
- **Gesture Parameters** – Synchronized communication mode must be selected to update the Gesture parameters during runtime from the Tuner application.

  Note The position filter parameters, Adaptive IIR filter parameters, and Gesture parameters reflect data stored in a loaded configuration file. Actual values may vary on a target device connected to the CapSense Tuner.

Tabs

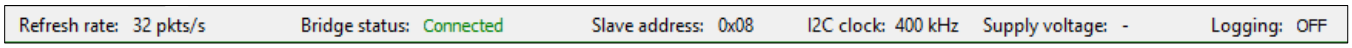
The application consists of the following tabs:

- Widget View – Displays the widgets, their touch status, and the touch signal bar graph.
- Graph View – Displays the sensor data charts.
- SNR Measurement – Provides the SNR measurement functionality.
- Touchpad View – Displays the touchpad heatmap.
- Gesture View – Displays the Gesture operation.

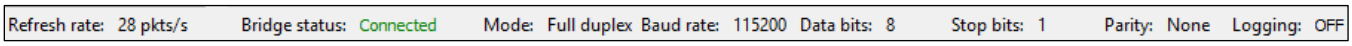
Status Bar

The status bar at the bottom of the GUI displays information related to the communication state between the Tuner and the device. Some sections differ depending on the communication type as follows:

I2C Clock



UART



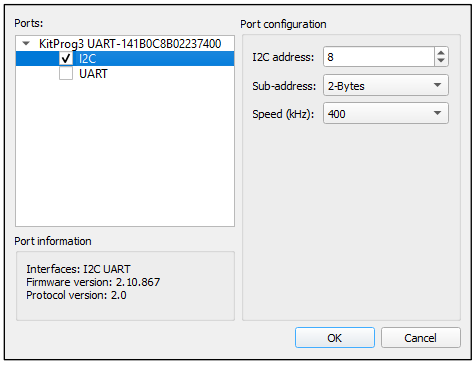
- **Refresh rate** – A count of read samples performed per second. The count depends on multiple factors: the selected communication channel, communication speed, and amount of time for a single scan.
- **Bridge status** – Either **Connected**, when the communication channel is active, or **Disconnected** otherwise.
- **I2C**
  - **Slave address** – The address of the I2C slave configured for the current communication channel.
  - **I2C clock** – The data rate used by the I2C communication channel.
  - **Supply voltage** – The supply voltage.
- **UART**
  - **Mode**

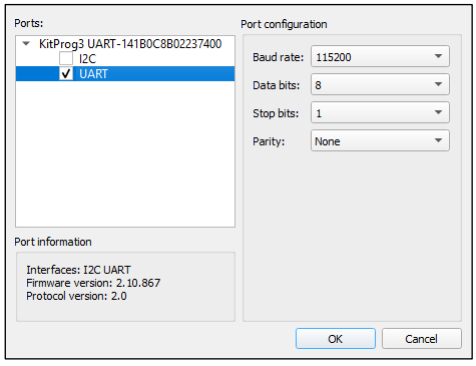
    Full duplex mode – Provides data transmission in two reverse directions: - target device → tuner→ target device. In this mode, you can change the parameters of the widgets / sensors at Run-time. To save the new configuration, click the **Apply to Device** button

    Simplex mode – Provides data transmission in one direction: - target device → tuner. In this mode, you cannot change the parameters of the widgets / sensors at Run-time.
  - **Baud rate** – The data rate at which CapSense operates with current settings.
  - **Data bits** – A count of data bits transmitted between the start and stop of a single UART transaction.
  - **Stop bits** – A count of stop bits implemented in the transmitter.
  - **Parity** – Functionality of the parity bit location in a transaction.
- **Logging** – Either **ON** (when the data logging to a file in progress) or **OFF**.

Tuner Communication Setup

The Tuner Communication Setup dialog is used to establish communication between the Tuner and target device. The Tuner supports I2C and UART communication interfaces.





Select **Tools > Tuner Communication Setup…** on the menu to open the dialog or click the **Tuner Communication Setup** button. The dialog opens automatically after clicking the **Connect** button if no device was previously selected.

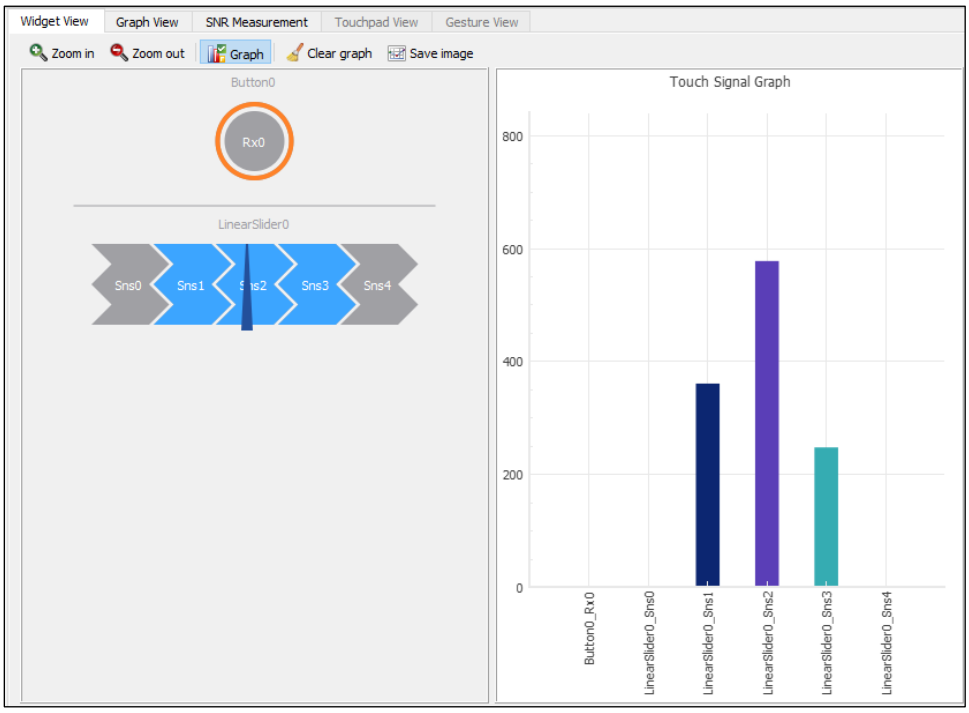
Select the device and communication protocol by checking the item with the protocol name. Change the protocol configuration parameters to match the configuration of the communication peripheral in the application.

Notes

- The parameters in the Tuner Communication Setup dialog must be identical to the parameters of the I2C or UART driver in the application.
- Not all UART communication properties combinations are supported. For more details, refer to the KitProg User Guide
- Recommended – enable the UART Flow control on the target device for bitrates equal or higher than 1000000.

Widget View Tab

The **Widget View** provides a visual representation of all widgets selected in the Widget Explorer Pane. If a widget consists of more than one sensor, individual selected sensors are highlighted in the Widget Explorer Pane and Widget/Sensor Parameters Pane.



The Widget and/or sensors are highlighted when the device reports their touch status as active.

Some additional features are available depending on the widget type.

Touch Signal Graph

The Widget view also displays a Touch Signal Graph. This graph contains a touch signal level for each sensor selected in the Widget Explorer Pane.

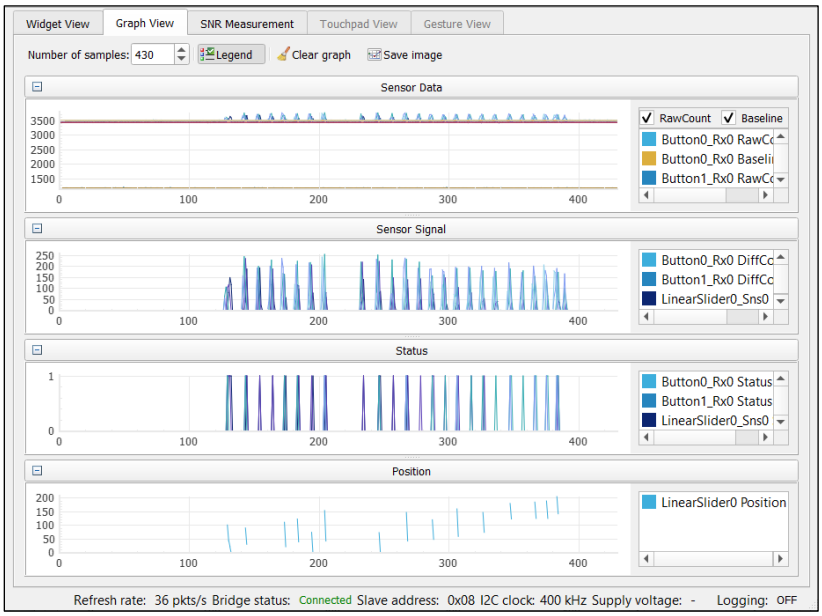
Toolbar

The toolbar provides quick access to different Tuner configuration options that affect the Widget View display.

- **Zoom in** – Zooms in the widgets.
- **Zoom out** – Zooms out the widgets.
- **Graph** – Shows or hides the Touch Signal Graph.
- **Clear graph** – Clears the Touch Signal Graph.
- **Save image** – Opens the dialog to save the Widget View tab as an image. Supported formats: .PNG, .JPG, .BMP.

Graph View Tab

The Graph View displays graphs for the sensors selected in the Widget Explorer Pane.



The following graphs are available:

- **Sensor Data** – Displays RawCount and Baseline. Click a corresponding check box to see them. Under Multi-frequency mode, RawCount and Baseline are available for three channels: 0, 1, 2.
- **Sensor Signal** – Displays signal differences.
- **Status** – Displays a sensor status (Touch/No Touch).
- **Position** – Displays touch positions for the Linear Slider, Radial Slider, and Touchpad widgets.

Toolbar

The toolbar provides quick access to different Tuner configuration options that affect the Tuner graphs display.

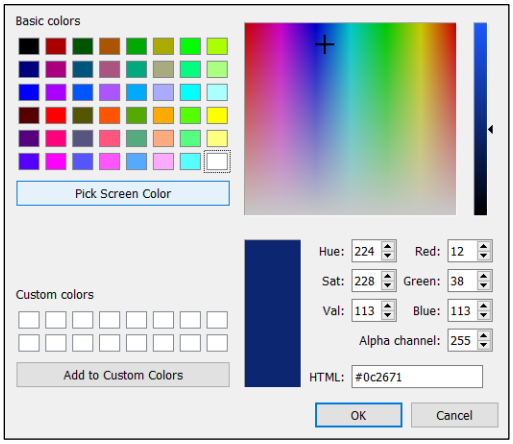
- **Number of samples** – Defines the total amount of data samples shown on a single graph.
- **Show / Hide legend** – Shows or hides the sensor series descriptions (with names and colors) in graphs.
- **Clear graph** – Clears all graphs.
- **Save image** – Opens a dialog to save the Graph View tab as an image. Supported formats: .PNG, .JPG, .BMP.

Graph Functionality

The Pan and Zoom features allow you to examine graphs in more detail. Use a mouse drag for Pan. Use the mouse wheel for Zoom. The [**Ctrl**] key + mouse wheel zooms all graphs simultaneously. Press the [**Esc**] key to undo zoom and pan.

If you click a sensor line series on the graph, the corresponding sensor series highlights on the legend pane. Likewise, if you click a sensor series on the legend pane, it highlights a sensor line series on the graph. To highlight lines for all graphs at the same time, press and hold the [**Ctrl**] key before clicking. To select multiple sensors in the other Views, press and hold the [**Alt**] key while selecting the sensors.

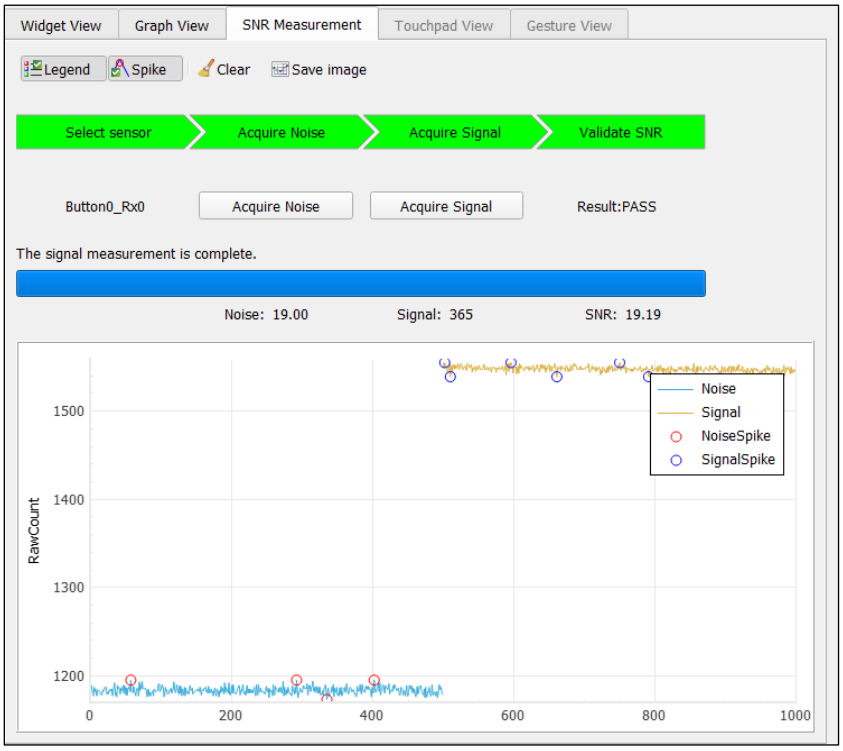
To change a sensor series color on the legend pane, double-click it. The “Please choose a color for …” dialog displays.



Select a color and click **OK**. To select a color for all the graphs at the same time, press and hold the [**Ctrl**] key while clicking.

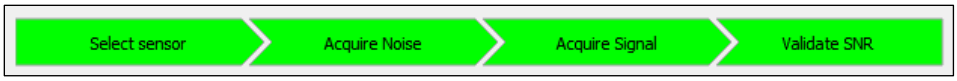
SNR Measurement Tab

The **SNR Measurement** tab allows measuring a SNR (Signal-to-Noise Ratio) for individual sensors. It provides the user interface to acquire noise and signal separately and then calculates a SNR based on the captured data. The obtained value is then validated by a comparison with the required minimum (5 by default, can be configured in the Options dialog).



The SNR Measurement tab contains several areas, as follows.

At the top of the **SNR measurement** tab, there is a bar with the status labels. Each label status is defined by its background color.

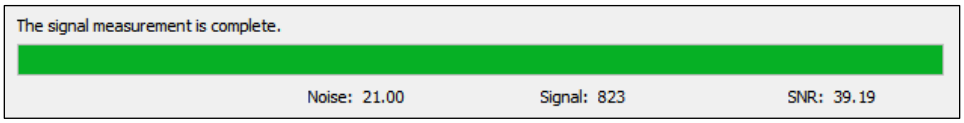


- **Select sensor** – Green when there is a sensor selected; gray otherwise.
- **Acquire noise** – Green when noise samples are already collected for the selected sensor; gray otherwise.
- **Acquire signal** – Green when signal samples are already collected for the selected sensor; gray otherwise.
- **Validate SNR** – Green when both noise and signal samples are collected, and the SNR is above the valid limit; red when the SNR is below the valid limit, and gray when either noise or signal are not yet collected.

Below the top status labels bar, there are the following controls.

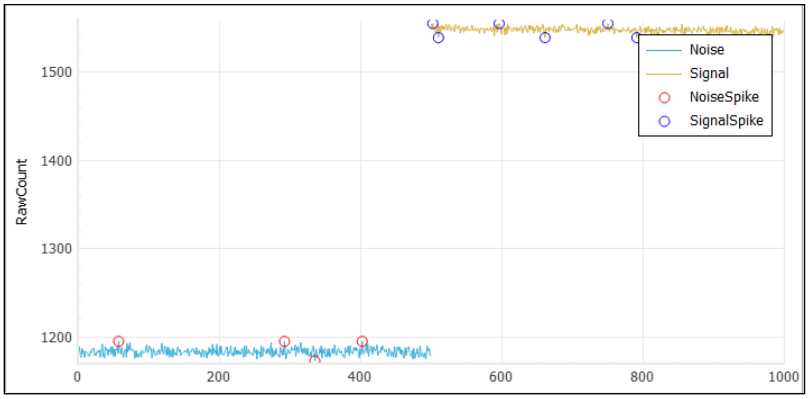
- **Sensor name** – The sensor selected in the Widget Explorer Pane or None (if no sensor selected).
- **Acquire Noise** – This button is disabled when no sensor is selected or communication is not started. When acquiring noise is in progress, clicking the button aborts the operation.
- **Acquire Signal** – This button is disabled when no sensor is selected, communication is not started, or noise samples are not yet collected for the selected sensor. When acquiring signal is in progress, clicking the button aborts the operation.
- **Result** – This label shows either N/A (when the SNR cannot be calculated due to noise/signal samples not collected yet), PASS (when the SNR is above the required limit), or FAIL (when the SNR is below the required limit).

Below the controls bar, there is the current status message and the progress of the current operation. Below the progress bar, there are three labels:



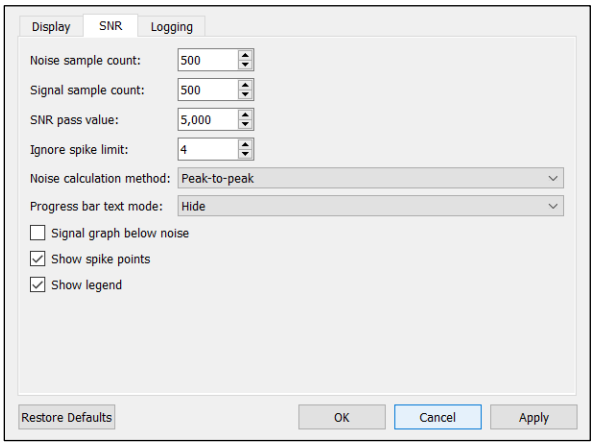
- **Noise** – Shows the last measured noise average value for the selected sensor, or N/A if no noise measurement is performed yet.
- **Signal** – Shows the last measured signal average value for the selected sensor, or N/A if no signal measurement is performed yet.
- **SNR** – Shows the calculated SNR value. This is the result of the Signal/Noise division rounded up to 2 decimal points. When an SNR cannot be calculated, N/A is displayed instead.

At the bottom, there is a chart that displays the measured sensor noise and signal. Noise and signal spikes displayed on the chart are points ignored during SNR calculation.



SNR Options

SNR parameters can be modified in the Options dialog. See Tuner Configuration Options for descriptions of other tabs on this dialog.



- **Noise sample count** – The count of samples to acquire during the noise measurement operation.
- **Signal sample count** – The count of samples to acquire during the signal measurement operation.
- **SNR pass value** – The minimal acceptable value of the SNR.
- **Ignore spike limit** – Ignores a specified number of the highest and the lowest spikes at noise / signal calculation. That is, if you specify number 3, then three upper and three lower raw counts are ignored separately for the noise calculation and for the signal calculation.
- **Noise calculation method** – Allows selecting the method to calculate the noise average. The following methods are available for selection:
  - **Peak-to-peak** (by default) – Calculates noise as a difference between the maximum and minimum value collected during the noise measurement.
  - **RMS** – Calculates noise as a root mean-square of all samples collected during the noise measurement.
- **Progress bar text mode** – This label is shown with the progress bar:

  - **Hide** (by default) – No label.
  - **Percent** – The number of samples in percent acquired during the signal measurement operation.
  - **Value** – The number of samples acquired during the signal measurement operation.
- **Signal graph below noise** – Displays the signal series below the noise threshold on the graph.
- **Show spike points** – Highlights the spike points on the graph.
- **Show legend** – Shows the graph legend.

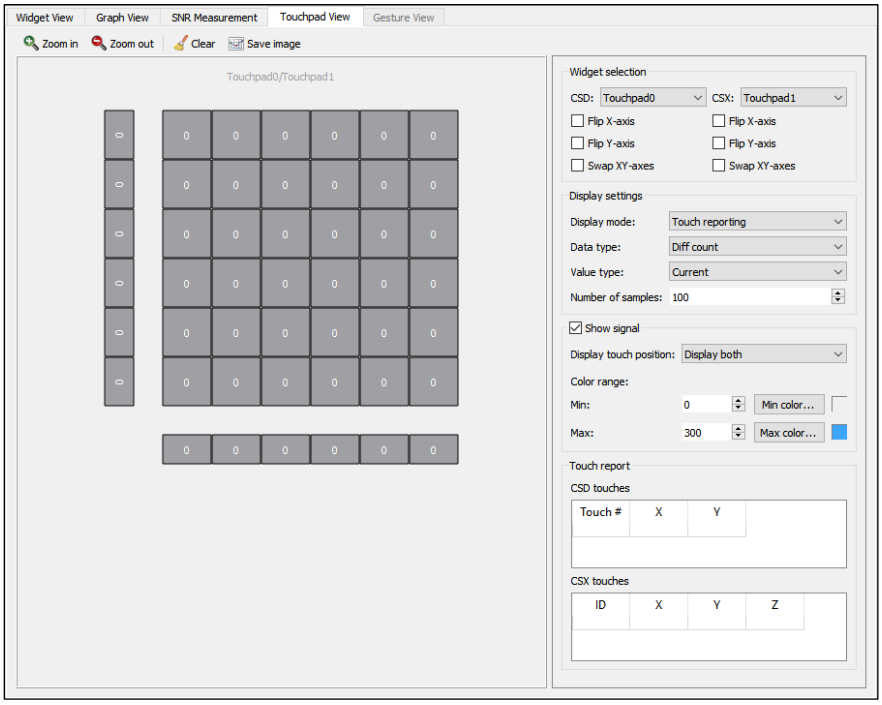
SNR Measurement Procedure

1. Connect to the device and start communication (by pressing **Connect**, then **Start** on the toolbar).
2. Switch to the SNR Measurement tab.
3. Select a sensor by clicking it in the Widget Explorer Pane.
4. Ensure no touch is present on the selected sensor.
5. Press **Acquire Noise** and wait for the required count of samples to be collected.
6. Observe the Noise label is updated with the calculated noise average value.
7. Touch the selected sensor to produce signal on it.
8. Press **Acquire Signal** and wait for the required count of samples to be collected.
9. Observe the Signal label is updated with the calculated signal average value
10. Observe the SNR label is updated with the SNR (signal-to-noise ratio).

Touchpad View Tab

This tab visually represents signals and positions of a selected touchpad widget in the heatmap form. Only one CSD and one CSX touchpad can be displayed at a time.

Note The Touchpad View tab is not visible when there are no touchpad widgets in the configuration.



Toolbar

- **Zoom in** – Zooms in the Touchpad widget.
- **Zoom out** – Zooms out the Touchpad widget.
- **Clear** – Clears the touchpad.
- **Save image** – Opens a dialog to save the Touchpad View tab as an image. Supported formats: .PNG, .JPG, .BMP.

Widget Selection

Consists of the configuration options for mapping the physical touchpad orientation to the identical representation in the heatmap:

- **CSD combo box** – Selects any CSD touchpad to display in the heatmap.
- **CSX combo box** – Selects any CSX touchpad to display in the heatmap.
- **Flip X-axis** – Flips the displayed X-axis to match the orientation of a physical touchpad.
- **Flip Y-axis** – Flips the displayed Y-axis to match the orientation of a physical touchpad.
- **Swap XY-axes** – Swaps the X- and Y-axes for the touchpad.

Display settings

Manages heatmap data to display. These options are applicable for a CSX touchpad only.

- **Display mode** – The drop-down menu with 3 options for the display format:
  - **Touch reporting** – Shows current detected touches only.
  - **Line drawing** – Joins the previous and current touches in a continuous line.
  - **Touch Traces** – Plots all the reported touches as dots.
- **Data type** – The drop-down menu to select the signal type to display: Diff count, RawCount, and Baseline. Under Multi-frequency mode, RawCount and Baseline are available for three channels: 0, 1, 2.
- **Value type** – The drop-down menu to select the type of a value to display:
  - **Current** – The last received value.
  - **Max hold** – The maximum value out of latest “Number of samples” received.
  - **Min hold** – The minimum value out of latest “Number of samples” received.
  - **Max-Min** – The difference between maximum and minimum values.
  - **Average** – The average value of latest “Number of samples” received.
- **Number of samples** – Defines a length of history of data for the **Line Drawing**, **Touch Traces**, **Max hold**, **Min hold**, **Max-Min,** and **Average** options.

Show signal

Enables displaying data for each sensor if checked, otherwise displays only touches. This option is applicable for the CSX touchpad only.

- **Display touch position** – Selects the touchpad from which data is displayed in the heatmap. The three options:
  - Display only CSX
  - Display only CSD
  - Display both
- **Color range** – Defines a range of sensor signals within which the color gradient is applied. If a sensor signal is outside the range, a sensor color is either minimum or maximum out of the available color palette.

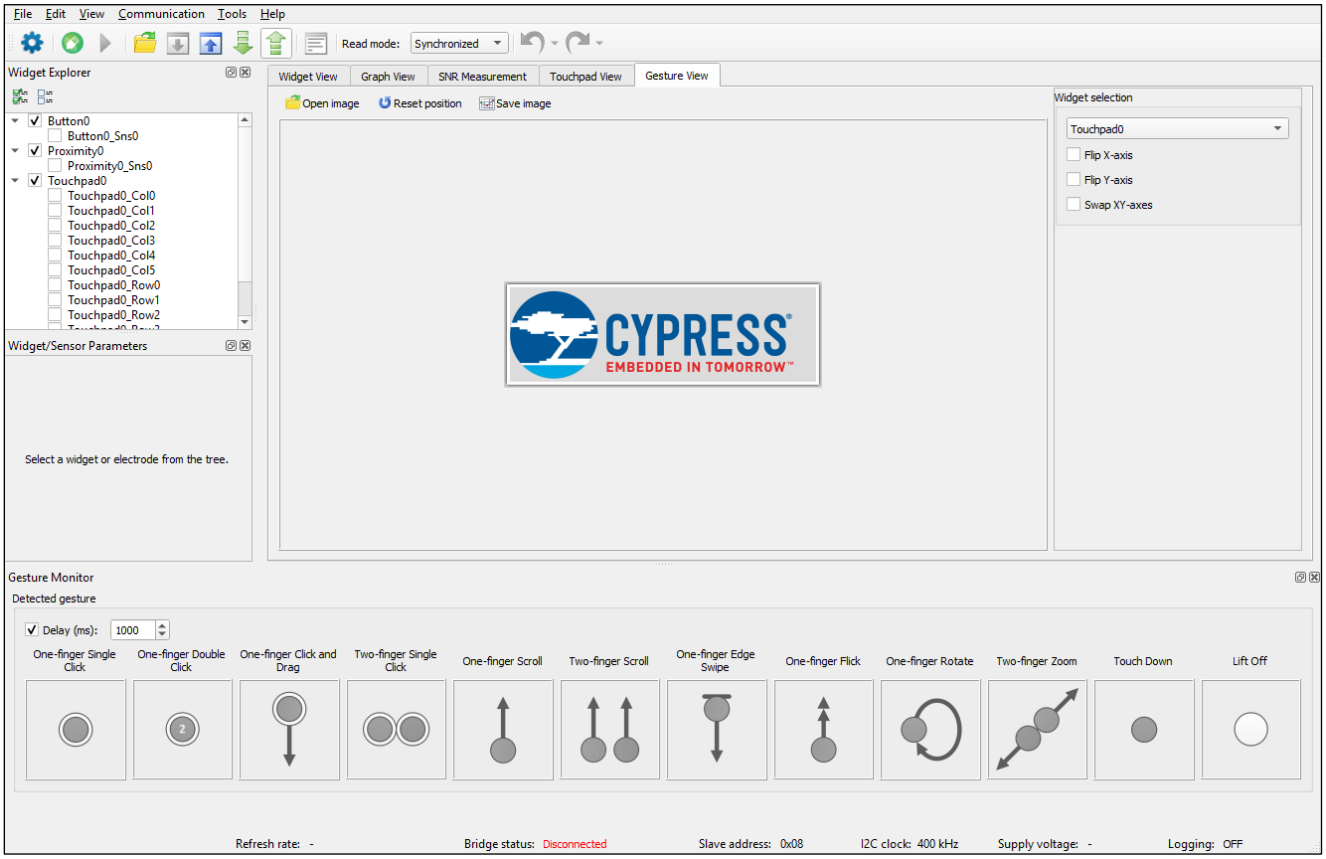
Touch report

- **CSD touches table** – Displays the current X and Y touch position (Z value is always 0) of the detected touches on the CSD touchpad. If the CSD touchpad is neither configured nor touch is detected, the touch table is empty. When two-finger detection is enabled for a CSD touchpad, then two touch positions are reported.
- **CSX touches table** – Displays the current X and Y touch position and Z values (amplitude) of the detected touches on the CSX touchpad. If the CSX touchpad is neither configured nor touches is detected, the touch table is empty. The middleware supports simultaneous detection up to three touches for a CSX touchpad touch, so the touch table displays all the detected touches.

Gesture View Tab

This tab visually represents evaluation and tuning of gestures (from one widget at a time).

Note The **Gesture View** tab is not visible when there is no gesture widget in the configuration.



Note Synchronized communication mode or UART communication is recommended for Gesture validation, to make sure no gesture event such as a touchdown or lift-off is missed during communication.

Toolbar

- **Open image** – Opens a custom image instead of Cypress logo.
- **Reset position** – Resets the image position and zoom. The image is moved to the center of the panel.
- **Save image** – Opens a dialog to save the Gesture View tab as an image. Supported formats: .PNG, .JPG, .BMP.

Widget Selection

Allows selecting a widget and controls that the display in the Tuner matches the orientation physical widget on hardware.

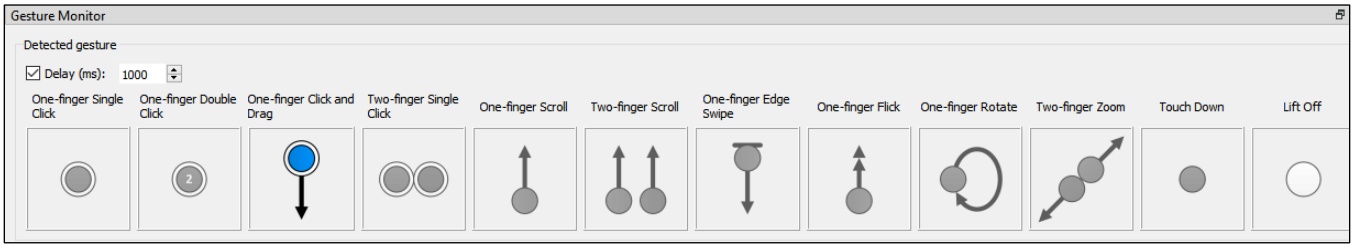
- **Combo box** – Selects the widget with Gesture enabled to display the Gesture from it on this pane.
- **Flip X-axis** – Flips the direction of the X-axis to match the orientation of a physical widget.
- **Flip Y-axis** – Flips the direction of the Y-axis to match orientation of a physical widget.
- **Swap XY-axes** – Swaps the X- and Y-axes to match orientation of a physical widget.

Image Pane

An image with Cypress logo reacts to the detected gestures (scroll and zoom) and moves around the pane. You can change the image by clicking the **Open image** tool button.

Detected Gesture

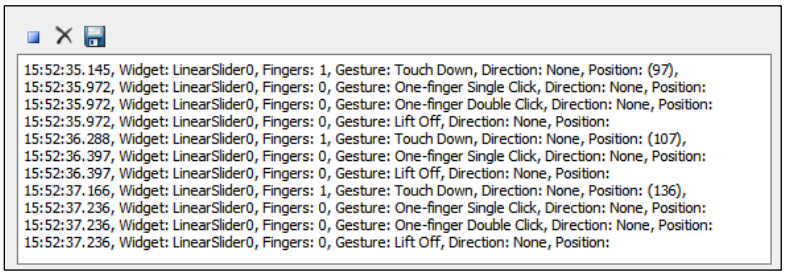
Provides visual indication for a detected Gesture.



If the Delay checkbox is enabled, a Gesture picture is displayed only for the specified time-interval. If disabled, the last reported gesture picture is displayed until a new Gesture is reported.

Gesture Event History

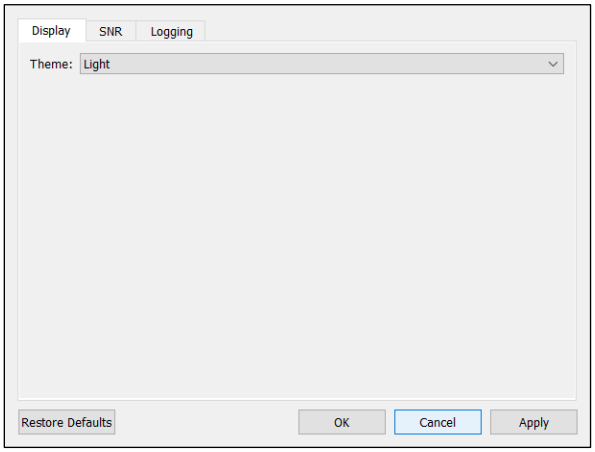
Logs the detected gestures information.



Tuner Configuration Options

The Tuner application allows setting different configuration options with the Options dialog. Settings are divided into groups:

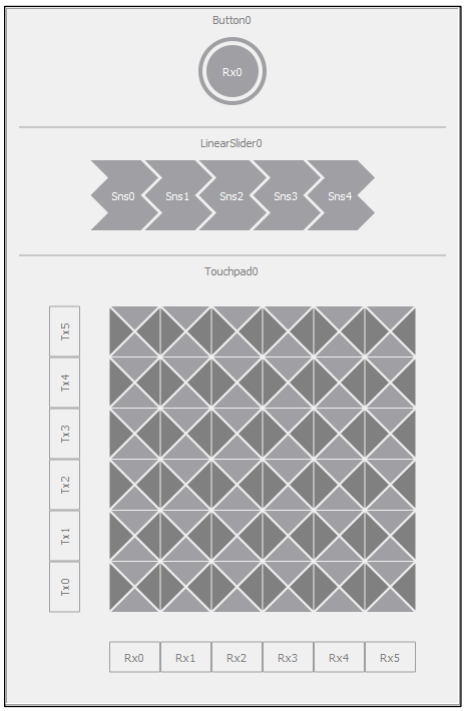
Display Options



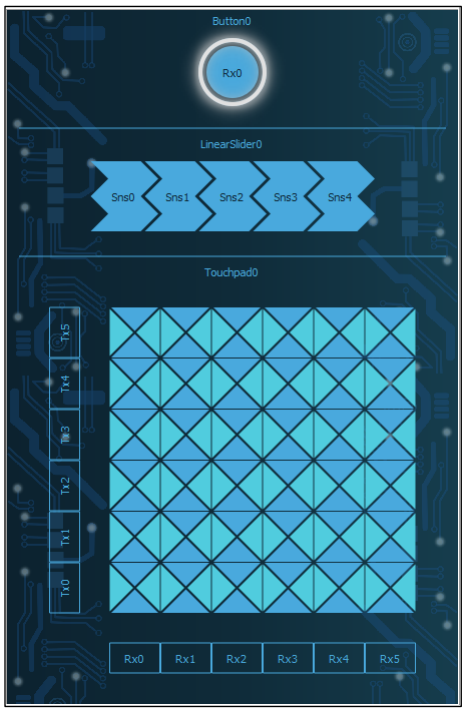
Theme

Defines the visual style of widgets.

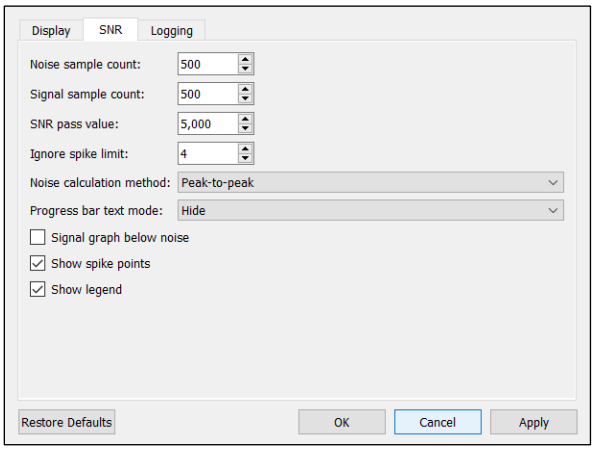
Light theme



Dark theme

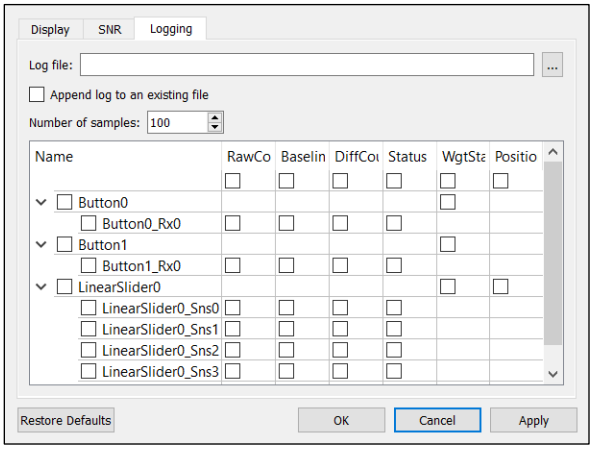


SNR Options



See the SNR Options section for a description of this tab.

Logging Options



- **Log File** – Selects a *csv* file and its location to store information.
- **Append log to an existing file** – If this option is enabled, the selected file cannot be overwritten and expanded with new data. When this option is not enabled, the selected file can be overwritten.
- **Number of samples** – Defines a log session duration in samples.
- **Data configuration check box table** – Selects data to collect into a log file.

References

For more information, refer to the following documents:

- Eclipse IDE for ModusToolbox User Guide
- CapSense Configurator Guide
- Device Configurator Guide
- PDL API Reference Guide
- Device Datasheets

- Device Technical Reference Manuals.

Troubleshooting

Problem	Workaround
On common Linux distributions, the serial UART ports (usually /dev/ttySx or /dev/ttyUSBx devices) belongs to the root user and to the dialout group. Standard users are not allowed to access these devices.	An easy way to allow the current user access to the Linux machine's serial ports is by adding the user to the dialout group. This can be done using the following command: `$sudo usermod -a -G dialout $USER` **Note** For this command to take effect, you must log out and then log back in.
On Linux, attempts to set up or start the CapSense Tuner communication leads to the Tuner closing without any messages.	To enable Tuner communication under Linux, install the udev rules for KitProg3: • Disconnect the KitProg device. • Execute in the terminal (root access required): `sh $CYSDK/tools/fw-loader/udev_rules/install_rules.sh` • Reconnect the KitProg device.
On macOS, the CapSense Tuner can't be launched from the Launchpad.	Launch the Tuner from within the Device Configurator or use the command line option.
On common Linux distributions, the Tuner forbids communication protocol selection after re-plugging KitProg during communication.	Refer to the “Installation Procedure on Ubuntu Linux (x64)” section in the *Cypress Programmer 2.1 CLI User Guide*.
KitProg3 UART is accessible but not able to read data on Linux kernel 4.15 and above or Mac OS X 10.13 and above.	1. Use third-party UART to USB bridge. 2. Update the KitProg3 firmware to version 1.11.243 or above.

Note The Tuner provides information about the root cause of connection to KitProg in an operation log. The operation log is located in:

Windows: *<user_home>\AppData\Local\Cypress Semiconductor Corporation\capsense-tuner\capsense-tuner.log*

Linux: */home/<user_home>/.config/Cypress Semiconductor Corporation/capsense-tuner/capsense-tuner.log*

macOS: */Users/<user_home>/Library/Preferences/Cypress Semiconductor Corporation/capsense-tuner/capsense-tuner.log*

Version Changes

This section lists and describes the changes for each version of this tool.

Version	Change Descriptions
1.0	New tool.
1.1	Added UART interface.
	Added possibility to save different views as images.
	Added zoom and pan functionality for graphs.
	Fixed minor issues.
2.0	Added "IDAC gain index" parameter. Changed the data storage location from a header (.h) file to an XML-based *.cycapsense file.

Version	Change Descriptions
	For backward compatibility, the configurator is still able to load the header (.h) file that contains the legacy format configuration. But, if the legacy header (.h) with the configuration is passed via a command-line parameter, a message appears saying that the .h file is not supported. Added the **Import** and **Export** options to the **File** menu that enable importing and exporting the configuration file from and into the external file. Added the **Reset View** command to the **View** menu that resets the view to the default. Added Multi-frequency support that enables selecting RawCount and Baseline. Added the UART option description in Status Bar. Changed generation of the middleware initialization structure according to the changes in CapSense v2.0 middleware (related to added fields for flash memory optimization, fixed defect with RawCount filters config, IDAC gain index, etc.). Added support for selecting multiple sensors in some view and reflecting the selection in other views. Added handling of invalid command line arguments. Added highlighting of modified properties in the property grid with bold. Added a warning if opening a broken configuration file. Fixed memory leaks. Fixed Graph View issues under high load of application.
3.0	Added the Undo / Redo feature.
	Changed the list of UART baud-rate possible values to match the values supported by KitProg. The highest rate is 3000000.
3.10	Updated versioning to support patches.
	Added a popup notification when the Apply to device command completes.
	Increased the communication speed between the device and Tuner when using the I2C interface.
	Improved the tool performance for loading large configurations.
	Fixed the issue with applying changes to the device when using the UART communication interface: now, many changes (>16) are applied correctly.
3.11	Updated versioning to support the updated backend, for detail, see Device Configurator User Guide.