CYAT6165X/8165X/8168X touchscreen controller

Tuning best practices

About this document

Scope and purpose

This document covers the tuning process for the Infineon CYAT6165X/8165X/8168X family of touchscreen controllers. It is written for the firmware versions shown on the front page. The tuning guide should be generally applicable to other firmware releases, however, there may be some differences in the tuning parameters. It is therefore recommended to use the tuning guide for the correct FW version (see Table 6).

This document version targets:

- CYAT6165x (TSG6L slider) base firmware version 1.5.1093522
- CYAT7165x/CYAT8165x (TSG6L panel) base firmware version 1.5.1093522
- CYAT8168x (TSG6XL) base firmware version 1.4.1084642

Intended audience

The intended audiences for this document are design engineers, technicians, and developers using Infineon touch controllers.

Table of contents

1 Overview

1.1 Document organization

Tuning is the process of characterizing the touch panel electrical characteristics and optimizing the touch controller to achieve the desired system performance. Thus, this document is a guide, intended to help those interested in tuning their touch screen solution.

There are a very large number of variables that control many aspects of a touch solution. Frequently different parameters have opposing effects. With so many variables there are usually many solutions (or even possibly no solution) to achieve a desired functional performance. Similarly, with so many variables there are as many possible ways to tune the solution. This document provides a process that is effective for most projects.

This document is structured as follows (also see the tuning flow in Section 2.1):

Step 1. Introduction (Ch.2)

- Aim – Information only
- Content – Familiarize reader with basic knowledge and issues needed to tune touch

Step 2. Setup before HW available (Ch.3):

- Aim – Create blind-build with best chance of detecting touches
- Content – Create the TTHE project. Set HW-constant TTHE parameters. Disable advanced features, to enhance the chance of the blind-build detecting touches.

Step 3. HW bring-up (Ch.5):

- Aim – Bring-up the HW setup and verify the pinout
- Content – Check the HW setup and pinout

Step 4. [OPTIONAL] Development release:

- Aim – Produce a usable touch solution allowing host and UI development, by performing only the basics of the next 3 steps, with only sanity testing. The release is not intended to meet EMC or performance requirements. Robust touch detection is the only goal.
- Content – N/A

Step 5. Analog front-end tuning (Ch.6):

- Aim – TX waveform that maximizes SNR and passes EMC requirements
- Content – Correct tuning of the analog front end is imperative to the configuration of everything that follows. This tuning section must therefore be completed before any fine-tuning of the digital section.

Step 6. Basic touch processing tuning (Ch.7):

- Aim – Basic touch reporting
- Content – Once the analog front-end performs acceptably, the correct recognition of objects can be configured. This step provides logical steps to generate basic touch reports.

Step 7. Advanced touch processing tuning (Ch.8):

- Aim – Full-featured touch release
- Content – After the basics are accomplished, it is appropriate to tune for detection of different objects and enable other advanced features of the touch controller if desired

1.2 Related documents

Infineon has created a collection of documents to support the design of PSoC™ Automotive Multitouch touchscreen controllers. The following list will guide you in identifying the proper document for your task. PSoC™ Automotive Multitouch technology is Infineon confidential information and is protected through a Non-Disclosure Agreement (NDA). Some of the documents are not publicly available on the Infineon website.

Table 1 Product specifications

Document number	Document title	Description
001-97033	CYAT8168X (61, 71, 77, 88 I/Os), PSoC™ Automotive Multitouch All-Points Touchscreen Controller	Datasheet containing features, touchscreen performance and electrical specifications, and package information.
001-99382	CYAT6165X/8165X/8168X Technical Reference Manual (TRM)	The technical reference manual for CYAT6165X/8165X/8168X.
002-10076	CYAT8165X (48 I/Os), Automotive PSoC™ Automotive Multitouch All-Points Touchscreen Controller Datasheet	Datasheet containing features, touchscreen performance and electrical specifications, and package information.
002-19012	CYAT6165X (41 I/Os), Automotive PSoC™ Automotive Multitouch All-Points Slider Controller Datasheet	Datasheet containing features, touchscreen performance and electrical specifications, and package information.

Table 2 Solution specifications

Document number	Document title	Description
001-49389	PSoC™ Automotive Multitouch Touchscreen Controller Performance Parameters	Contains Infineon touchscreen parameter definitions, justification for parameters, and parameter test methodologies.
001-50467	PSoC™ Automotive Multitouch Touchscreen Controller Module Design Best Practices	A system-level design guide for building a capacitive touchscreen module, covering topics such as touchscreen traces, shielding, mechanical design, FPC/PCB design, and display considerations.
001-83948	Touch Tuning Host Emulator User Guide	User guide for the Touch Tuning Host Emulator. It is included with the installation of the TTHE software, at location *[installation folder]\[version]\documentation*.

1.3 Document conventions

Table 3 Document conventions

Convention	**Usage**
Italics	Used for file names and paths, and reference documentation: Read about the *sourcefile.hex* file in the *PSoC™ Designer User Guide*.
Bold Italics	Used for terms described in the Glossary of this manual.
Bold	Used for commands, menu paths, and GUI elements in procedures: Click the **File** icon and then click **Open**.
[**Bracketed Bold**]	Used for keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
File > Open	Represents menu paths: **File** > **Open** > **New Project**
Cambria Math	Used for equations: $2 + 2 = 4$

`Courier New`	Used for code examples:
	`-BBootloaderRAM:0`
	`-BInterruptRAM:0`

Group: Parameter	Used for Touch Tuning Host Emulator Configurable Parameters and Configurable Registers, for example **TSS: SCANNING_MODE_BUTTON**
Group: Parameter: Field	Used for Touch Tuning Host Emulator Configurable Parameters and Configurable Registers with sub-fields, for example **TSS: REFGEN_CTL: RXDAC**

1.4 Units of measure

Table 4 Units of measure

Symbol	**Unit of measure**
°C	degrees celsius
µA	microamperes
µF	microfarads
µs	microseconds
µV	microvolts
µVrms	microvolts root-mean-square
Ω	ohms
b	bit
dB	decibels
fF	femtofarads
g	gram
Hz	hertz
k	kilo, 1000
K	210, 1024
KB	1024 bytes
Kbit	1024 bits

Symbol	Unit of measure
Kbps	kilobits per second
kHz	kilohertz
Mbps	megabits per second
kΩ	kilohms
MHz	megahertz
MΩ	megaohms
mA	milliamperes
ml	milliliters
min	minutes
mm	millimeters
ms	milliseconds
mV	millivolts
nA	nanoamperes
ns	nanoseconds
nV	nanovolts
pF	picofarads
pp	peak-to-peak
ppm	parts per million
s	seconds
sps	samples per second
V	volts

1.5 Acronyms

Table 5 Acronyms

Acronym	Description
AC	alternating current
ADC	analog-to-digital converter
AFH	adaptive frequency hopping
CDC	capacitive to digital converter
CMF	common mode filter
CSV	comma-separated values
DAC	digital-to-analog converter
DC	direct current
DVK	development kit
EMC	Electromagnetic compatibility or reducing emission
EMI	Electromagnetic interference or immunity to external noise
FF	fat finger
FW	firmware

Acronym	Description
GIDAC	global IDAC
GUI	graphical user interface
HTI	highest touch intensity
IDAC	current digital-to-analog converter (I = current)
IIR	infinite impulse response
IMO	internal main oscillator
I/O	input/output
ITO	indium tin oxide
LCD	liquid crystal display
LFT	look-for-touch
LO	large object
LSB	lease significant bit
LTI	least touch intensity
MC	mutual capacitive or mutual-cap
NDA	nondisclosure agreement
PWC	pulse width control
RC	resistor-capacitor
RF	radio frequency
RX	receiver/reception
SAR	Successive Approximation Register
SC	self-capacitive or self-cap
SNR	signal-to-noise ratio
SWD	serial wire debug
TBD	to be defined
TMA	touch multitouch all-points
TMG	touch multitouch gesture
TRM	technical reference manual
TTHE	touch tuning host emulator
TX	transceiver/transmission

1.6 Base firmware versions

It is important to use the document for the correct base FW version. The table below lists all the official base FW release versions, with the corresponding tuning document version.

Table 6 Base firmware versions

Document revision	CYAT6165X	CYAT8165X	CYAT8168X
**	Undefined	Undefined	Undefined
*A	n/a	n/a	1.2.935791
*B	n/a	1.0.951457	1.2.935791
*C	n/a	1.1.966753	1.2.935791
*D	n/a	1.2.1004589	1.2.935791
*E	1.2.1004589	1.2.1004589	1.3.1036671
*F	1.2.1004589	1.2.1004589	1.3.1036671
*G	1.4.1053878	1.4.1053878	1.4.1084642
*H	1.4.1053878	1.4.1053878	1.4.1084642
*I	1.4.1053878	1.4.1053878	1.4.1084642
*J	1.5.1093522	1.5.1093522	1.4.1084642

2 Introduction

This section introduces the basic tuning flow as well as some of the fundamentals of touch screens and sensing their capacitance, and the tools that will be needed in the tuning process.

The tuning process requires the knowledge of a vast array of material, it would be counterproductive to present everything in one document. The resulting document would be unwieldly, difficult to navigate, and impossible to maintain. Thus, this section is limited only to the essentials such as how sensing is performed and what might be expected in a real system that would need to be accounted for.

During a tuning effort, the user will inevitably need to jump to different sections of this guide, search for terms, and so on. Because of this, it is strongly recommended to enable the “Back” or “Previous View” navigation icon in your pdf viewer. In Adobe Acrobat Reader DC, 2017 Release, this is done by right-clicking on the menu bar, then selecting “Show Page Navigation Tools”, and making sure that “Previous View” is enabled. After doing this, a back arrow should be available in the menu bar, allowing faster navigation through the document.

2.1 Tuning flow

The tuning flow is project dependent. However, this document is organized in the same order as the most frequent tuning process. There may be some repetition, most notably if an initial release is needed for host interface development and early customer samples for UI evaluation.

This tuning guide has been organized to match the most frequent tuning flow. A very brief overview of each section is given in the Document organization Section (see Section 1.1).

The tuning flow has one stage that is not represented as a chapter in this guide, the initial quick tuning pass to produce a “development release”. While this stage is not part of a pure tuning flow, it is almost always required in a real project. As soon as the HW is available, the touch is required to function robustly so that host and UI development can proceed apace. At this stage, there is not the time to perform the EMI/EMC emissions testing required to produce the final analog front-end (AFE) settings. Therefore, performing the detailed tuning of the digital section (without the AFE being fixed) would result in unnecessary rework. It is therefore suggested to do a quick/coarse tune of the AFE, basic touch processing, and as much of the advanced touch processing as required by the particular project, so that the development release can be produced.

The basic flow is shown in the following figure:

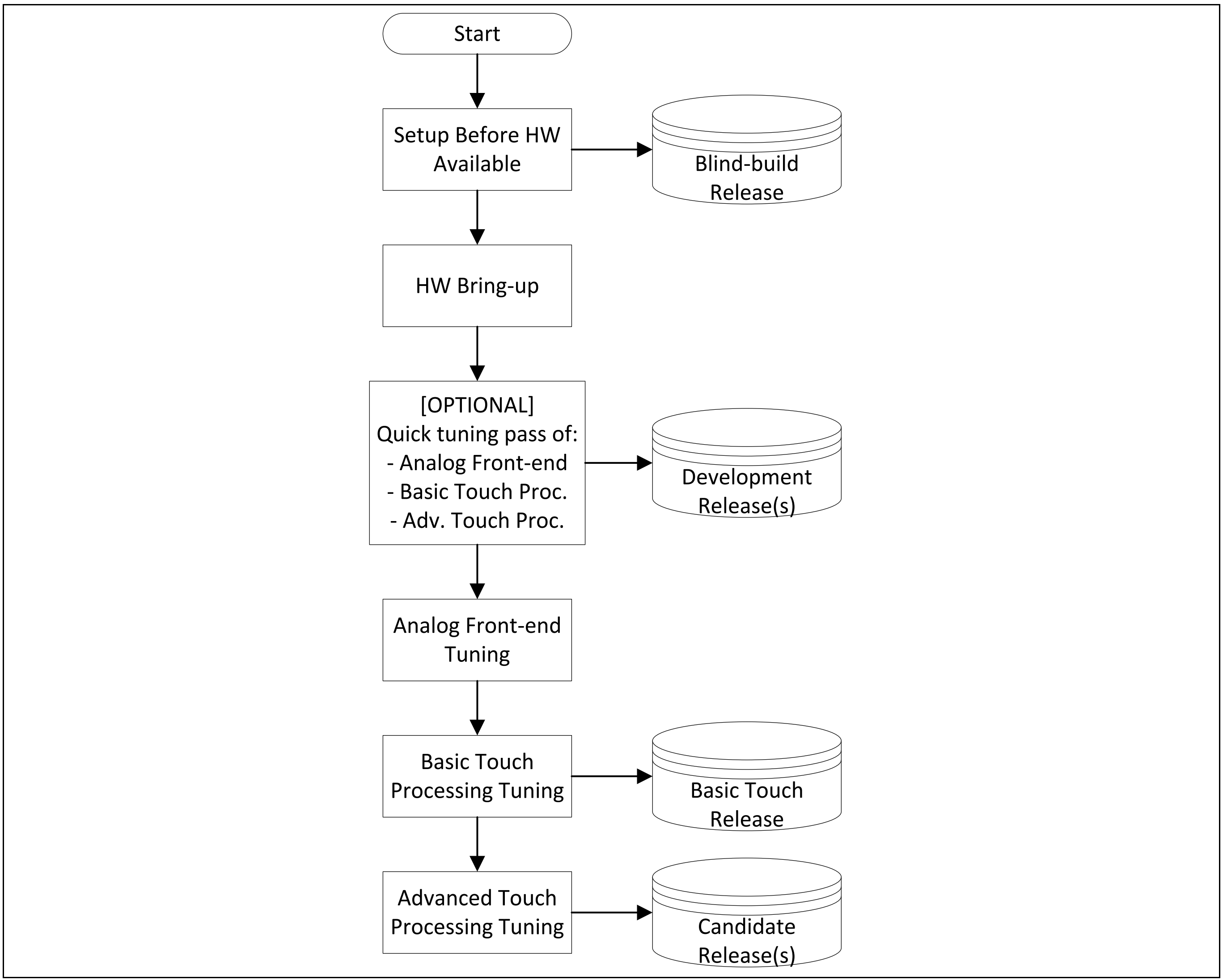


Figure 1 Tuning flow

2.2 Default parameter values

At the end of most sections is a list of parameters referenced in that section. This parameter list contains a brief description of the parameter, valid range, and default value. The default value can be one of two things:

- Setting to disable the feature, for example:
  - **Device Setup: CLIPPING_X_LOW** defaults to 0.
  - **Calibration: DYNAMIC_CALIBRATION_ENABLED** defaults to Disabled.
- Typical value that is likely to work with most panels, for example:
  - **TSS: MTX_ORDER** defaults to 4, which is unlikely to be the final setting, but gives good SNR improvement with minimal chance of increasing the refresh rate.
  - **TSS: TX_PERIOD_MC** defaults to 235, which is slower than most panels require, but it is likely to work with almost all panels.

2.3 Capacitance and capacitive sensing

The touch controller supports both mutual-capacitive (mutual-cap or MC) and self-capacitive (self-cap or SC) sensing. Figure 2 shows a simplified view of the sensor model. The simplified model is structured to represent the parasitic elements that are characteristic of a unit cell in a touch screen. Thus, there is trace resistance on both the TX and RX lines. There is self-cap for both TX and RX with respect to ground. And then there is the mutual-cap between TX and RX. A finger touch grounded via the human body adds C_{TxF} and C_{RxF} self-cap and reduces mutual-cap between sensors.

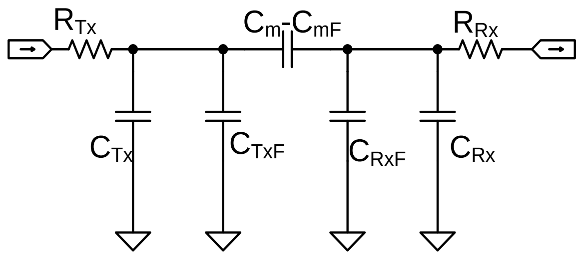


Figure 2 Simplified touched sensor model

2.3.1 Mutual-cap sensing

Mutual-cap (mutual-capacitance or MC) sensing is used to scan each TX/RX intersection. This results in a map of capacitances for the entire surface of the panel, where an object can be correlated to a relative position. Thus, MC sensing is the primary method to determine the location of one or more fingers touching a screen.

The mutual-cap of a unit cell is measurable by examining the charge stored in the cell. This can be done by simply moving charge through the cell and measuring the change. And one way to move charge is to simply modulate the voltage across the cell and monitor the current. Figure 3 shows a simplified diagram of one such sensing circuit.

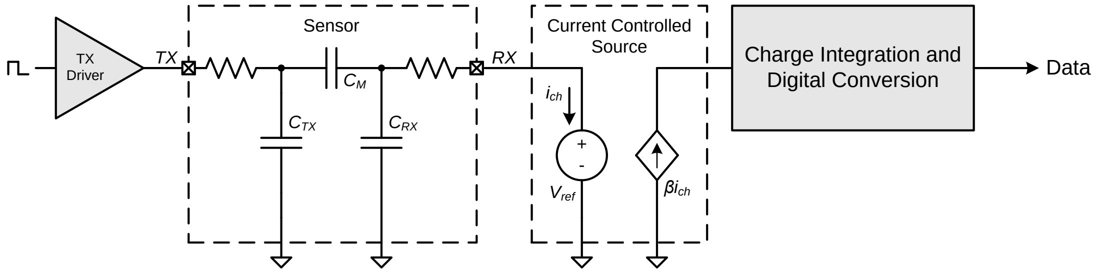


Figure 3 Simplified mutual-cap sensing

The input of the sensor shown in Figure 3 is driven between 0 and VTX volts. This induces a current that flows through the RC network and into the voltage source at the input of the channel. Note the voltage source at the input of the channel is typically one half of V_{DDA}. Thus, a net charge is transferred into or out of the sensing channel that is proportional to the changing voltage and the capacitance. The net change in charge is represented as:

$$\Delta Q_M = (C_M)(\Delta V_{tx})$$

Equation 1

The net charge is mirrored in the channel for conversion into the digital domain, which is discussed a little later.

2.3.2 Self-cap sensing

Self-cap (self-capacitance or SC) sensing is used to determine the capacitance of rows and/or columns of sensors. While MC, a much longer process, is used to determine finger position(s), SC is used to determine if there is something on the panel that needs further inspection. Thus, SC information is used to determine when it is appropriate to perform a MC scan (i.e. faster and saves energy). In addition, the SC information is used to assist in determining fingers in the presence of moisture on the touch screen.

The self-cap of an array of cells is measurable by examining the charge stored relative to the ground or other signals near the sensor. This can be done by simply moving charge through the self-cap and measuring the change. And one way to move charge is to simply modulate the voltage across the cell while keeping the voltage across the sensor mutual-cap at zero volts (i.e. zero volts means zero current through the mutual-cap). Then monitor the current into or out of the sensing channel. Figure 4 shows a simplified diagram of one such sensing circuit.

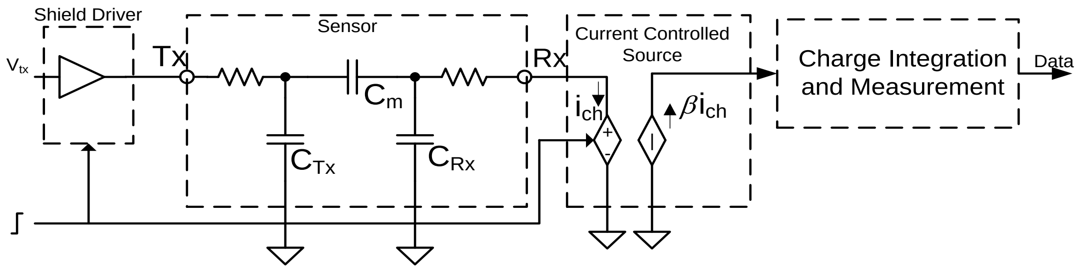


Figure 4 Simplified self-cap sensing

Note that the net current through the mutual-cap is effectively zero; therefore, the net charge transfer through the mutual-cap is also zero. Only the charge related to the total self-cap is transferred and mirrored in the channel:

$$\Delta Q_P = (C_P)(\Delta V_{tx})$$

Equation 2

And just like the mutual-cap sensing method, the net charge is mirrored in the channel for conversion into the digital domain.

2.3.3 The channel and measuring charge

The channel is designed to, quite literally, count packets of charge. Sequentially the charge from the sensor is first integrated on to a holding capacitor in the channel. After the integration period is finished the capacitor is balanced. During the balancing period charge is counted at a rate of 48 MHz until the capacitor is balanced or the period finishes. Note that it is expected that the calibration of the channel to the touch screen will always yield a situation where the integrating capacitor is always balanced before the end of the period.

It is important to note that the channel is divided into two halves. Thus, there are two integrating capacitors. While one capacitor is integrating the other is balancing. This tends to lead to a design where one capacitor represents the positive (incoming) charge, and the other represents the negative (outgoing) charge, although the actual design is not strictly limited to this. A simplified view of the channels is shown in Figure 5.

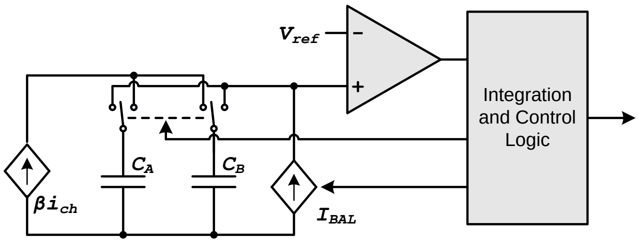


Figure 5 Simplified channel model

There is also additional logic to assist in digital integration of data in the channel to improve the resolution. Thus, the channel is often configured to take multiple samples (pulses) and accumulate the data to generate a larger total (or higher resolution).

2.4 Touch screens, displays, and more

Every ITO pattern requires a different customized tuning/configuration of the touch sensing solution. For example, a Manhattan (MH3) pattern will perform very differently from a Dual Solid Diamond (DSD) pattern. Pattern design, materials, material stack-up, and the environment affect the decisions made in the tuning process. This section puts some basic perspective on what makes up a touch screen and how it is affected.

2.4.1 Bonding and stack-up

Understanding the stack-up and bonding is essential, whether it is a sensor-on-lens design, a design that employs an air gap, or a new process is used where little is known about the bonding. Different display bonding methods will yield a different amount of capacitive coupling to the RX lines. For example, a UV-curable glue is less favorable than optically-clear adhesive (OCA). A simple example stack-up is shown in Figure 6.

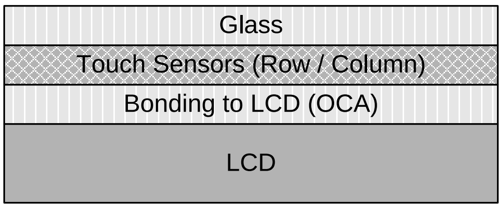


Figure 6 Example stack-up

When considering the bonding and stack-up, the parameters to be understood are:

- The distance (*d*) between the display and the touch screen panel

- The type of material (such as air, OCA, and UV glue) between the two layers, which relates to the permittivity (ε_r), and
- The surface area (*A*) of the sensors

These parameters determine the capacitive coupling as described by the following equation.

$$C_p = \frac{\varepsilon_0 \varepsilon_r A}{d}$$

Equation 3

Notice that higher permittivity materials as well as larger surface area will increase the capacitive coupling. Likewise, lower distance will also increase coupling. This equation is important to keep in mind when thinking about sensing very small capacitance in the presence of noise.

2.4.2 Display noise coupling

Capacitive sensing is usually coupled with a display. Different sensor patterns will have a different amount of coupling (C_P) to the display. For example, an MH3 pattern has large TX traces on the bottom layer, and narrow RX traces on the top layer. The benefit of placing the TX traces on the bottom layer is they provide shielding for the RX layer from the display, which reduces coupling. Designs that employ metal mesh, instead of ITO, do not have quite as much benefit from any shielding layers because the metal material is not solid across the entire cell but rather an outline.

Thus, understanding what is behind the touch screen is fundamental when starting to tune, whether it is a ground/shield layer or a display with difficult harmonics. A higher amount of capacitive coupling between the display and the touch screen panel will yield more display energy measured on the RX channel in the touch screen controller, which is usually detrimental to sensing capacitance. Figure 7 shows an example spectrum of a real display.

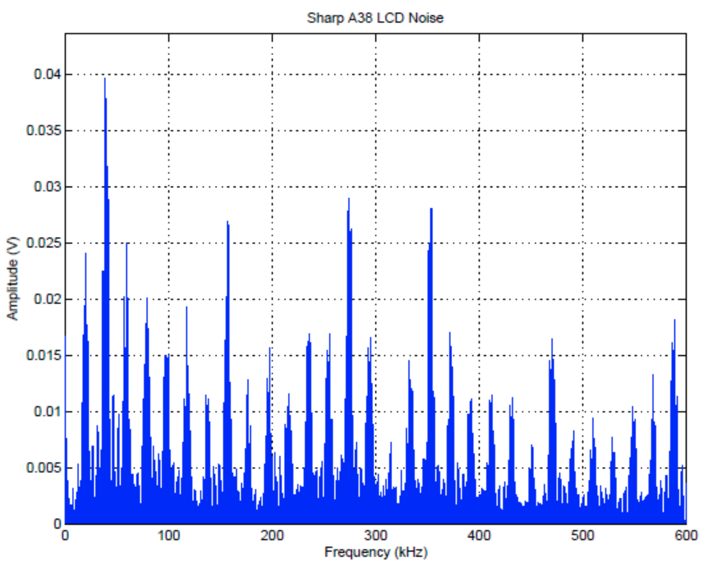


Figure 7 Example display noise spectrum

Notice the peaks and valleys in the spectrum in Figure 7. When tuning with a display, a reasonable goal is to avoid the peaks and aim for the valleys when selecting a TX frequency. Refer to Section 8.2 for details.

A project may require the touch scanning to be synchronized with the display driver. The need for display synchronization may be known before the project starts (such as, in-cell design), or may become a requirement after hardware bring-up (such as excessive display noise). Display synchronization is detailed in Section 8.3. In either case, hardware support is required, so early planning is important. The display driver must provide the synchronization signal(s) (usually HSync and/or VSync, the display's horizontal-line and vertical-frame synchronization) to the touch controller's GPIO.

2.4.3 Sensor pitch

The pitch of the panel affects several other system parameters. If the display size and bonding have already been determined, and a particular pattern is already defined, determining the pitch will also define the amount of parasitic capacitance on the panel. As already mentioned, different patterns will lead to different capacitance; likewise, scaling the unit cell for different pitches will also affect the capacitance because it is scaling the surface area of the unit cell. Also note that the same is true for the lines connecting the sensors. Sensor designs with large RX traces will also have more capacitive coupling, which is important to understand with noise sources involved.

The pitch also influences accuracy and the finger sizes the system will be able to reliably resolve. For finger sizes that are much smaller than the pitch size, the width is too narrow to achieve an accurate measurement. This is akin to signal aliasing because multiple sensors are usually required to accurately determine the position of an object, yet the object is smaller than a single sensor. When tuning, it is important to know the pitch size to identify the limits of the tuning.

2.4.4 Sensor settling

The R-C time constant (τ) of a sensor in the panel determines how fast a given panel can be driven. The time constant is a function of the resistance of the TX and RX lines, as well as the parasitic capacitance of the TX and RX lines. Figure 8 shows the step response of an example sensor (R-C circuit) with R = 5 kΩ and C = 5 pF.

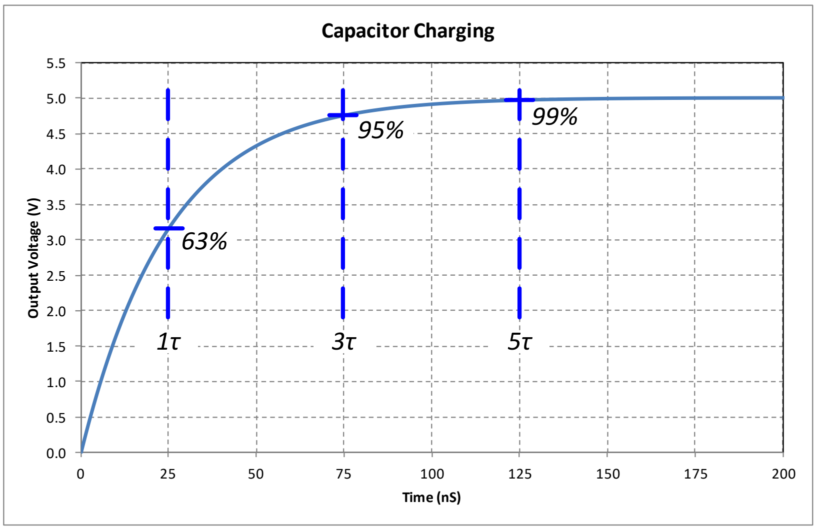


Figure 8 Capacitor charging example

Figure 8 shows the different voltage levels achieved depending on how long the sensor can be charged. The traditional capacitor charging equation is shown in Equation 4.

$$V_{out} = V_{in}\left(1 - e^{-t/\tau}\right)$$

Equation 4

When setting the TX frequency, it is typical to hit the highest frequency possible that allows the panel to reach at least 3τ settling (~95% of the input voltage) at the worst case (slowest settling) intersection on the panel. This means, the 3τ value can be calculated or measured to determine the maximum TX frequency the panel is capable of being driven at. Note that some panels, such as metal mesh designs, will have a 3τ value much smaller than the silicon can drive the panel. In this case, the TX frequency will be silicon limited, not sensor limited.

To determine the maximum TX frequency, drive a square wave onto the panel at a low frequency (25 kHz for example) and measure the time it takes the line to charge up to 95%. Use Equation 5 to determine the maximum acceptable TX frequency. The actual TX frequency will be determined in Section .

$$f_{tx} = \frac{1}{(2)(T_{95\%})}$$

Equation 5

For example, assume the 95% settling point was measured to be ~0.9 µs. Thus, the desired TX frequency is ~555 kHz. Note that this is the desired frequency, but the actual frequency will be limited to 350 kHz, which is the limit of the controller.

2.4.5 Screen overlay

The material type and thickness of the overlay also have a system impact. Systems with thin glass overlays do not have much signal spreading, so the signal width is typically narrower. Thicker overlays have a spreading effect on the signal due to the additional paths for the field lines. This spreading can be beneficial for smaller finger sizes, particularly as it relates to accuracy, but it does have a negative impact on the peak signal (smaller peaks, larger tails). Charger related noise is also mitigated by thicker overlays simply because the capacitive coupling between the finger and the RX line is reduced.

2.4.6 Charger noise

Charger noise is the electrical noise that occurs when a switching DC convertor is connected to device with touch controller. The noise typically develops with respect to earth ground and is common to all points in the device. Because the human body is referenced to earth ground potential, the charger noise is not apparent until the person touches the touch screen or is very close to the screen. A simplified representation of this is shown in Figure 9.

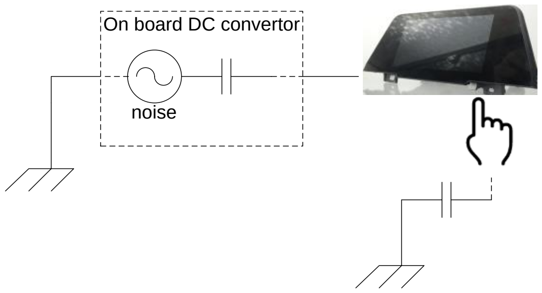


Figure 9 Charger noise

Aside from applying some hardware level change (such as extra filters in the charger), charger noise is typically mitigated in tuning by employing filtering or in some cases allow frequency hopping to find a quiet location to operate.

2.4.6.1 Observing charger noise

Charger noise can be observed by probing the charger ground with an oscilloscope while leaving the probe ground unconnected. Figure 10 shows an example of charger noise captured. The capture shows the harmonics of the grid as well as higher frequency harmonics (~100 kHz) commonly associated with the switching regulator. Note that this is just an example and does not necessarily represent all chargers. Some chargers can have significantly better or worse noise characteristics.

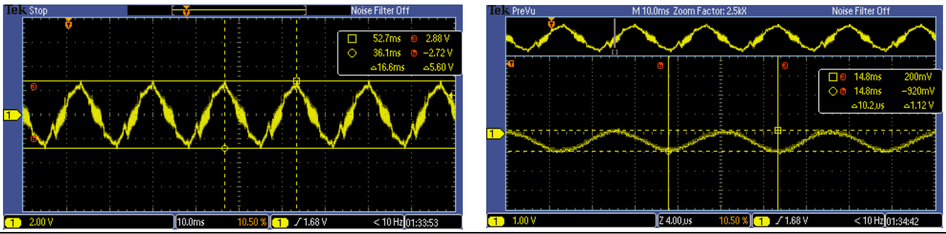


Figure 10 Example measured charger noise

2.5 Calibration

2.5.1 Calibration overview

Calibration can be automatically performed (first power-up or dynamic calibration) or manually triggered by a host command. Dynamic calibration is described in Section 8.16. The host directed calibration command is described in the device TRM.

Automatic calibration at first power-up should only occur in the manufacturing facility, or during product development. After the touch controller is programmed (using SWD not via a bootload), the first time the touch controller enters the touch application (entering the bootloader does not trigger the calibration) a calibration is performed.

The primary purpose of calibration is to determine the channel attenuation ratio, baseline charge injection, and balancing current in an automated but guided fashion. To provide proper guidance to the calibration algorithms it is important to understand general the effects of tuning.

As depicted in Figure 11, tuning can have a big effect on the way the integrator balances. Tuning in one direction can give great sensitivity but result in poor margin for environmental influences. Tuning in the opposite direction can lead to very wide operating range but yield very poor sensitivity to the signal of interest.

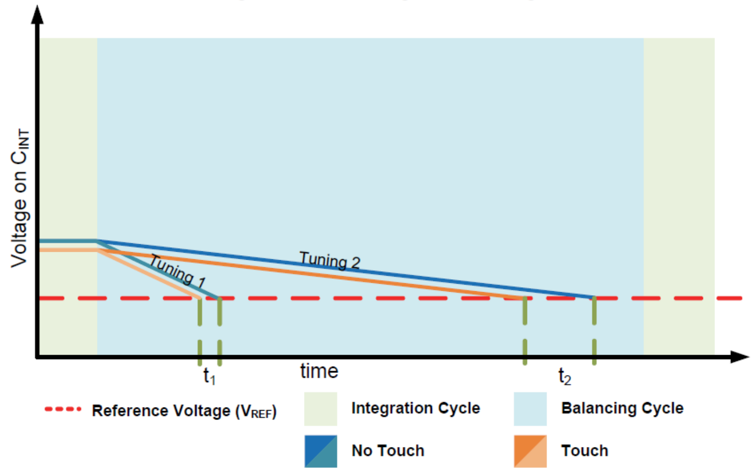


Figure 11 Balancing current for different tuning

In general, if both the attenuation ratio and the balancing current are reduced, the system has higher sensitivity and the count value increases. One might be tempted to simply guide tuning the system to its most sensitive setting, but this is usually a bad idea. The best way to tune any touch screen system is to push for the longest balancing but provide sufficient margin to allow for varying levels of noise under normal operating conditions. In fact, it is suggested to tune the system to accommodate the "normal" noise level (environmental noise, power supply noise to the touch controller, and so on) and display noise, but not charger noise.

To demonstrate some tuning trade-offs, consider a tuning where the attenuator is kept at the same value, but we modify the balancing current. Figure 11 shows two different balancing currents for the same signal level. The two different configurations shown both have benefits and problems. Tuning 2 has a lower balancing current and provides the largest difference counts (t2 > t1). The problem with Tuning 2, as opposed to Tuning 1, is that there is little margin for noise. If the touchscreen is in a system with a noisy display, the incoming charge will have large fluctuations. Figure 12 shows the same two tuning examples with noise.

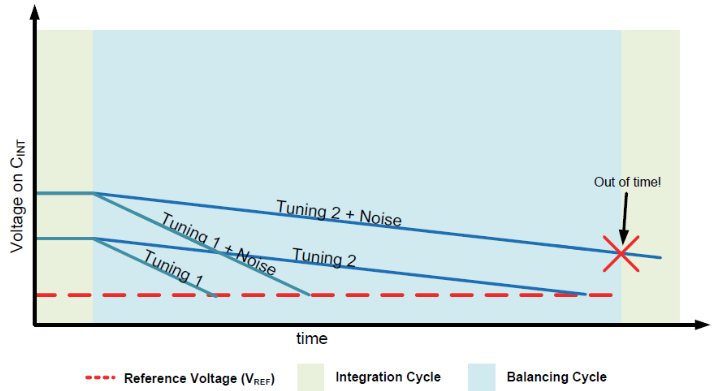


Figure 12 Balancing current for different tuning

Enough margin should be provided that the system has sufficient time to balance the incoming charge in a half TX period but does not provide excessive margin for the worst-case noise condition the system will ever experience. Excessive margin will result in a loss in sensitivity. Because the system retains the residual charge on the integration capacitor, a single noise spike has minimal impact on the system. Consider the following example.

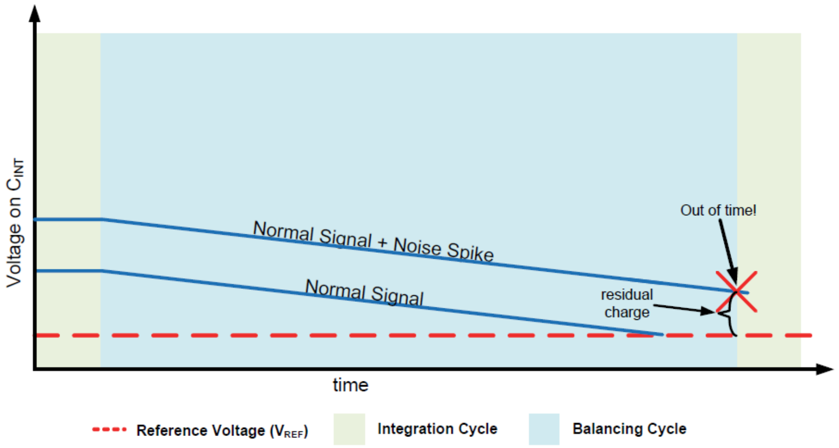


Figure 13 Balancing current for different tuning

Figure 13 shows a system where a noise spike has increased the starting voltage on the integration capacitor to a level where the system cannot balance all the charge in a single half period. If there are enough TX pulses, the residual charge will ultimately be measured. Figure 14 shows what happens to the charge, due to the noise spike, over time. The first TX pulse following the noise spike starts at a slightly lower voltage than the spike. The second TX pulse following the noise spike starts at an even lower voltage still. Ultimately, the voltage works its way back to the "normal" level.

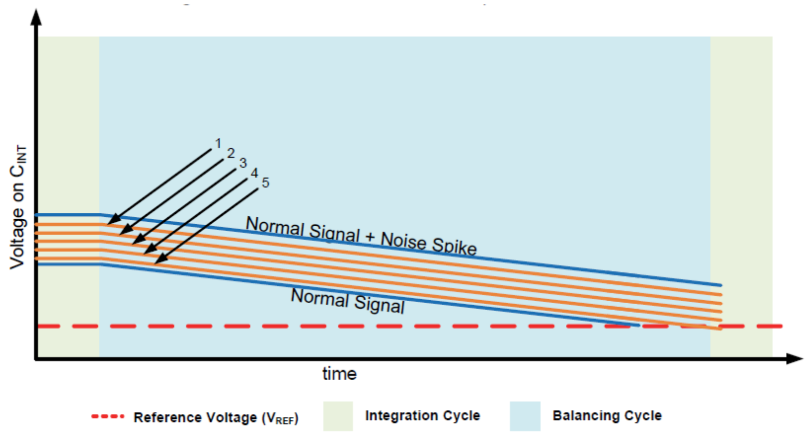


Figure 14 Balancing current for different tuning

2.5.2 Calibration output

The calibration output is a set of:

- Attenuation ratio
- Balancing current (I_{BIAS})
- Baseline current and pulse width for each Sensing mode

For mutual-cap Sensing mode, there is one set of the calibration output for the base TX frequency and for all Charger Armor hop frequencies if Charger Armor is enabled. And the same set of values is globally applied to all the mutual-cap TX/RX intersections. The same is true for Self-cap Sensing mode and CAPSENSE™ buttons. The same set of values is applied to all the corresponding sensors.

	Rx Attenuator	IDAC
Mutual	4.8	45
Self Rx	3	41
Self Tx	3	51
Button Mutual	3	14
Button Self	6	37

Figure 15 Data window example for attenuator ratio and the balancing current

In addition, the calibration also performs the gain correction and attenuator trimming for all enabled RX channels minimizing channel-to-channel mismatch and attenuator offset. The 8-bit gain correction is specifically designed to minimize the I_{BIAS} balance current mismatch between RX channels. In CYAT6165/CYAT8165 (TSG6L) gain correction is 16-bit. Each channel is evaluated with an internal touch emulator for both mutual-cap and self-cap sensing.

Both the sink and the source I_{BIAS} current are digitally adjusted to minimize the variation to within ±1.6% for mutual-cap and ±3% for self-cap. Each attenuator supports an 8-bit trimming to improve the attenuation ratio precision. The attenuation ratio can be affected by the offset current, which is caused by current mirror device mismatch and silicon process variation. The process includes minimizing mismatch between the primary and replica current sources as well as the mismatch between the sourcing and sinking current imbalance. Figure 16 shows an example from the TTHE IDAC Data window for the gain correction and attenuator trim.

Gain Correction		Rx0	Rx1	Rx2	Rx3	Rx4	Rx5	Rx6	Rx7	Rx8	Rx9	Rx10	Rx11	Rx12	Rx13	Rx14	Rx15	Rx16	Rx17	Rx18
Mutual	(+ve phase)	174	77	33	147	135	149	142	191	199	118	155	168	107	69	0	185	215	79	231
	(-ve phase)	61	59	62	59	63	60	62	58	62	58	62	59	62	59	62	59	61	60	62
Self	(+ve phase)	184	33	222	171	102	245	43	97	167	76	120	63	97	171	43	153	176	161	254
	(-ve phase)	61	61	61	60	63	61	62	60	62	60	62	61	62	60	62	61	61	61	62

	RxCh0	RxCh1	RxCh2	RxCh3	RxCh4	RxCh5	RxCh6	RxCh7	RxCh8	RxCh9	RxCh10	RxCh11	RxCh12	RxCh13
Attenuator Trim	5	35	7	1	7	7	7	5	3	5	7	9	5	5

Figure 16 Data window example for gain correction and attenuator trim

2.5.3 Calibration target

Calibration is the process of determining the optimal baseline, attenuation ratio, and the balance current to meet the specified balancing targets. There are two balancing targets:

- Integration voltage - This is the voltage on the integration capacitor after each integration cycle. Calibration determines an optimal configuration such that this voltage is within the specified target voltage.
- Balancing time - The time it takes to “balance” the voltage across the integration capacitor back to the reference voltage (V_{REF}). Balancing time is expressed in percentage of one half of the TX period. Calibration will determine an optimal configuration such that balancing time is approximately the same as the specified target.

2.6 TX slew rate

The TX driver edge rate can be adjusted to place some control over noise emission (for EMC/EMI compliance) while trading off against the maximum sample rate. The **TSS: DRV_STRENGTH_MC** and **TSS: DRV_STRENGTH_SC** parameters enable current limiting in the TX driver, thus reducing the TX slew rate.

An example is shown in Figure 17. Lower values reduce the edge rate, and higher values increase the rate. Setting to the maximum value disables the slew rate control, thus yielding the highest slew rate possible.

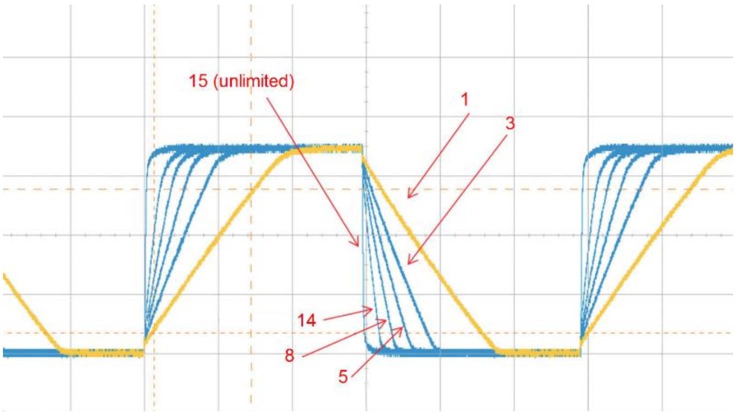


Figure 17 Typical TX waveforms at different values of TSS: DRV_STRENGTH_MC

The slew rate adjustment allows reduction of EMI/EMC emissions at the harmonics of the TX frequency. The lower edge rates (at smaller parameter values) can result in slower refresh rates due to longer sensor settling, so this adjustment allows a tradeoff between refresh rate and EMI/EMC emissions. In most cases, a parameter value of 14 will not affect TX panel settling, so disabling slew rate control is not recommended.

2.7 Estimating SNR procedure

SNR is used throughout the document to evaluate system performance. Use the following steps to estimate the SNR:

1. In Touch Reporting Display mode, select a sensor of interest. Make sure that “Show Sensors” in the Touch Pad Display Settings window is set to Mutual & Self (if self-cap is enabled).
2. Open the Sensor Monitor window by selecting View > Tool Windows > Sensor Monitor.
3. Center a grounded metal finger with a diameter at least twice the sensor pitch size on the selected sensor. Firmly hold the finger on panel. Finger movement during the test can artificially increase noise level.
4. Make sure the “Sensor Monitor Data Logging” icon is selected, by clicking on it if necessary. It is selected when it is surrounded by an outline. Otherwise, the Statistics Tool will not work.
5. Click on the “Clear All Chart” icon to clear all previous measurement history. Start Continuous Sampling.
6. Stop sampling after collecting at least 500 samples.
7. Open the Statistics Tool window. The minimum, maximum, and standard deviation data is shown on the right side of the window. Select SNR on the pull-down menu above the statistics data. It is recommended to tuning the mutual-cap and self-cap $SNR_{PK\text{-}PK}$ to at least 10.

2.8 What you will need

2.8.1 Touch Tuning Host Emulator software (TTHE)

The TTHE is a PC-based software tool designed to emulate a typical host. It talks to a touch device through a PC much like any application processor would in a phone or tablet. However, the TTHE contains many more features than a typical host would contain. In general, these features are present to enable advanced testing and tuning. The TTHE software must be installed before beginning tuning. For more information about the host emulator, a guide is available in the TTHE under the **Help** > **Documentation** menu. Note that the TTHE is also able to communicate with a system running Android over a Wi-Fi connection. This enables tuning of a complete system in its working environment.

Add easy-access buttons for calibration and re-baselining. To do this, right click in the toolbar of the TTHE and select Customize:

Click the Commands tab and drag and drop the  icon and the  icon to the TTHE toolbar. These icons are referred to throughout the document.

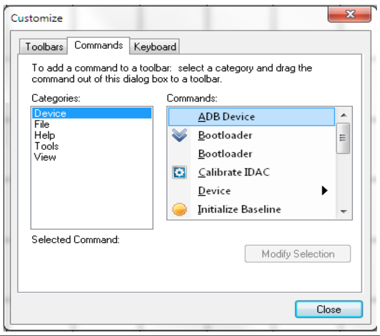


Figure 18 Customizing toolbar

2.8.2 Touch Tuning Bridge

If the system is in the beginning stages of tuning, a Touch Tuning Bridge (TTBridge) is typically used to allow communication between the touch device and PC hosting the TTHE. The TTBridge can use either I2C or SPI protocols.

2.8.3 Metal fingers

To assist in tuning, a selection of metal fingers of different sizes is required for tuning, typically brass or aluminum fingers. An example set of finger sizes is: 5 mm, 9 mm, and 22 mm.

Note that terminology relating to metal fingers is commonly used throughout this document. Here are the typical definitions:

- Grounded Metal Finger – A metal finger that is connected to the DUT ground
- Touched Metal Finger – A metal finger that is grounded through the touch of a human finger
- Finger Size – The size of the finger in contact with the touch panel. Typically, a finger is assumed to be an ideal cylinder with a known diameter. The cylinder's diameter is typically referred to as the finger size
- Glove - A metal finger with glued Delrin or PMMA, see Figure 20.

Figure 19 Metal fingers for tuning

"Glove" metal fingers may also be required to assist in tuning. A selection of "gloves" fingers use Delrin or PMMA of different thickness, typically glued to brass or aluminum fingers. Both Delrin and PMMA have a dielectric constant of about 4; glass is typically about 4.7. Typical thick and thin gloves are simulated by Delrin/PMMA of 1.0 mm and 4.0 mm respectively (see below).

Figure 20 Thin and thick gloves for tuning

3 Setup before HW available

3.1 Create the TTHE project

If creating a new project from an existing project, ensure all parameters that are not set at their defaults are checked to ensure they apply to the new project.

1. Launch the TTHE by the Windows Start button and search for “Host Emulator”.
2. For a new project, click **File** > **New Project**, which opens the Project Configuration Wizard, as shown in Figure 21.
3. Select the required part number and set project Name / Location.

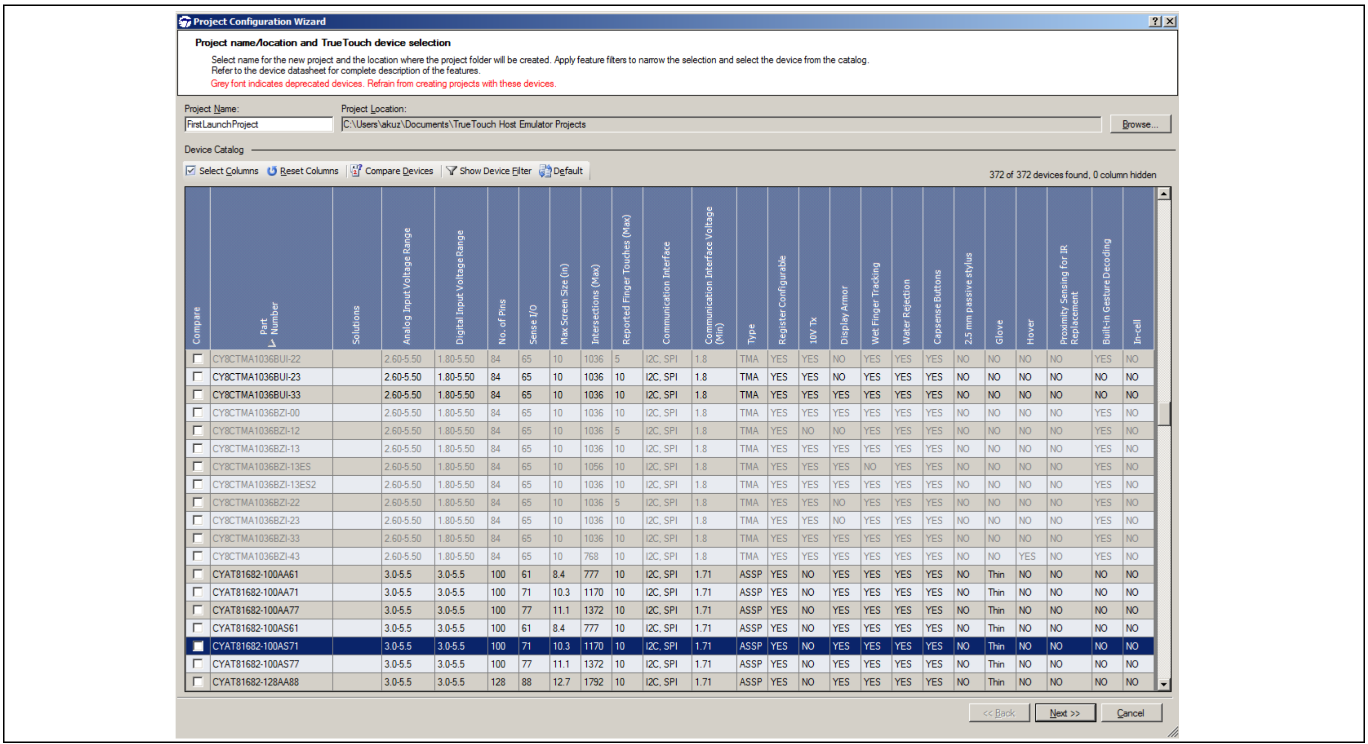


Figure 21 Screenshot of project configuration wizard – device selection

4. Click **Next** in the MPN selection screen, which brings up the static parameter selection screen, see Figure 22.
5. At this stage, the project can be saved by (optional):

- Click **Next** to go to the pin configuration wizard (see Figure 23)
- Click **Assign Dummy Sensors**, and then click **OK**
- The project can now be saved (**File** > **Save**) and the TTHE exited (**File** > **Exit**)

3.2 TTHE project static parameters and pinout

The static TTHE parameters cannot be changed by a bootload. The only way to change a static parameter is by programming via SWD. The static parameters can be seen in Figure 22 (**View** > **Project Configuration Wizard...**).

If SPI is selected, the clock phase and clock polarity can be set. If I2C is selected, the 7-bit slave address can be set.

If any more information is required on these parameters, refer to the device TRM.

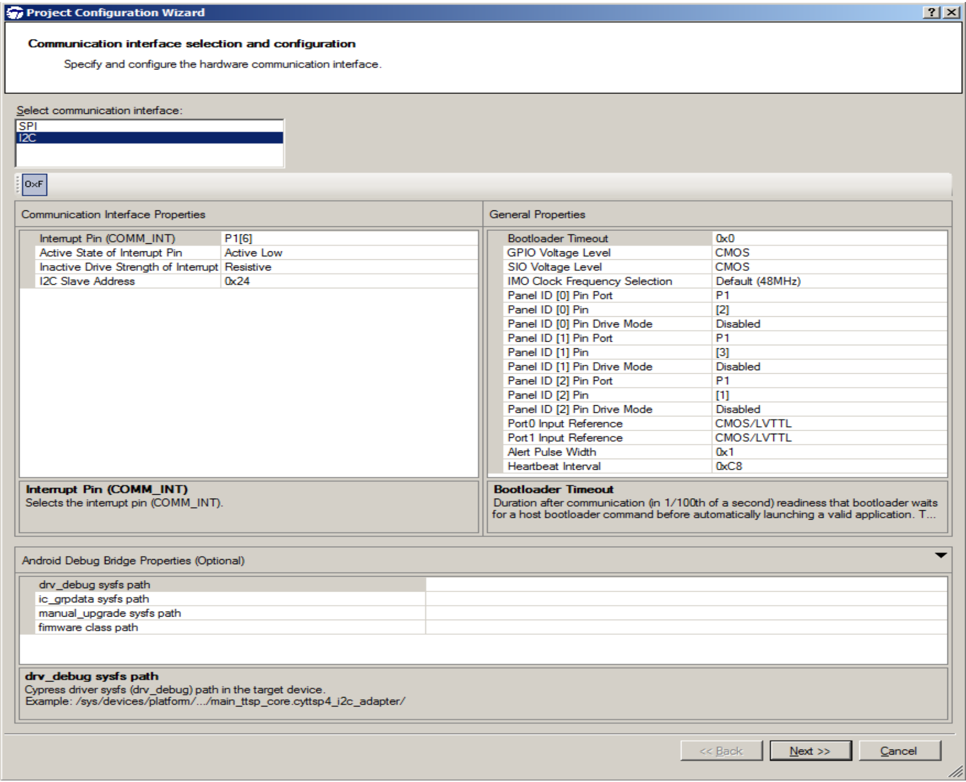


Figure 22 Screenshot of project configuration wizard – static parameters

To access the pinout, click **Next** on the project configuration wizard's static parameter screen. An example of the project configuration wizard's pinout screen can be seen Figure 23. in On this screen, set the following values (which should already be known):

- Number of X and Y sensors
- X and Y resolution (if unknown, refer to Section 3.3.5)
- VDDA level
- Number of buttons and button mode (scan type) and layout type (common-TX or common-RX)

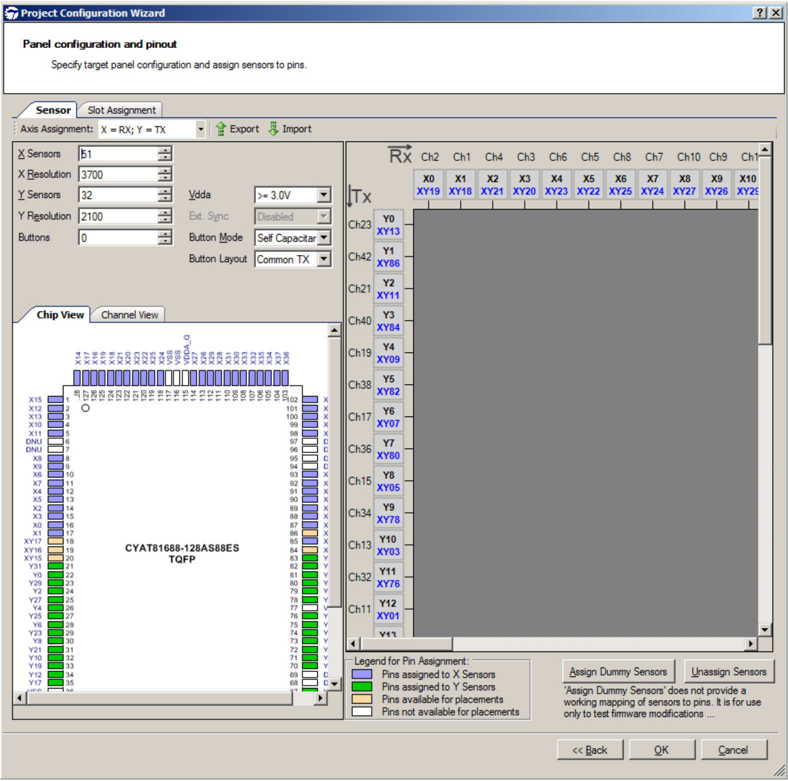


Figure 23 Screenshot of project configuration wizard – pinout

The pin assignment is critical to optimal sensing performance. Because there are parallel sensing channels, it is important to choose your pins and channels in such a way that your design operates optimally:

- Determine the TX and RX axes for scan time optimization:
  - Either the X or Y sensor electrode group can be TX or RX
  - Normally the longest axis is RX (#RX > #TX)
- Assign the RX sensors to slots:
  - If number of sensors is less than the number of RX-channels, there should only be 1 slot
  - Otherwise, the number of slots should be equal to Roundup(#RX-sensors / #RX-Channels)
  - Slots are initially assigned automatically in the Project Configuration Wizard
  - Slots can be configured manually using the “Slot Assignment” tab
- The RX sensors should be distributed as evenly as possible between the slots:
  - #RX-sensors per slot ~= #RX-sensors / #slots
  - This keeps the peak supply current at a minimum during scanning
- The maximum number of sense pins supported is:
  - CYAT8168 (TSG6XL) - 88 sense pins (XY matrix), with 54 RX channels
  - CYAT8165 (TSG6L) - 48 sense pins, with 17 RX channels
  - CYAT6165 (TSG6L Slider) - 48 sense pins, with 17 RX channels
- XY sensor mapping is limited:
  - Each sense pin is assigned to a specific RX channel
  - A slot cannot contain multiple pins assigned to the same RX channel
  - An example of the XY pins mapping to the RX channels using the TSG6XL (as shown in Table 7):

  - RX channel 0 can never be used twice as it is only mapped onto pin XY17
  - A single slot design cannot use both XY00 and XY27 as RXs because they both use channel 10
  - To use all channels in consecutive pins, an example is using pins XY17 to XY70 (channels 0-17: XY17-XY34, channels 18-35: XY35-XY52, and channels 36-53: XY53-70)
- A pin can be assigned to any slot where its RX-channel is not already used
- The main rule is that only one channel instance can be used in a slot

Table 7 Correspondence between RX channels and available XY sensors

RX Channel #	Sensors available for the channel					RX Channel #	Sensors available for the channel		RX Channel #	Sensors available for the channel	
	CYAT8168 (TSG6XL)		CYAT8165/6165 (TSG6L)				CYAT8168 (TSG6XL)			CYAT8168 (TSG6XL)	
0	-	XY17	-	XY06	XY15	18	XY08	XY35	36	XY53	XY80
1	-	XY18	-	XY07	XY16	19	XY09	XY36	37	XY54	XY81
2	-	XY19	-	XY08	XY17	20	XY10	XY37	38	XY55	XY82
3	-	XY20	XY00	XY09	XY18	21	XY11	XY38	39	XY56	XY83
4	-	XY21	XY01	XY10	XY19	22	XY12	XY39	40	XY57	XY84
5	-	XY22	XY02	XY11	XY20	23	XY13	XY40	41	XY58	XY85
6	-	XY23	XY03	XY12	XY21	24	XY14	XY41	42	XY59	XY86
7	-	XY24	XY04	XY13	XY22	25	XY15	XY42	43	XY60	XY87
8	-	XY25	XY05	XY14	XY23	26	XY16	XY43	44	XY61	-
9	-	XY26	XY24	XY32	XY40	27	XY71	XY44	45	XY62	-
10	XY00	XY27	XY25	XY33	XY41	28	XY72	XY45	46	XY63	-
11	XY01	XY28	XY26	XY34	XY42	29	XY73	XY46	47	XY64	-
12	XY02	XY29	XY27	XY35	XY43	30	XY74	XY47	48	XY65	-
13	XY03	XY30	XY28	XY36	XY44	31	XY75	XY48	49	XY66	-
14	XY04	XY31	XY29	XY37	XY45	32	XY76	XY49	50	XY67	-
15	XY05	XY32	XY30	XY38	XY46	33	XY77	XY50	51	XY68	-
16	XY06	XY33	XY31	XY39	XY47	34	XY78	XY51	52	XY69	-
17	XY07	XY34	-	-	-	35	XY79	XY52	53	XY70	-

If spare pins are available, they should be chosen such that they are between RX and TX sensors. Doing this will minimize the parasitic mutual-cap between RX and TX.

Check the “Slot Assignment” tab to ensure the expected number of slots exist. With the pinout satisfactorily configured, select “OK”, and save the project.

3.3 Project-specific constant parameter setting

3.3.1 Initial setup

When creating a new project from an existing project, double check that all the non-default settings are reviewed. The default values in a new project are defined to give the highest probability of hardware working immediately.

The following parameters should be known before hardware is available, and should be set in the initial (blind-build) release:

- **I2C Edge Rate Level**
  - FAST if current leakage from the I2C lines is more important (default)
  - SLOW if better I2C standard compliance is more important
- **TSS: SCANNING_MODE_BUTTON**
- **TSS: VDDA_MODE**
- **Device Setup: X_LENGTH_100xMM** and **Device Setup: Y_LENGTH_100xMM**
- **Device Setup: X_PITCH_10xMM** and **Device Setup: Y_PITCH_10xMM**
- **Device Setup: LOW_POWER_ENABLE**, **Device Setup: LP_INTRVL0**, and **Device Setup: TCH_TMOUT0**
- **Device Setup: ACT_INTRVL0** and **Device Setup: ACT_LFT_INTRVL0**
- **Device Setup: CONFIG_VER**
- **Device Setup: OPMODE_CFG: Handshake Configuration**
- **Device Setup: OPMODE_CFG: Deep Sleep Wakeup Trigger**
- **Device Setup: XY_AXES_FLIP_ROTATE_CFG: Flip X axis enable**, **Device Setup: XY_AXES_FLIP_ROTATE_CFG: Flip Y axis enable**, and **Device Setup: XY_AXES_FLIP_ROTATE_CFG: Rotate XY axes enable**

Disable advanced features for initial bring-up:

- **Device Setup: POST_CFG** = “No Response”
- **Scan Filtering: WATER_REJ_ENABLE**
- **Wet Finger: WF_ENABLE**
- **Raw Processing: PALM_STARTUP_NUM_OF_SENSORS** > #TX * #RX
- **SD Effect: SD_CORR_ENABLE** (**SD Correction: SD_CORR_ENABLE** in CYAT6165X/8165X (TSG6L))
- **Calibration: DYNAMIC_CALIBRATION_ENABLE**
- **Fingers: ACT_LFT_EN**
- **Touch Mode: TOUCHMODE_CONFIG** = “FingerOnly”

Additional considerations for a slider project:

- **TSS: MTX_ORDER** = 1
- **Device Setup: SENSOR_ASSIGNMENT** = RX=X; TX=Y
- **Device Setup: SELF_Z_MODE** = Self Cap RX
- **Fingers: FINGER_Z8_FILT_SCALE** = 0

3.3.2 Communication settings

Most of the communication settings, for example I2C and SPI, communication address, interrupt pin configuration, etc., are set in the TTHE by selecting **View** > **Project Configuration Wizard…** The meanings of these items are described in the TTHE User Guide and the device Technical Reference Manual. Since these items are not tuning parameters, they are not covered in this document.

The **Device Setup: INTERRUPT_PULSE_WIDTH** parameter controls the width (in µs) of the interrupt pulse generated by the touch controller. The proper setting depends on the host controller's capabilities. Set to the lowest value guaranteed by the project's host controller.

3.3.3 Interrupt pin

Ensure that the system design for the interrupt pin is well understood. If the internal pull-up/down is used for the interrupt signal and an external pull-up/down is not, ensure that the host can recognize/discard all interrupts caused by transients during device resets. Note that is it recommended to use an external pull-up/down on the touch interrupt signal.

3.3.4 Touches and large objects reported

Large objects are only processed if the parameter **Device Setup: LRG_OBJ_CFG** is set to "Fat Fingers & Large Objects Detected". If large object reporting is not wanted, set **Device Setup: LRG_OBJ_CFG** to "Fat Fingers Detected". It is not recommended to set **Device Setup: LRG_OBJ_CFG** to "Regular Fingers Detected Only" for normal applications.

Large object reporting can be configured to be either simple or complex:

- Simple Large Objects (**Device Setup: MAX_REPORTED_LO_NUM** = 0):
  - A large object is reported only by a flag to indicate if a large object is currently detected
  - No other information about the large object is available
- Complex Large Objects (**Device Setup: MAX_REPORTED_LO_NUM** > 0):
  - Each complex large object (up to 4) is processed and reported as a touch object
  - Each touch object contains size and center of mass information

The maximum number of reported touches and complex large objects is usually pre-defined, set these values in **Device Setup: MAX_REPORTED_TOUCH_NUM** and **Device Setup: MAX_REPORTED_LO_NUM**. The maximum complex large objects that can be reported is 4. The maximum total objects that can be reported is 10.

The touch controller will track up to 11 touch objects and assign a unique touch ID to each one of the detected objects. The objects are reported with the following priority:

1. #Reported large objects (LOs): MIN(#detectedLOs, **Device Setup: MAX_REPORTED_LO_NUM**)
2. #ReportedFingers: MIN(#detectedFingers, **Device Setup: MAX_REPORTED_TOUCH_NUM** - #reportedLOs)

There are different reporting orders of the two touch object types. While up to 11 objects maintain their touch IDs, the reported objects are governed by the following rules.

Complex large objects (when **Device Setup: MAX_REPORTED_LO_NUM** > 0) are prioritized in size order. If there are more than 4 large objects, the smallest object will lose its touch ID. Further, only the largest **Device Setup: MAX_REPORTED_LO_NUM** large objects will be reported.

Normal fingers are prioritized in age order (longest time of continuous detection). If there are more than 11 touch objects detect, the newest touch will lose its touch ID. Further, if there are more touches than slots available to report (see #ReportedFingers), then the touches that have been continuously detected on the panel for the longest are reported.

3.3.5 Panel resolution

The panel resolution is normally defined at the project start, and not a setting to be tuned. The following information is therefore included for reference only. However, if the panel resolution is not pre-defined, this section can be used to propose a resolution configuration.

The X and Y touch positions are reported in number of pixel counts. The parameters **Device Setup: X_RESOLUTION** and **Device Setup: Y_RESOLUTION** specify the x-direction and y-direction panel length in number of pixels respectively. Their value should be equal to the panel length in mm divided by the pixel size of the corresponding axis. The pixel size is expressed in units of mm.

$$\text{X\_RESOLUTION} = \frac{\text{panel length in X}}{\text{pixel size in X}}$$

Equation 6

$$\text{Y\_RESOLUTION} = \frac{\text{panel length in Y}}{\text{pixel size in Y}}$$

Equation 7

The ratio between the X and Y pixel size must be less than 1.25 for proper finger orientation reporting. Violating this recommendation can cause a round finger to be reported as an oval finger.

$$\frac{\max(\text{pixel size})}{\min(\text{pixel size})} < 1.25$$

Equation 8

3.3.6 Multi-phase TX order

This technique is not applicable for slider projects, where **TSS: MTX_ORDER** should be set to 1.

3.3.6.1 Background

Multi-phase TX (MTX) is a signal processing technique that increases SNR without increasing the scan time. The technique scans multiple sensors simultaneously. For example, if the parameter **TSS: MTX_ORDER** is set to 4, four TX electrodes are driven simultaneously with different phases (i.e. Multi-Phase TX). This method has the effect of integrating more data without increasing scan time. Thus, the SNR is improved. However, this is not without some side effects.

One effect of using a high MTX order is an increase in the TX pump capacitive loading (or VDDA loading), thus slowing down TX edges and requiring longer panel settling time. This could potentially result in fewer TX pulses, lowering the SNR.

Another effect of using a high MTX order is the decrease in sensitivity to dynamic objects. This is because complete sampling of any given sensor is distributed in time. Thus, a higher MTX order will degrade dynamic performance because the sampled object may move before the sampling is complete. For example, with a high

MTX order (>10) and a fast-moving finger (>2 m/s) sliding across a large touch screen, can result in the false detection of more than one finger. Because of this effect, it is recommended to keep the MTX order:

- ≤17 for a solution expecting moving object velocities up to 1 m/s
- ≤9 for a solution expecting moving object velocities up to 2 m/s

Fast moving objects will have their energy distributed across the multiple TXs (the same process that removes the noise). The effect is larger the bigger the ratio between the speed of the finger movement and the panel scan rate. The effect is due to the same TX electrode being re-scanned MTX_ORDER times. In the extreme case where a touch is only on a TX electrode for a single TX scan, then that sensor will show a value scaled by 1/MTX_ORDER.

The effect is three-fold. Firstly, ghost signal is produced by the deconvolution in locations it should not be. Secondly, the "true" signal is reduced by the displaced signal. Thirdly, the location of the ghost signal can be up to (MTX_ORDER-1) sensors from the true location. In summary, the following effects can be seen when moving a touch at about 1 m/s (panel refresh rate 100 Hz):

- MTX_ORDER of 7 or 9: Displaced signal is about 20% of that of a stationary touch
- MTX_ORDER of 11 or 13: Displaced signal is about 33% of that of a stationary touch

For example, if the signal of a stationary touch is 2000 counts and the finger threshold is 600 counts, an MTX_ORDER of 11 or 13 can produce false touches (ghost signal up to about 667 counts). The effects are most easily seen when moving a touch quickly along the edge of an MTX pattern (i.e. TX0), with the TTHE in Heat-map mode, showing diff-counts, and max-hold enabled.

3.3.6.2 Tuning

For initial setup, it is suggested to set **TSS: MTX_ORDER** to an odd value (even values result in significant radiated noise) between 5 and 9 inclusive, while maximizing MOD((TX_NUM-1), MTX_ORDER). The **TSS: MTX_ORDER** selection must also satisfy the following requirements:

- **TSS: MTX_ORDER** cannot exceed the number of TX-sensors
- The TX-patterns (TX-sensors rounded-up to MTX order, see Equation 9) is chip dependent:
  - TSG6L must not exceed 34
  - TSG6XL must not exceed 76
- The product of TX-patterns and RX-sensors is chip dependent:
  - TSG6L must not exceed 680 (maximum sensor crossings)
  - TSG6XL must not exceed 2304 (maximum sensor crossings)
- The TX sensors are divided into multiple groups (i.e. TX-patterns) of size **TSS: MTX_ORDER**
- The TX-pattern is the signal phase pattern for driving each TX group
- The total number of TX-patterns can be calculated using the following equation

$$\text{TX-patterns} = \text{RoundDown}\left(\frac{\text{Number of TX}-1}{\text{TSS: MTX\_ORDER}} + 1\right) \cdot \text{TSS: MTX\_ORDER}$$

Equation 9

3.4 Boundary clipping

The touch panel active area is usually larger than the display active area. In this case, a touch can be reported even if the touch is located outside of the display active area. Boundary clipping suppresses touch object reporting when the touch is located outside the specified touchscreen boundaries. The reported coordinate is linearly scaled within the boundaries, as shown in Figure 24.

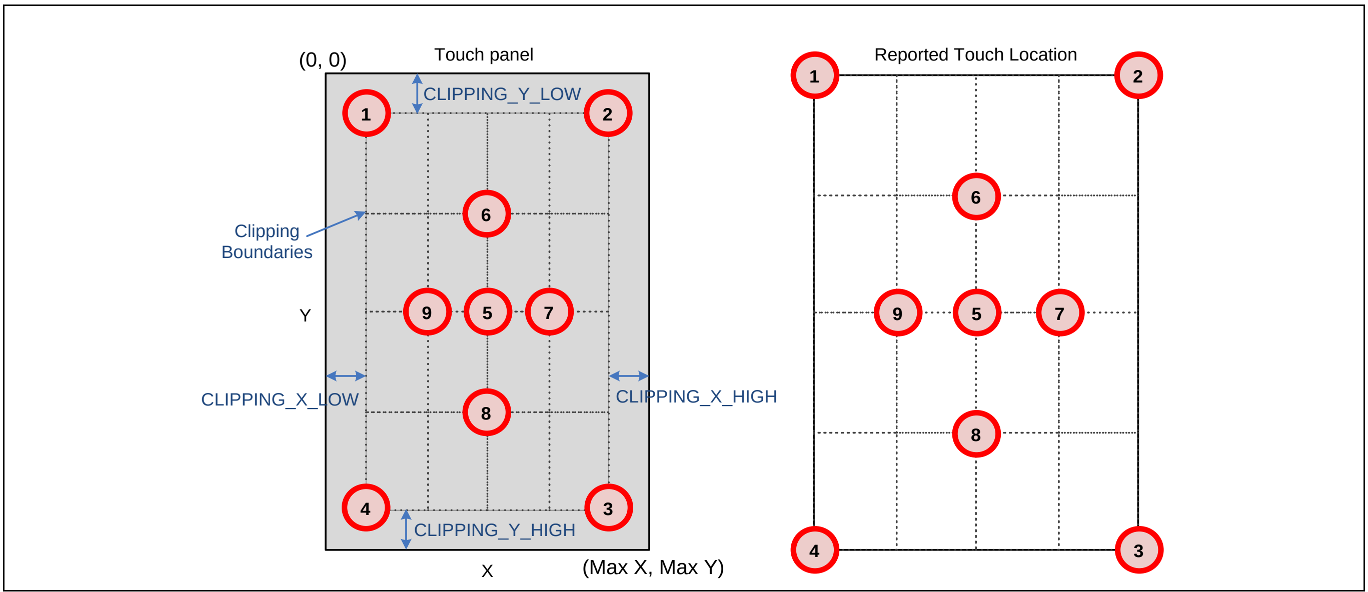


Figure 24 Boundary clipping illustration

To calculate the clipping size, divide the physical clipping size required by the pixel size. Use the following equation to set the clipping boundaries (assuming the display is centered, otherwise replace the 2 with the normalized display offset ratio):

$$\text{Clipping\_value} = \text{Round}\left[\left(\frac{\text{Panel\_size} - \text{Display\_size}}{\text{Panel\_size / Resolution}}\right) / \ 2\right]$$

Equation 10

As an example, see the following table. The X-axis data in the table is entered into Equation 10. The numerator is 1.5 mm (58.2 – 56.7). The denominator is 0.05 mm (58.2 / 1164). This works out to 15 ((1.5 / 0.05) / 2) pixels.

Table 8 Boundary clipping tuning example

Description	X	Y	Unit
Display resolution	1,164	2,044	pixels
Panel active area	58.2	102.2	mm
Display active area	56.7	100.8	mm
Delta	1.5	1.4	mm
Clipping boundary on each side	0.75	0.7	mm
	15	14	pixels
	Device Setup: CLIPPING_X_LOW = 15	**Device Setup: CLIPPING_Y_LOW** = 14	pixels

Description	X	Y	Unit
Tuning parameters	**Device Setup: CLIPPING_X_HIGH** = 15	**Device Setup: CLIPPING_Y_LOW** = 14	pixels

3.5 Parameters

Table 9 Boundary clipping tuning example

Configurable parameter	Description	Selection
TSS: SCANNING_MODE_BUTTON	Select button scan type: Self, Mutual or Hybrid = Self+Mutual (Note: Hybrid = Mutual only if Water rejection is Disabled).	Self Capacitance / Mutual Capacitance / Hybrid (default = Self Capacitance)
TSS: VDDA_MODE	Whether the TX drive voltage is sourced from the charge-pump or directly from the VDDA rail.	Pump Mode / Bypass Mode (default = Pump Mode)
TSS: MTX_ORDER	This parameter defines the multi-phase TX order.	1 – 19 (default = 4)
Device Setup: SENSOR_ASSIGNMENT	Assignment of TX/RX sensors to X/Y axes.	RX=X; TX=Y / RX=Y; TX=X (default = RX=X; TX=Y)
Device Setup: SELF_Z_MODE	Determines a set of sensors which are used for self-cap scan.	Self Cap RX / Self Cap TX / Self Cap TX & RX (default = Self Cap RX)
Device Setup: CLIPPING_X_LOW	Parameter defines the clipping region boundary at the low X side in pixel. The X coordinate is reported as 0 when a touch is at this boundary. The reported X coordinate is linearly scaled when a touch within the high X and low X boundary. A touch between this boundary and the neighboring panel edge is reported at panel edge.	-128 – 127 (default = 0)
Device Setup: CLIPPING_X_HIGH	Parameter defines the clipping region boundary at the high X side in pixel. The reported X coordinate is the maximum panel X resolution when a touch is at this boundary. The reported X coordinate is linearly scaled when a touch within the high X and low X boundary. A touch between this boundary and the neighboring panel edge is reported at panel edge.	-128 – 127 (default = 0)
Device Setup: CLIPPING_Y_LOW	Parameter defines the clipping region boundary at the low Y side in pixel. The Y coordinate is reported as 0 when a touch is	-128 – 127 (default = 0)

Configurable parameter	Description	Selection
	at this boundary. The reported Y coordinate is linearly scaled when a touch within the high Y and low Y boundary. A touch between this boundary and the neighboring panel edge is reported at panel edge.	
Device Setup: CLIPPING_Y_HIGH	Parameter defines the clipping region boundary at the high Y side in pixel. The reported Y coordinate is the maximum panel Y resolution when a touch is at this boundary. The reported Y coordinate is linearly scaled when a touch within the high Y and low Y boundary. A touch between this boundary and the neighboring panel edge is reported at panel edge.	-128 – 127 (default = 0)
Device Setup: X_RESOLUTION	X (horizontal) axis length in pixels. This value equals to the horizontal panel length divided by the x-direction pixel size.	0 – 32767 (default = 3700)
Device Setup: Y_RESOLUTION	Y (vertical) axis length in pixels. This value equals to the vertical panel length divided by the y-direction pixel size.	0 – 32767 (default = 2100)
Device Setup: X_LENGTH_100xMM	X (horizontal) axis physical length (in 1/100 mm).	0 – 65535 (default = 21700)
Device Setup: Y_LENGTH_100xMM	Y (vertical) axis physical length (in 1/100 mm).	0 – 65535 (default = 13600)
Device Setup: X_PITCH_10xMM	X (horizontal) axis pitch size (in 1/10 mm).	0 – 255 (default = 43)
Device Setup: Y_PITCH_10xMM	Y (vertical) axis pitch size (in 1/10 mm).	0 – 255 (default = 43)
Device Setup: MAX_REPORTED_LO_NUM	Maximum number of reported Large Object touches. Set to 0 disables Large Object report.	0 – 4 (default = 0)
Device Setup: MAX_REPORTED_TOUCH_NUM	Maximum number of touches can be reported to the host simultaneously.	1 – 10 (default = 10)
Device Setup: LOW_POWER_ENABLE	Enable/disable low power processing.	Enabled / Disabled (default = Disabled)
Device Setup: ACT_INTRVL0	Interval time before scan in Active mode (in ms).	0 – 250 (default = 9)
Device Setup: ACT_LFT_INTRVL0	Interval time before scan in Active-look-for-touch mode (in ms).	0 – 1000 (default = 16)
Device Setup: LP_INTRVL0	Interval time before scan in Low-power mode (in ms). Only used if low power processing is enabled.	0 – 1000 (default = 100)
Device Setup:	Initial touch timeout (in ms). TCH_TMOUT0 is the time that the device waits after the	0 – 60000

Configurable parameter	Description	Selection
TCH_TMOUT0	last touch liftoff event before transitioning from the active look-for-touch state to the low power state. Only if low power processing is enabled.	(default = 1000)
Device Setup: POST_CFG	Specifies actions the firmware shall take in response to a power-on self-test (POST) failure. Rarely changed from default value.	No Response / Disable Touch Sensing (default = No Response)
Device Setup: CONFIG_VER	Configuration Version (customer-defined). Stored in System Information registers CFG_VER. This can be used when the HOST is determining whether a bootload is necessary.	0 – 65535 (default = 0)
Device Setup: OPMODE_CFG: Handshake Configuration	Sets the HOST notification and handshaking mechanism for operating mode touch data transfer.	Asynchronous Pulse / Synchronous Pulse / Synchronous Level (default = Asynchronous Pulse)
Device Setup: OPMODE_CFG: Deep Sleep Wakeup Trigger	Specifies the trigger source for a wakeup from deep sleep.	Interrupt Pin Only / Interrupt Pin or Communication (default = Interrupt Pin Only)
Device Setup: INTERRUPT_PULSE_WIDTH	Minimum width of the interrupt pulse (in µs) generated by the touch controller.	3 – 255 (default = 100)
Device Setup: XY_AXES_FLIP_ROTATE_CFG: Flip X axis enable	Flip X axis (x = X_RES-1 - x) enable.	Enabled / Disabled (default = Disabled)
Device Setup: XY_AXES_FLIP_ROTATE_CFG: Flip Y axis enable	Flip Y axis (y = Y_RES-1 - y) enable.	Enabled / Disabled (default = Disabled)
Device Setup: XY_AXES_FLIP_ROTATE_CFG: Rotate XY axes enable	Rotate XY axes (x <-> y) enable.	Enabled / Disabled (default = Disabled)
Scan Filtering: WATER_REJ_ENABLE	Enables or disables the advanced water rejection feature; when enabled, self-cap scanning is performed on every panel scan.	Enabled / Disabled 6165X/8165X (TSG6L) (default = Disabled) 8168X (TSG6XL) (default = Enabled)

Configurable parameter	Description	Selection
Wet Finger: WF_ENABLE	Enables or disables the advanced wet finger processing feature; when enabled, additional processing is applied to the touch detection.	Enabled / Disabled (default = Enabled)
Raw Processing: PALM_STARTUP_NUM_OF_SENSORS	Number of mutual-cap sensors that have the negative diff data below (-finger threshold) to trigger a baseline reset. Applicable only when **Scan Filtering: WATER_REJ_ENABLE** is disabled. Set this parameter greater than total X*Y sensor number to disable Palm at Start Up detecting feature.	0 – 32767 (default = 32767)
SD Effect: SD_CORR_ENABLE CYAT8168X (TSG6XL) **SD Correction: SD_CORR_ENABLE** CYAT6165X/8165X (TSG6L)	Enables or disables SD correction.	Enabled / Disabled (default = Disabled)
Calibration: DYNAMIC_CALIBRATION_ENABLE	Enables or disables dynamic calibration.	Enabled / Disabled (default = Disabled)
Fingers: ACT_LFT_EN	Enables or disables self-cap scanning in look-for-touch mode. When enabled (and water rejection is disabled), self-cap scanning is only performed in LFT mode (and once every MAX_SELF_SCAN_INTERVAL).	Enabled / Disabled (default = Disabled)
Touch Mode: TOUCHMODE_CONFIG	Select whether gloves objects can be detected, or fingers only.	FingerOnly / FingerAndGlove (default = FingerAndGlove)

4 Parameters not covered in this tuning guide

There are many parameters available for configuration. Some of them are not needed for most projects. These are not covered by this tuning guide (see the following table).

Table 10 TTHE parameters not covered in this tuning guide

Configurable parameter	Description	Selection
TSS: SHORT_EDGE_CORRECTION	Short sensors diff. data amplification *(1+x/16). 0 - no amplification. Rarely changed from default value.	0 – 31 (default = 0)
TSS: TX_FREQ_METHOD_MC	This parameter sets the programmable integration option for the converter mutual-cap scans. See Section 15.1.4 for additional information about integration. Available options are: • Maximizing Integration Time. Integration time equals to the current TX period minus the integration reset time • Fixed Integration Time - ~100% integration. Limit the integration time based on the highest TX frequency of hop frequencies for the Charger Armor. The integration time equals to the minimum Mutual-cap TX period minus the reset time. • Limited Integration Time - ~66.7% integration. Integration time is limited to two-third of the current TX period. The reset time is ignored.	Maximizing Integration Time / Fixed Integration Time / Limited Integration Time (default = Maximizing Integration Time)
TSS: TX_FREQ_METHOD_SC	This parameter sets the programmable integration option for the converter self-cap scans. See Section 15.1.4 for additional information. Available options are: • Maximizing Integration Time - ~100% integration. Integration time equals to the current TX period minus the integration reset time • Limited Integration Time - ~66.7% integration. Integration time is limited to two-third of the current TX period. The reset time is ignored.	Maximizing Integration Time / Limited Integration Time (default = Maximizing Integration Time)
TSS: NM_WB_IDAC	Defines the IDAC value for the wide band noise metric.	12 – 127 (default = 12)

Configurable parameter	Description	Selection
TSS: RX_ATTEN_RES_BYPASS CYAT6165X/8165X (TSG6L) only	Dampening resistor bypass enable.	No Bypass / Bypass (default = No Bypass)
Device Setup: BR2_ALWAYS_ON_FLAG	Enables the BR2 algorithm to be used all the time. Rarely changed from default value.	Enabled / Disabled (default = Disabled)
Device Setup: PREFERRED_TOUCH_SIZE	Specifies preferred touch size in mm which is used as default touch size for centroid. Rarely changed from default value.	1 – 9 (default = 6)
Device Setup: LRG_OBJ_CFG	Defines when a large object will trigger a new touch report. Does not affect reports triggered by other events. Recommended not to change from the default. Rarely changed from default value.	Regular Fingers Detected Only / Fat Fingers Detected / Fat Fingers & Large Objects Detected
Device Setup: REPORT_CFG: Large Object Event	Defines when to trigger a new touch report while a large object is detected. Recommend leaving at the default. Rarely changed from default value.	Disabled / Change / Detection and Removal (default = Change)
Device Setup: SEND_REPORT_AFTER_ACTIVE_ INTERVAL_CFG	This parameter configures when touch report is sent to the HOST. Touch report can be sent right after touch processing is completed (default) or at the end of each active interval. The first option provides the fastest response time, but the second option provides a more consistent report interval. Rarely changed from default value.	Send touch report before active interval / Send touch report after active interval (default = Send touch report before active interval)
Device Setup: INTERRUPT_PIN_OVERRIDE: Beginning of Refresh Interval	Used for debugging purposes only. Force the interrupt pin to toggle at the beginning of Refresh Interval. Rarely changed from default value.	Enabled / Disabled (default = Disabled)
Device Setup: INTERRUPT_PIN_OVERRIDE: End of all Scanning/Processing	Used for debugging purposes only. Force the interrupt pin to toggle at the end of all scanning/processing. Rarely changed from default value.	Enabled / Disabled (default = Disabled)
Device Setup: INTERRUPT_PIN_OVERRIDE: Refresh Mode Change	Used for debugging purposes only. Force the interrupt pin to toggle at the refresh mode change. Rarely changed from default value.	Enabled / Disabled (default = Disabled)
Scan Filtering: XY_FILT_AXIS_IIR_COEF	Weight applied to the input of the IIR filter for touch ellipse major and minor axis lengths. Rarely changed from default value.	1, 1/2, 1/4, 1/8, 1/16 (default = One)

Configurable parameter	Description	Selection
Scan Filtering: **XY_FILT_AXIS_HYST**	Active distance for reporting axis changes. Rarely changed from default value.	0 – 255 (default = 0)
Scan Filtering: **XY_FILT_ANGLE_IIR_COEF** CYAT6165X/8165X (TSG6L) only	Note: CYAT6165X/8165X (TSG6L) only. Weight applied to the input of the IIR filter for touch ellipse angle with respect to the Y-axis. Rarely changed from default value.	1, 1/2, 1/4, 1/8, 1/16 (default = One)
Scan Filtering: XY_FILT_ANGLE_HYST CYAT6165X/8165X (TSG6L) only	Note: CYAT6165X/8165X (TSG6L) only. Active distance for reporting angle changes. Rarely changed from default value.	0 – 255 (default = 0)
SD Effect: **SD_CORR_MIN_COEFF** CYAT8168X (TSG6XL) **SD Correction:** **SD_CORR_MIN_COEFF** CYAT6165X/8165X (TSG6L)	Used to avoid overcompensation with a weak or floating touch.	0 – 65535 (default = 500)
SD Effect: **SD_CORR_MC_TARGET** CYAT8168X (TSG6XL) **SD Correction:** **SD_CORR_MC_TARGET** CYAT6165X/8165X (TSG6L)	Target MC signal in the non-SD state.	0 – 65535 (default = 1600)
SD Effect: **SD_CORR_SC_TARGET** CYAT8168X (TSG6XL) **SD Correction:** **SD_CORR_SC_TARGET** CYAT6165X/8165X (TSG6L)	Touch detection threshold in the SD state.	0 – 65535 (default = 150)
SD Effect: **SD_CORR_MC_SUM_THRESH** CYAT8168X (TSG6XL) **SD Correction:** **SD_CORR_MC_SUM_THRESH** CYAT6165X/8165X (TSG6L)	Threshold for touch validation in the SD state, applies to the 3x3 signal sum.	0 – 65535 (default = 800)
Fingers: **FINGER_Z8_FILT_SCALE**	Z8 filter scale for finger detection. A valid touch must satisfy this requirement: Z8 Sum > Finger Peak Diff-count * FINGER_Z8_FILT_SCALE. Rarely changed from default value.	0 – 127 (default = 0)
Fingers: **FINGER_POS_CALC_METHOD**	Centroid method. Rarely changed from default value.	BR2 / STANARD (default = STANDARD)

Configurable parameter	Description	Selection
Fingers: VP_DLT_RST_THRESH	The virtual peak (VP) algorithm is used to improve finger separation performance for small fingers. Rarely changed from default value.	0 – 65535 (default = 135)
Fingers: VP_DLT_THRESH	The virtual peak (VP) algorithm is used to improve finger separation performance for small fingers. Rarely changed from default value.	0 – 65535 (default = 2700)
Fingers: FAT_AXIS_LENGTH_THRESH	Touch Zone width threshold (unit: intersection cell) to determine whether a touch zone is fat or not; therefore, determine proper centroid algorithm to be used; default value: 5 (if 5x5 centroid is applied). Rarely changed from default value.	0 – 128 (default = 5)
Fingers: AXIS_ORIENTATION_ENABLE: Axis Length Enable CYAT8168X (TSG6XL) only	Note: CYAT8168X (TSG6XL) only. Enables the calculation and reporting of the length of the detected object's longest diameter.	Enabled / Disabled (default = Enabled)
Fingers: AXIS_ORIENTATION_ENABLE: Orientation Enable CYAT8168X (TSG6XL) only	Note: CYAT8168X (TSG6XL) only. Enables the calculation and reporting of the angle of the detected object's longest diameter.	Enabled / Disabled (default = Enabled)
Gloves: GLOVES_POS_CALC_METHOD	Position calculation algorithm selection register.	BR2 / STANDARD (default = STANDARD)
Gloves: GLOVES_MAX_ACCEPTABLE_ NOISE_LEVEL	Acceptable max noise level in glove mode.	0 - 255 (default = 0)
Touch Mode: TOUCHMODE_LFT_NEG_SC_THRESH	Self-cap negative LFT threshold (in counts). When self-cap signals are below this level in Look-for-Touch Mode it means that there is unsettled signal on the panel and application should proceed to an active mode to force baseline update. Rarely changed from default value.	-32768 – 0 (default = -32768)
Touch Mode: TOUCHMODE_LFT_NEG_MC_THRESH	Mutual-cap negative LFT threshold (in counts). When mutual-cap signals are below this level in Look-for-Touch Mode it means that there is unsettled signal on the panel and application should proceed to an active mode to force baseline update. Rarely changed from default value.	-32768 – 0 (default = -32768)

5 HW bring-up

The touch system should be placed on a flat surface that represents a typical environment for the system. Start tuning with no display or injected noise. Place the sensor on a non-conductive surface or a grounded metal plate. All components should be assembled. For example, the sensor must be connected to the touch controller (usually by a flexible printed circuit (FPC)). The TTBridge (Touch Tuning Bridge) must be connected to the touch controller to interface to the TTHE. If the system being tuned includes a display, ensure the display is off and grounded.

Before starting to tune the project, verify communication with the TTHE, program and calibrate the touch controller, and check TX/RX pin connection.

5.1 Initial TTHE connection

For the first connection to a new project, use the following procedure:

1. Launch the TTHE software.
2. Open the TTHE project (*.config) created in the Section 3.
3. Click the Connect button in the toolbar to get device connected to the TTHE

   The TTBridge detects if the DUT is self-powered (on either VAUX or VCOM).

   If the DUT is self-powered, the TTBridge disables power to both VAUX and VCOM.

   If the DUT is not powered, the TTBridge enables power output (VAUX and VCOM).
4. Program the device with the firmware and tuning configuration

   Skip this step if the device is already programmed.

   If the SWD connection is available, click Program to program.

   If the SWD connection is not available, click Bootloader to bootload.
5. Click Run to start communication

   If the device was programmed by SWD, a calibration will automatically be performed during the startup of the touch application.

   If the device was programmed by a bootload, click to calibrate the device.

5.2 TX/RX pin assignment check

The purpose of this check is to make sure that all the device I/O pin assignments match the panel I/O specifications. An incorrect assignment affects finger tracking, and may cause a single finger touch being reported as multiple fingers. Example heat map plots are shown in Figure 25. The same 22-mm round finger touch results in broken finger segments because of the incorrect I/O assignments.

Figure 25 shows three examples, two with incorrect pin assignments. The leftmost image shows the correct I/O assignments. The center image shows row 10 and row 11 swapped. The rightmost image shows rows 10 and 11 swapped with rows 18 and 19.

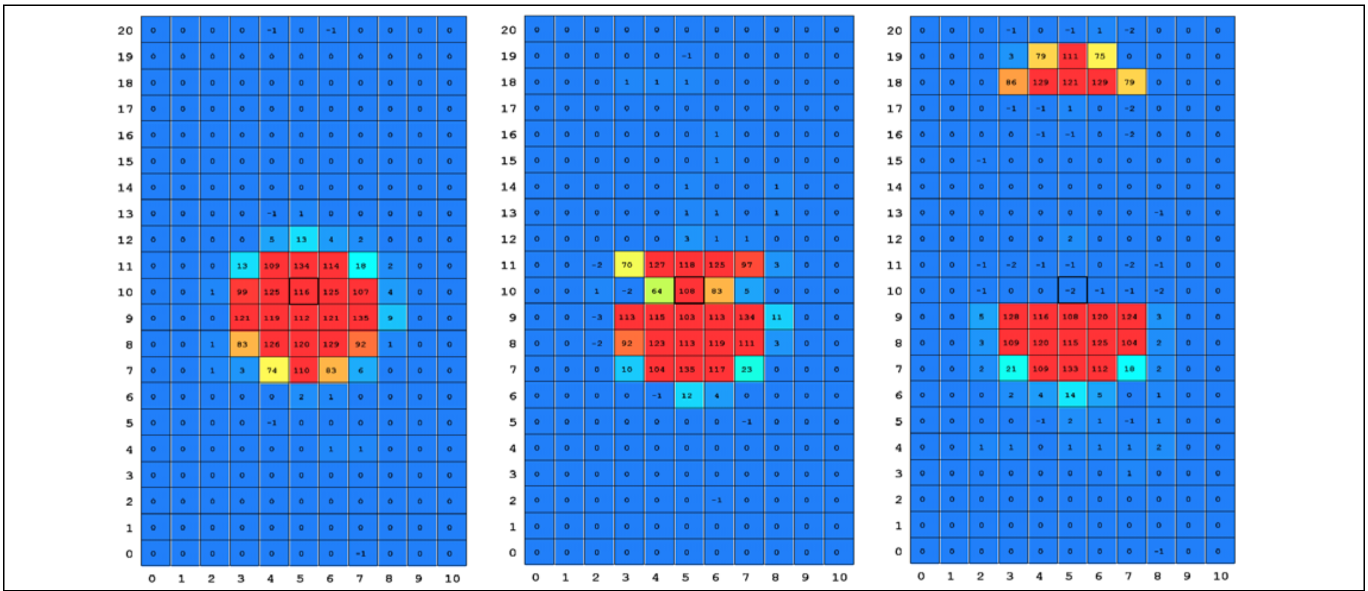
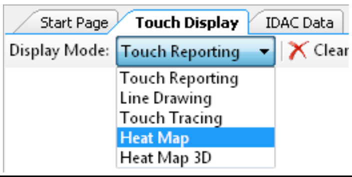
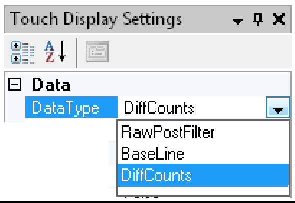


Figure 25 Correct and incorrect pin assignment examples

1. Go to the Heat Map display mode.



2. Set “DataType” to “DiffCounts” in the Touch Display Settings window (enabled from View -> Tool Windows).



3. Slowly run a large grounded metal finger vertically from the top to the bottom of the panel and look for inconsistent finger detection.
4. Repeat the same test running from the left side to the right side of the panel.
5. If inconsistent diff-counts are observed for a fat finger touch:

- Update the TX/RX pin configuration in View > Project Configuration Wizard...
- Refer to the Technical Reference Manual for more information about pin assignment.

5.3 Verify error signal

The touch controller supports reporting errors through either hardware or software. The hardware method requires GPIO P1[4] (P1[2] for CYAT6165X/8165X (TSG6L)) to be routed to the Touch Tuning Bridge (TT Bridge) RX_HOST pin. If the hardware method is not supported, configure TTHE to check the error register with a software command by selecting Tools -> Options -> Error Pin Configuration -> Error resgister only.

The functionality of the error signal can be verified by setting:

1. **MFG: POST_SHORT_OPEN_CTRL** = Enable All
2. **MFG: ILEAK_MAX** = 0

After applying these changes, the error signal should be asserted on every touch controller start-up. After confirming the error signal is correctly reported, revert the two settings to their original project values.

6 Analog front-end tuning

Tuning a solution starts at the point where data is acquired. There are numerous parameters that guide the acquisition of data, and fortunately many of the parameters are automatically determined, but not all. This section describes tuning the acquisition of data. The essential steps are listed and then detailed in the following sections.

- Overview – First, an overview of the whole analog front end is shown.
- Drive and reference voltages – Set the parameters that control the different voltages in the system.
- TX waveform – The most important (and most complex) step is to setup the TX waveform, as this must be configured to pass SNR and EMI/EMC emissions/immunity requirements.
- Calibration verification – The automatic calibration results should be checked to ensure the proper operation of the algorithms that control the integration and current balancing.
- TX scan – The parameters that control the sampling and conversion of data (such as sampling rate and integration) need to be setup for data acquisition.
- Signal scaling – Finally, scale the data generated for touch processing.

6.1 Overview

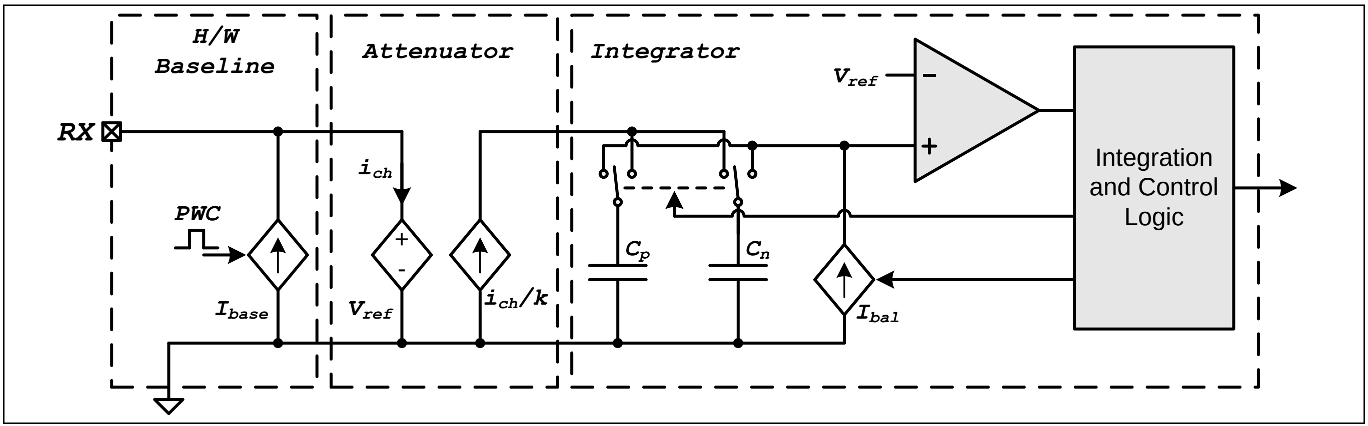


Figure 26 Analog to digital converter

The analog to digital converter contains several parameters that are worthy of note to understand the design; however, they are not directly adjustable, as they are automatically determined through the calibration process.

- Hardware baseline – Hardware baselining is the process of removing or compensating the nominal signal charge that is always present with or without any touch on the panel. Removing some charge (injecting offset) before conversion enables sensing over a wider range of sensor capacitance, thus improving SNR. The amplitude and modulation period of the baseline IDAC are automatically configured.
- Attenuator – The attenuator clones the current at its input source and provides an isolated and attenuated copy of that current inside the channel for integration. The attenuator is automatically configured to any one of the following options: 3x, 4x, 4.8x, 6x, 8x, 12x, and 24x.
- Integrator – The integrator is a pair of 10pF capacitors that are charged/discharged through the attenuator and discharged/charged via the balancing IDAC. The integration time, integration relationship to TX period, and IDAC settings are automatically configured.

6.1.1 Self-cap

If the project requires self-cap, enable it here. Self-cap can serve three purposes.

First, it can be used for a quick (hence low-power) look-for-touch (LFT) when no finger is on the panel. A self-cap scan is faster than mutual-cap, and can provide a faster response to new touches. The scan method to use for LFT is defined by the parameter **Fingers: ACT_LFT_EN**.

Second, when a finger is on the panel, self-cap data can be used along with the mutual-cap data to significantly improve water rejection performance. The usage of self-cap data for water rejection is controlled by the parameter **Scan Filtering: WATER_REJ_ENABLE**.

Third, hover is implemented using self-cap scanning. See the hover chapter for more information on hover set-up and tuning.

Table 11 summarizes the scanning as a function of the two parameter settings.

Table 11 Self-cap configuration

Look-for-touch Fingers: ACT_LFT_EN	Water rejection Scan Filtering: WATER_REJ_EN	Scan with no finger	Scan with finger
Disabled	Disabled	Mutual only	Mutual only
Disabled	Enabled	Mutual (self for baseline only)	Both
Enabled	Disabled	Self (mutual for baseline only)	Mutual (self for baseline only)
Enabled	Enabled	Self (mutual for baseline only)	Both

In Table 11, some states use one scan method, but the other scan method also occurs "for baseline only". Baselines track environmental changes. However, if a scan method is not being used, the baselines cannot be updated. Therefore, the inactive scan method forces periodic scans just to update the baselines. The maximum interval between scans is controlled by the two parameters **Device Setup: MAX_MUTUAL_SCAN_INTERVAL** and **Device Setup: MAX_SELF_SCAN_INTERVAL**. These parameters can normally be left at their default values.

6.1.2 Parameters

Table 12 Analog front-end overview parameters

Configurable parameter	Description	Selection
Device Setup: MAX_MUTUAL_SCAN_INTERVAL	Maximum interval between mutual-cap scans in ms. Only used when in the LFT state and **Fingers: ACT_LFT_EN** enabled.	0 – 65535 (default = 100)
Device Setup: MAX_SELF_SCAN_INTERVAL	Maximum interval between self-cap scans in ms. Only used when self-cap is enabled, but the current scan method is mutual-only.	0 – 65535 (default = 100)

Configurable parameter	Description	Selection
Fingers: ACT_LFT_EN	Selects the scan method used in look-for-touch mode. If enabled, self-cap scanning is used. If disabled, mutual-cap scanning is used.	Enabled / Disabled (default = Disabled)
Scan Filtering: WATER_REJ_ENABLE	Enables or disables the advanced water rejection feature. When enabled, both self-cap and mutual-cap scanning are performed on every active-state panel scan.	Enabled / Disabled 6165X/8165X (TSG6L) (default = Disabled) 8168X (TSG6XL) (default = Enabled)

6.2 Drive and reference voltage tuning

6.2.1 VDDA settling time

In most systems, the VDDA and VDDD rails settle at the same time. In this case, the **TSS: STARTUP_DELAY** parameter can be set to zero.

If the VDDA rails does not settle until after the VDDD rail. Measure the delta and set the **TSS: STARTUP_DELAY** parameter to the correct delay.

6.2.2 Reference voltage

The reference voltage is used in all the RX channels for precise charge measurement. For proper device operation, the voltage must be configured based on the VDDA level. The **TSS: REFGEN_CTL: REF_SCALE** parameter in Table 13 defines the proper settings based on the supplied VDDA. Note that EMI/EMC emissions requirements might require the **TSS: REFGEN_CTL: REF_SCALE** to be set lower than the ideal.

6.2.3 Mutual-cap TX driver voltage

The touch controller allows driving the mutual-cap TX signal at the voltage supplied by VDDA, or at a voltage driven by the internal charge pump. This allows a wide range of voltage programmability to achieve better SNR for the touch solutions that need it. The charge pump is programmable to supply 4.3 V to 5.0 V. Note that it is recommended to use the charge pump to generate 5 V (default) if VDDA is less than 4.7 V. If a well-regulated low-noise VDDA supply is available, the TX Pump can be disabled and an external supply connected to the VCCTX pin (through an external low-pass filter).

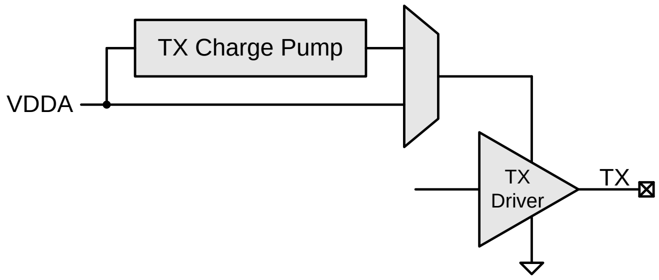


Figure 27 TX charge pump

The parameter **TSS: VDDA_MODE** controls whether the device uses the charge pump (“Pump Mode”) or passes VDDA through to the TX driver (“Bypass Mode”). Note that the hardware layout is different for a design with and without the charge pump enabled. Therefore, **TSS: VDDA_MODE** should not be changed, unless the hardware can support it.

The parameter **TSS: PUMP_DELAY_US** specifies the delay in µs from the device exiting sleep mode to the charge pump being enabled. This allows sufficient time for the charge pump regulation to settle. It is recommended leaving this at 1000 µs (default), unless a VCCTX capacitance is different from the datasheet recommendation.

6.2.4 Self-cap TX voltage

By driving both sides of the sensor, we can reduce (or ideally, eliminate) the charge that couples across the mutual-cap embedded in the sensor. If the potential on the left of the mutual-cap is the same as the potential on the right, then there will be no current flow through this capacitor. The sensor charge collected will be dominated by the self-cap of the sensor. The touch controller has an adjustable shield driver and reference to enable self-cap sensing. Figure 28 shows the basic configuration for self-cap TX drive.

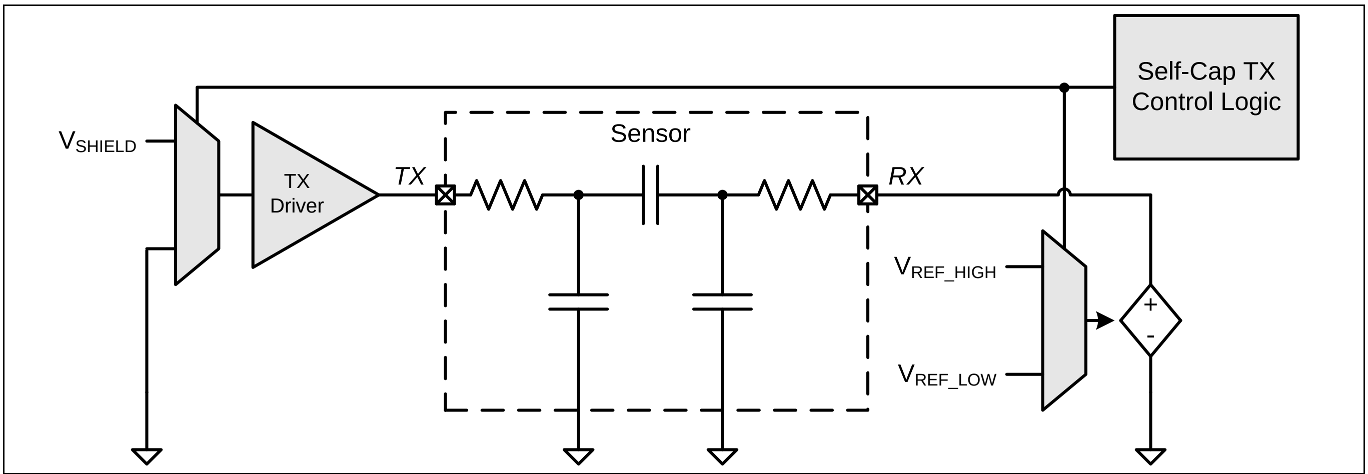


Figure 28 Self-cap drive voltages

The **TSS: REFGEN_CTL: RXDAC** parameter adjusts the “swing" of the sensing signal. The swing is between VREF_HIGH and VREF_LOW as shown in Figure 28. Table 13 describes these reference voltages for each combination of **TSS: REFGEN_CTL: RXDAC** and **TSS: REFGEN_CTL: REF_SCALE** settings. **TSS: REFGEN_CTL: REF_SCALE** is set based on the VDDA level. In general, the self-cap signal swing increases with higher value of **TSS: REFGEN_CTL: RXDAC**.

The settings in Table 13 give the best channel SNR for a given VDDA value. For systems with a VDDA greater than 4.8 V, a **TSS: REFGEN_CTL: REF_SCALE** of 5 should be used.

In some cases, it may be desirable to use lower voltage swings to reduce EMI/EMC emissions. In these cases, the **TSS: REFGEN_CTL: REF_SCALE** parameter can be set lower, or even to 0, and **TSS: REFGEN_CTL: RXDAC** parameter can be set to give the desired voltage swing. Note that the minimum VDDA supported by the touch controller is 3.0 V; therefore, **TSS: REFGEN_CTL: REF_SCALE** should only be set to 0 when needed for EMC-emissions.

Table 13 Self-cap reference voltage levels

		REF_SCALE																
		0			1			2			3			4			5		
		2.6V < VDDA < 2.9V			2.9V < VDDA < 3.2V			3.2V < VDDA < 3.6V			3.6V < VDDA < 4.0V			4.0V < VDDA < 4.5V			4.5V < VDDA < 4.8V		
		Low	Mid	High	Low	Mid	High	Low	Mid	High	Low	Mid	High	Low	Mid	High	Low	Mid	High
RXDAC	0	1.05	1.3	1.55	1.17	1.45	1.73	1.29	1.6	1.91	1.44	1.79	2.13	1.63	2.02	2.4	1.8	2.23	2.66
	1	1	1.3	1.6	1.12	1.45	1.79	1.23	1.6	1.97	1.37	1.79	2.2	1.55	2.02	2.48	1.72	2.23	2.75
	2	0.95	1.3	1.65	1.06	1.45	1.84	1.17	1.6	2.03	1.3	1.79	2.26	1.47	2.02	2.56	1.63	2.23	2.83
	3	0.9	1.3	1.7	1.01	1.45	1.9	1.11	1.6	2.09	1.24	1.79	2.33	1.39	2.02	2.63	1.54	2.23	2.92
	4	0.85	1.3	1.75	0.95	1.45	1.95	1.05	1.6	2.15	1.17	1.79	2.4	1.32	2.02	2.71	1.46	2.23	3
	5	0.8	1.3	1.8	0.89	1.45	2.01	0.99	1.6	2.22	1.1	1.79	2.47	1.24	2.02	2.79	1.37	2.23	3.09
	6	0.75	1.3	1.85	0.84	1.45	2.07	0.92	1.6	2.28	1.03	1.79	2.54	1.16	2.02	2.87	1.29	2.23	3.18
	7	0.7	1.3	1.9	0.78	1.45	2.12	0.86	1.6	2.34	0.96	1.79	2.61	1.08	2.02	2.94	1.2	2.23	3.26
	8	0.65	1.3	1.95	0.73	1.45	2.18	0.8	1.6	2.4	0.89	1.79	2.68	1.01	2.02	3.02	1.12	2.23	3.35
	9	0.6	1.3	2	0.67	1.45	2.23	0.74	1.6	2.46	0.82	1.79	2.74	0.93	2.02	3.1	1.03	2.23	3.43
	10	0.55	1.3	2.05	0.61	1.45	2.29	0.68	1.6	2.52	0.75	1.79	2.81	0.85	2.02	3.18	0.94	2.23	3.52
	11	0.5	1.3	2.1	0.56	1.45	2.35	0.62	1.6	2.59	0.69	1.79	2.88	0.77	2.02	3.25	0.86	2.23	3.6
	12	0.45	1.3	2.15	0.5	1.45	2.4	0.55	1.6	2.65	0.62	1.79	2.95	0.7	2.02	3.33	0.77	2.23	3.69
	13	0.4	1.3	2.2	0.45	1.45	2.46	0.49	1.6	2.71	0.55	1.79	3.02	0.62	2.02	3.41	0.69	2.23	3.78
	14	0.35	1.3	2.25	0.39	1.45	2.51	0.43	1.6	2.77	0.48	1.79	3.08	0.54	2.02	3.49	0.6	2.23	3.86
	15	0.3	1.3	2.3	0.34	1.45	2.57	0.37	1.6	2.83	0.41	1.79	3.16	0.47	2.02	3.56	0.52	2.23	3.95

As an example, a **TSS: REFGEN_CTL: REF_SCALE** setting of 0 and an **TSS: REFGEN_CTL: RXDAC** setting of 11 would yield a V_{REF_LOW} value of 0.5 V and a V_{REF_HIGH} value of 2.1 V. The signal swing equals 1.6 V.

The color coding in Table 13 indicates acceptable and unacceptable (shown in red) settings. The first rule is to select V_{REF_HIGH} to be at least 500 mV below VDDA. If this rule is not satisfied, the device may still work normally, but there is no guarantee it will work for all devices in a production run at all temperatures specified for the touch controller.

$$V_{REF\_HIGH} \leq VDDA - 500\ mV$$

Equation 11

The second rule is to limit the self-cap signal swing to 2 V or less.

$$V_{REF\_HIGH} - V_{REF\_LOW} \leq 2\ V$$

Equation 12

This rule is guided by V_{SHIELD}. The TX shield driver may fail to keep up with the signal swing if the signal swing is larger than 2 V. And, as already noted, it is desirable to match the voltage swing on both TX and RX.

$$V_{SHIELD} \approx V_{REF\_HIGH} - V_{REF\_LOW}$$

Equation 13

Finally, the TX driver is by design most efficient within the 1 V to 2 V range. Therefore, the last rule is:

$$1\ V \leq V_{SHIELD} \leq 2\ V$$

Equation 14

TSS: REFGEN_CTL: TX SHIELD is the configurable parameter for the TX shield driver voltage. The values shown in Table 14 are the voltages achievable for V_{SHIELD} from Figure 28 for a given setting. Note that this table has been color coded to indicate which settings are recommended or not recommended (red) by design.

For example, a **TSS: REFGEN_CTL: REF_SCALE** setting of 0 and a **TSS: REFGEN_CTL: TX SHIELD** setting of 11 would produce a V_{SHIELD} value of 1.6 V. Notice that this value matches the signal swing with RXDAC of 11. Thus, keep **TSS: REFGEN_CTL: TX SHIELD** = **TSS: REFGEN_CTL: RXDAC**.

Table 14 Self-cap TX shield voltage values

		REF_SCALE					
		0	**1**	**2**	**3**	**4**	**5**
TX SHIELD	**0**	0.50	0.56	0.62	0.69	0.78	0.86
	1	0.60	0.67	0.74	0.83	0.93	1.03
	2	0.70	0.78	0.87	0.96	1.09	1.20
	3	0.80	0.89	0.99	1.10	1.24	1.38
	4	0.90	1.01	1.11	1.24	1.40	1.55
	5	1.00	1.12	1.23	1.38	1.55	1.72
	6	1.10	1.23	1.36	1.51	1.71	1.89
	7	1.20	1.34	1.48	1.65	1.86	2.06
	8	1.30	1.45	1.60	1.79	2.02	2.24
	9	1.40	1.56	1.72	1.93	2.17	2.41
	10	1.50	1.67	1.85	2.06	2.33	2.58
	11	1.60	1.78	1.97	2.20	2.48	2.75
	12	1.70	1.90	2.09	2.34	2.64	2.92
	13	1.80	2.01	2.21	2.48	2.79	3.10
	14	1.90	2.12	2.34	2.61	2.95	3.27
	15	2.00	2.23	2.46	2.75	3.10	3.44

6.2.5 Parameters

Table 15 Drive and reference voltage parameters

Configurable parameter	Description	Selection
TSS: REFGEN_CTL: TX SHIELD	Output voltage for the TX shield.	0 – 15 (default = 0)
TSS: REFGEN_CTL: RXDAC	Output voltage for the RX DAC.	0 – 15 (default = 0)
TSS: REFGEN_CTL: REF_SCALE	Scaling for the reference voltage.	0 – 5 (default = 0)
TSS: STARTUP_DELAY	This parameter sets a delay to allow VDDA settling. The value is specified in ms.	0 – 255 (default = 100)
TSS: VDDA_MODE	Whether the TX drive voltage is sourced from the charge-pump or directly from the VDDA rail.	Pump Mode / Bypass Mode (default = Pump Mode)
TSS: PUMP_DELAY_US	The delay time to allow pumping VCCTX to the stable target voltage. The value is specified in µs.	0 – 10000 (default = 1000)

6.3 TX waveform tuning

TX waveform tuning is perhaps the most complex part of the tuning process. There are many requirements, frequently demanding opposite tuning actions, or actions that significantly reduce the achievable SNR.

The aim of this tuning is to provide a TX waveform that maximizes SNR and passes EMI/EMC emissions requirements. The major parameters to be tuned control the TX frequency, TX slew rate, and TX spreader. To a lesser extent, the TX pulses and refresh rate also impact display noise and EMI/EMC emissions performance.

In the following sections, the parameter tuning is explained. However, despite the document being linear, all these parameters change the TX waveform. A configuration must be found where the combined effects of all these parameters produce a waveform that passes EMI/EMC emissions and maintains enough SNR to meet the performance goals.

If the project requires display synchronization, then this should be configured first; see Section 8.3. Display synchronization may also cement some of the TX waveform parameters, which will simplify the TX waveform tuning in this section.

6.3.1 TX frequency overview

The TX frequency is the frequency of the waveform used to scan the panel. Separate frequencies can be used for self-cap and mutual-cap scans, and each should be tuned independently using the steps in this section.

The TX frequency is perhaps the most critical tuning parameter. It affects the EMI/EMC emissions, display noise immunity, touch SNR, and panel refresh rate. The choice of TX frequency is driven by the following constraints:

- Panel speed: The ITO traces on the panel have series resistance and shunt capacitance. As such, they form a low-pass filter to the TX signal. The TX frequency must be chosen that does not exceed the inherent speed of the panel, otherwise the TX waveform and scan results will be distorted. This puts an upper limit on the range of workable TX frequencies.
- EMI/EMC emissions: Each transmit electrode is driven at the TX frequency. Each electrode can be thought of as an antenna, and the electromagnetic emission from the system will be highest at the TX frequency and its odd harmonics. Even harmonics will also be present if a duty cycle other than 50% is selected (**TSS: TSS_TX_DUTY_CYCLE** tuning parameter). The TX frequency can often be set such that fundamental and some harmonics fall outside the band of interest for emission.
- Display noise immunity: Display noise depends strongly on frequency. If the display noise spectrum has appreciable energy at the TX frequency, or a harmonic, then the negative effects of display noise are significantly stronger.
- SNR: For the same amount of scan time, a higher TX frequency allows scanning for more TX pulses, improving SNR. For example, in the same scan time, a TX frequency of 300 kHz can have twice as many TX pulses as a TX frequency of 150 kHz. In addition, a higher TX frequency tends to result in better noise performance, because noise tends to be attenuated at higher frequencies.
- Panel refresh rate: Higher TX frequencies allow faster scanning of the panel.

The goal is to find a TX frequency that meets all the above constraints. If multiple TX frequencies can meet the constraints, then typically the highest frequency is the best choice. Note that the display stack-up affects the panel speed, so if a bare sensor (no display) is tuned, it may result in a TX frequency that is not acceptable after the display has been added to the setup.

The TX waveform is a divided copy of the 48 MHz IMO clock. The parameters **TSS: TX_PERIOD_MC** and **TSS: TX_PERIOD_SC** are the number of IMO clock cycles required for each integration cycle or balancing cycle for mutual-cap and self-cap scans. One TX pulse is therefore two of these periods, which results in Equation 15.

$$\text{TX Frequency} = \frac{48\ \text{MHz}}{2 \cdot \text{TX\_PERIOD\_xx}}$$

Equation 15

The allowed range for the TX period is 68 to 1000. This corresponds to TX frequencies of 352.9 kHz to 24.0 kHz. Mutual-cap scans and self-cap scans each have their own TX frequency.

The parameter **Device Setup: SELF_Z_MODE** selects the axis that is scanned for self-cap. The options are "Self Cap RX" (scans only RX electrodes), "Self Cap TX" (scans only TX electrodes), and "Self Cap TX & RX" (scans all electrodes).

6.3.2 Procedure to determine TX frequency

In most automotive applications, EMI/EMC emissions is the most restrictive constraint. The steps in the following sections are structured accordingly. Process overview:

1. Find the panel speed to set an upper limit on the TX frequency.
2. Choose a TX frequency/harmonics to minimize overlap with emission requirements.
3. Confirm that display noise is not a problem.
4. Review and make trade-offs as needed.

6.3.2.1 Define maximum TX frequency

Typically, Infineon performs a panel speed simulation prior to panel manufacturing. One output of the simulation is the panel's three-time-constant frequency ($F_{3\tau}$). This value is an estimate of the upper limit on the range of workable TX frequencies. If the selected TX frequency (determined from the steps below) is significantly (for example, 30%) lower than the simulated panel frequency, then the panel speed can safely be assumed to be acceptable. If the selected TX frequency is close to the F3τ value, then the panel uniformity test (see Section 6.3.7) should be performed to check the panel speed. If the TX frequency is significantly above the $F_{3\tau}$ value, then it should not be used.

If the simulation data is not available, or a double-check is desired, then follow one of the processes in Section 6.3.3 to measure the $F_{3\tau}$ value.

6.3.2.2 TX frequency meeting emissions requirements

The system will have the highest emissions at the fundamental and odd harmonics of the TX frequency. Typical system requirements specify a frequency-dependent emission profile, for example having strict emission requirements at AM radio bands and looser requirements at other frequencies.

Consider an example with strict emission requirements from 150 kHz to 300 kHz, and above 540 kHz. In this case, we can choose a frequency whose fundamental falls below 150 kHz, and whose third and fifth harmonics fall between 300 kHz and 540 kHz. For example, a TX frequency of 105 kHz will have the fundamental at 105 kHz, third harmonic at 315 kHz, and fifth harmonic at 525 kHz, avoiding the emission requirements. In this case, a TX period of 229 (TX frequency of 104.8 kHz) would be a good choice.

If no single frequency can be found:

- Evaluate the TX spreader, see Section 6.3.4
- Refer to the fine-tuning in Section 6.3.5

Note that the touch controller has a clock frequency tolerance of ±2%, and the display controller will have some frequency tolerance as well. These variations must be considered when selecting the TX frequency.

6.3.2.3 Check display noise

Display noise can be reduced by using a Manhattan style sensor pattern (which effectively shields the RX lines from noise), having a large air gap between display and ITO panel (which reduces coupling from display to panel), or using a low-noise display technology such as OLED. However, display noise can be significant when using a diamond sensor pattern, bonding the sensor directly to the display (no air gap), and using a high-noise display technology (e.g. dot or line inversion). Unfortunately, the latter is typically the case for automotive projects.

To check for display noise:

1. Disable all other scan types:
   a) Set **Touch Mode: TOUCHMODE_CONFIG** to "FingerOnly"
   b) Set **HOVER: HOVER_ENABLE** to Disabled
2. Disable the raw data filters by disabling the parameters **Raw Processing: MC_RAW_FILTER_MASK: xxx Filter** and **Raw Processing: SC_RAW_FILTER_MASK: xxx Filter**.
3. In the TTHE:
   a) Select Heat Map display mode.
   b) In Touch Display, set DataType to RawPostFilter, and SensorValueType to "Max - Min".
4. Check the heat map with the display off, or with an all-black image.
5. Then try different images, including high-noise images (such as sub-pixel chess, vertical 1-pixel bars, and horizontal 1-pixel bars), and see how much noise is introduced.
6. If the change in noise is more than a few percent of the value seen with a standard-sized finger, then display noise must be considered.

If display noise may be a problem, follow the steps in Section 6.3.6 to evaluate the spectrum of display noise, and select a TX frequency that has low noise.

6.3.2.4 Evaluate selected TX frequency

If the best TX frequency found does not meet all the requirements, then a trade-off will need to be made. The final TX frequency selection may require testing different values in an EMI/EMC emissions chamber, and selecting the best value.

If the simulated F3τ frequency is close to or below the selected TX frequency, or if a check of the panel speed is beneficial, then perform the panel uniformity test (see Section 6.3.7).

6.3.2.5 Set panel refresh rate

The panel refresh rate should be set prior to further tuning. Set the TTHE Display Mode to "Touch Reporting" and place a single finger on the panel. The TTHE will display the scan rate in the "Packets/sec" field in the information strip at the bottom of the window. Set the mutual-cap TX pulses (**TSS: TX_PULSES_MC**) such that the required panel refresh rate is achieved.

6.3.3 Alternate procedures to determine panel 3τ value

This step is only needed if the panel speed simulation result is not available, or if it is desirable to double-check the result. It provides two methods of measuring the panel's $F_{3\tau}$ value.

6.3.3.1 TTHE assisted tuning

The recommended method of determining the minimum TX frequency is to use the TTHE assisted tuning. This allows a sweep across the available range of TX frequencies to evaluate the incoming sensor charge settling of the sensors with the longest TX/RX combined ITO route. The benefits of this method are that: the result is repeatable, no finger touch is needed, and it is suitable for automation. An example illustration is shown in Figure 29. The blue curve shows the total sensor charge of a sensor closer to the flex cable. The magenta curve shows the total sensor charge of another sensor further away from the flex cable with longer TX/ RX combined ITO route. Because of the larger RC time constant, the second sensor requires a longer time to reach the same percentage level of charge settling.

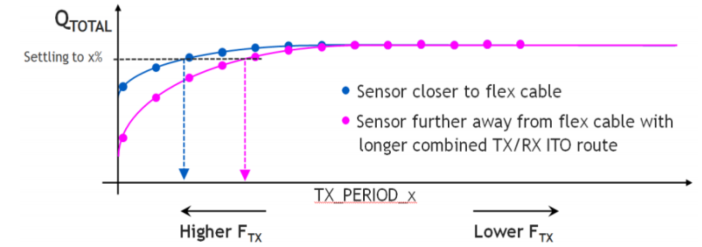


Figure 29 TX frequency sweep illustration

To sweep the TX frequencies, perform the following steps:

1. For mutual-cap panel scanning, set the parameter **TSS: MTX_ORDER** to '1' to minimize the TX charge pump drive limitation.
2. Set **TSS: TX_FREQ_METHOD_MC** (or **TSS: TX_FREQ_METHOD_SC** for self-cap) to "Maximizing Integration Time".
3. Set **Charger Armor: CHARGER_ARMOR_ENABLE** to Disabled.
4. Double-click or Right-click the AT button AT next to **TSS: TX_PERIOD_MC or TSS: TX_PERIOD_SC** that is shown in Figure 30 to open the Assisted Tuning window. Select "TX_PERIOD Sweep (panel settling)".

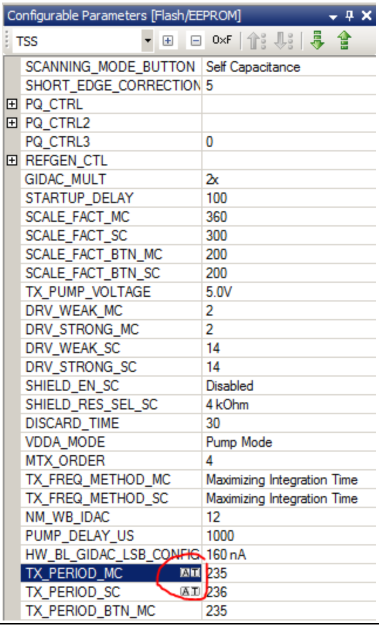


Figure 30 Assisted tuning buttons

5. Select sensors with the highest RC time constant, which is usually the sensor with the longest TX/RX combined (see Figure 33).
6. Set Target Percentage Settling to the desired settling level. In general, it is recommended to reach about 90% settling.
7. Set Min TX Period to 68 (F_{TX} = 353 kHz), Max TX Period to 500 (F_{TX}=48 kHz) and Step Size to 2.
8. Click "Run TX Sweep" to start the analysis, which will take several minutes to complete.

An example of the Assisted Tuning window is shown in Figure 31. Inputs are entered in the lower-right panel of the window. Sensors can be selected in the lower-left touch display panel. The results are plotted on the upper panel along with the target settling percentage level for each sensor. Each curve can be individually selected.

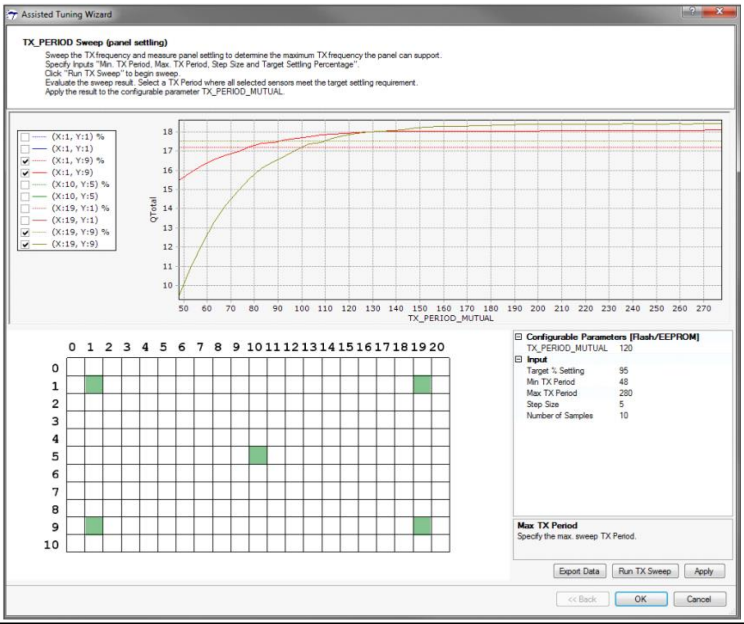


Figure 31 Example of TTHE assisted tuning window for signal settling TX frequency sweep

Identify the sensor that requires the longest to reach the target settling. Deselect all other sensors in the touch display. The minimum **TSS: TX_PERIOD_MC** (or **TSS: TX_PERIOD_SC**) is the value required to reach the target settling percentage level.

For example, if the period value required to meet the selected settling percentage is found to be 200, then the 3τ frequency is 120 kHz (see Equation 15).

6.3.3.2 Manual measurement

Manually measuring the $F_{3\tau}$ value requires using an oscilloscope to measure when the voltage at the worst-case sensor reaches 95% of its final value. Figure 32 shows a generic example of a capacitor charging through a 5-kΩ resistor.

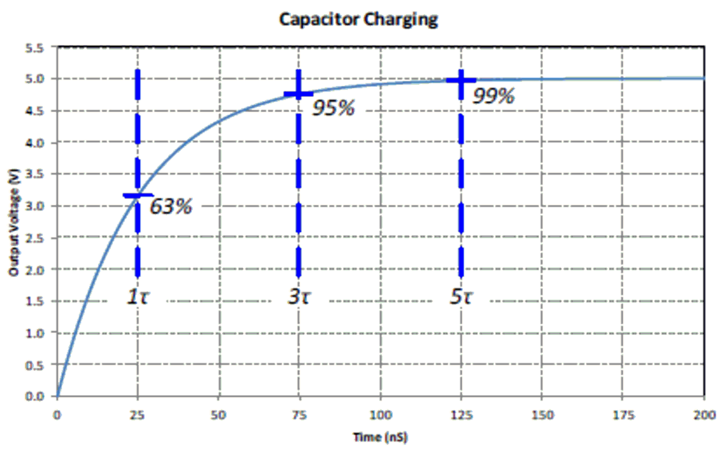


Figure 32 Capacitor charging example with R = 5 kΩ and C = 5 pF

Firstly, determine the sensors with the worst-case RC values. The highest RC time constant is usually for sensors with the longest combined TX/RX path. It is also recommended to avoid corner and edge sensors. Figure 33 shows the sensors to measure for single- and double-routed panels.

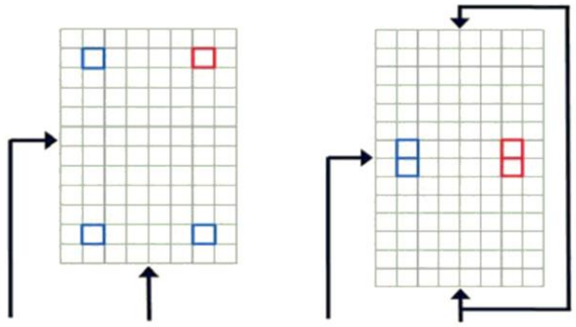


Figure 33 Sensors to measure single-routed (left) and double-routed (right) panels

To measure the 3τ value, place a small (approx. 1 cm diameter), metal, coin-shaped probe on the panel over the worst-case sensor locations described earlier. Set the TX frequency to 48 kHz by setting the parameter **TSS: TX_PERIOD_PANEL** to 500. Use an oscilloscope to measure the waveform at the probe. The $F_{3\tau}$ value is given by the following equation.

$$F_{3\tau} = \frac{1}{2 \cdot T_{95\%}}$$

Equation 16

For example, if the 95% time is found to be 4 µs, then the 3τ frequency is 125 kHz. Therefore, the **TSS: TX_PERIOD_PANEL** value must be a minimum of 192 (see Equation 15).

6.3.4 TX spreader

6.3.4.1 Background

Note that the TX spreader has some differences between CYAT6165X/8165X (TSG6L) and CYAT8168X (TSG6XL):

- TX spreader was not implemented in TSG6L base FW version 1.0
- Implementation of the spreader differs very slightly (see Section 6.3.4.2)
- TSG6XL has a separate spreader for self-cap and mutual-cap
- Parameter name for the number of different TX frequencies:
  - TSG6L: **TSS: TX_SPREADER_STEP_NUM**
  - TSG6XL: **TSS: MC_TX_SPREADER_PULSES** and **TSS: SC_TX_SPREADER_PULSES** (used in this doc)
- Parameter name for the change in size of each TX phase:
  - TSG6L: **TSS: TX_SPREADER_STEP**
  - TSG6XL: **TSS: MC_TX_SPREADER_STEP** and **TSS: SC_TX_SPREADER_STEP** (used in this doc)

Spreading the TX frequency helps to reduce the peaks of the emission spectrum, but also makes channel frequency response wider. Therefore, display noise can have more significant impact. Interference with channel frequency response and the display noise spectrum should be minimized. This can be done by proper selection of TX frequency and spread band.

From the EMI/EMC emissions point of view, the spread range should be as wide as possible. However, at the same time, a particular display noise spectrum should be considered when determining the TX frequency spread range.

The highest TX frequency in a spreading sequence is called the base frequency and is defined by the **TSS: TX_PERIOD_MC** (**TSS: TX_PERIOD_SC** for self-cap) parameter. This parameter defines the number of system clocks in each half period. The TX period is therefore the system clock period multiplied by double this parameter.

The TX spreader only changes the TX frequency to lower values. Two parameters control the TX spreader properties:

- **TSS: MC_TX_SPREADER_STEP** (or **TSS: SC_TX_SPREADER_STEP**) is the step size. This is the number of system clocks added to each phase (high and low) of the current scan period to create the next TX period. Each TX period:
  - The next TX period parameter (e.g., **TSS: TX_PERIOD_MC**) is increased by this parameter (as the parameter defines only a half-period).
  - The actual TX period is increased by twice the number of system clocks specified by this parameter (as it is applied to each half-period).
- **TSS: MC_TX_SPREADER_PULSES** (or **TSS: SC_TX_SPREADER_PULSES**) is the number of steps. This is the number of different TX frequencies used for one complete modulation cycle.

The TX spreader is disabled if the step size is zero, or the number of pulses is one. The maximum and minimum scan periods can be calculated as follows (shown for mutual-cap only):

$$\mathrm{MinTXClks} = \mathrm{TSS{:}\ TX\_PERIOD\_MC} * 2$$

Equation 17

$$\mathrm{MaxTXClks} = \mathrm{Min} + ((\mathrm{TSS{:}\ MC\_TX\_SPREADER\_PULSES} - 1) * (\mathrm{TSS{:}\ MC\_TX\_SPREADER\_STEP} * 2))$$

Equation 18

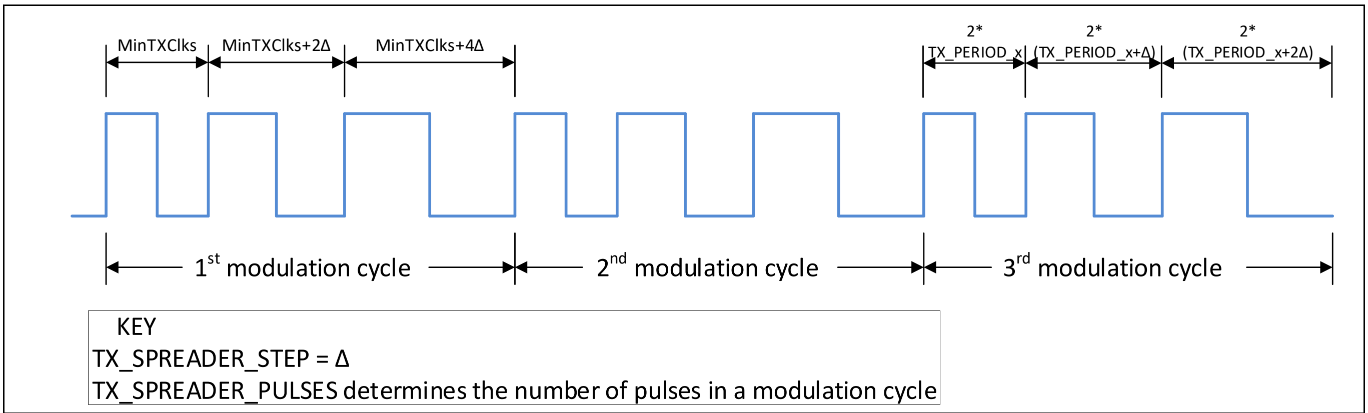


Figure 34 Example of TX conversions with spread

6.3.4.2 TX spreader implementation differences

When the number of TX pulses is not an exact multiple of the number of spreader pulses, the TSG6L and TSG6XL implementations differ slightly.

TSG6L devices prioritize the number of TX pulses. The last (incomplete) spreader modulation cycle is truncated. The next TX sensor scan then starts from the base frequency. So the number of spreader steps in the last modulation cycle is MOD(**TSS: TX_PULSES_MC** / **TSS: TX_SPREADER_PULSES**). For example (row 2 in Table 16), a project with 13 TX pulses, a base period of 200, a spreader step of 4, and a spreader pulses of 3 would have just 1 pulse in the last modulation cycle, and therefore have the following TX scan format (TX_PERIOD half-period clocks): 200, 204, 208, 200, 204, 208, 200, 204, 208, 200, 204, 208, 200.

TSG6XL devices prioritize the spreader modulation cycle. The number of TX pulses is increased to the next multiple of the spreader pulses. For example (row 2 in Table 16), a project with 13 TX pulses, a base period of 200, a spreader step of 4, and a spreader pulses of 3 would have the number of TX pulses increased to 15, and therefore have the following TX scan format (TX_PERIOD half-period clocks): 200, 204, 208, 200, 204, 208, 200, 204, 208, 200, 204, 208, 200, 204, 208.

These different methods are shown in the equations below, and illustrated in the following table.

$$\text{TSG6L TX Pulses} = \text{TSS: TX\_PULSES\_MC}$$

Equation 19

$$\text{TSG6XL TX Pulses} = \text{roundup}\left(\frac{\text{TSS: TX\_PULSES\_MC}}{\text{TSS: TX\_SPREADER\_PULSES}}\right) * \text{TSS: TX\_SPREADER\_PULSES}$$

Equation 20

Table 16 Parameter and actual spreader pulses

Base TX pulses	Base TX half-period	Spreader step	Spreader pulses	TSG6L last scan half-period	TSG6XL last scan half-period	TSG6L real TX pulses	TSG6XL real TX pulses
12	200	4	3	208	208	12	12
13	200	4	3	200	208	13	15
14	200	4	3	204	208	14	15
15	200	4	3	208	208	15	15
16	200	4	3	200	208	16	18
17	200	4	3	204	208	17	18
18	200	4	3	208	208	18	18
19	200	4	3	200	208	19	21
20	200	4	3	204	208	20	21
21	200	4	3	208	208	21	21
22	200	4	3	200	208	22	24

6.3.4.3 TX spreader TSG6L specifics

The TSG6L TX spreader also has two other specific features that the tuning engineer should be aware of.

Firstly, if glove is enabled, the spreader must by synchronized between finger mode and glove mode. The only requirement is that the last scan period have the same number of spreader pulses. This is expressed in Equation 21, which can be applied to either mutual-cap or self-cap.

$$\text{MOD}(\text{TX\_PULSES}, \text{SPREADER\_STEPS}) = \text{MOD}(\text{TX\_PULSES\_GLOVE}, \text{SPREADER\_STEPS})$$

Equation 21

Secondly, the TSG6L TX spreader is not implemented in HW, as is the case for the TSG6XL. Therefore, the TSG6L TX spreader may occasionally miss (or duplicate) a TX frequency due to the small interrupt lock-out periods in critical code.

Using the settings in row 2 of Table 16 as an example, the expected TX frequencies would be: 200, 204, 208, 200, 204, 208, 200, 204, 208, 200, 204, 208, 200. However, the SW TX spreader could result in a sequence of TX frequencies of: 200, 204, 208, **208**, 204, 208, 200, **200**, 208, 200, 204, 208, 200. However, this would be a very rare occurrence.

6.3.5 EMI/EMC emissions fine tuning

The hardware should have been optimized for EMI/EMC emissions at the design stage, using the guidelines in the Module Design Best Practices document (001-50467). Most automotive EMI/EMC emissions standards specify emission standards specify limits above 100 kHz, for example, CISPR 25 – 150 kHz. A TX frequency should be chosen below the lowest frequency in the EMI/EMC emissions requirements. Doing this keeps the 1st harmonic (the most powerful) out of the tested emission spectrum. At the same time, the TX frequency selection must account for the display noise spectrum as well as minimum panel settling.

Emission tuning is an iterative flow, if the requirements cannot be met, trade-offs will need to be made.

6.3.5.1 TX driver parameters

Table 17 shows TX driver parameters which influence the emission spectrum. Following the tuning priority is not strictly required, but it is recommended. Each project has different requirements, influencing which parameters should be tuned depending on the impact on the whole system.

Table 17 TX driver parameters

Tuning order	Parameter	Description	Influence on emission spectrum	Influence on the whole system
1	**TSS: TX_PERIOD_MC**	Determines half of TX period in system clocks.	Select the appropriate TX frequency to hide the 1st harmonic on the emission spectrum out of the specified limits.	Always check a particular display spectrum during TX frequency selection. 1. The lower the TX frequency the smaller number of TX pulses can be done to achieve required refresh rate. That correspondingly decreases the system immunity to the external noises. 2. TX frequency cannot be higher than 3 times of the panel's settling time.

Tuning order	Parameter	Description	Influence on emission spectrum	Influence on the whole system
2	**TSS: TX_PERIOD_SC**	↑	↑	↑
3	**TSS: MTX_ORDER**	Multi-phase TX order. Number of TX lines driven simultaneously.	1. Single TX (**TSS: MTX_ORDER** = 1) produces the lowest emission. 2. Odd Multi TX orders produce lower emission than even.	The higher Multi TX order the better system's immunity to the external noises.
4	**TSS: MC_TX_SPREADER_STEP** **TSS: SC_TX_SPREADER_STEP**	Number of system clocks for stepping width applied to each TX phase. Each subsequent TX period will be elongated by **twice** this value.	TX spreader spreads TX harmonics and makes emission spectrum flatter.	TX spreader decreases system's immunity to the external noises including display noise.
5	**TSS: MC_TX_SPREADER_PULSES** **TSS: SC_TX_SPREADER_PULSES** CYAT8168 (TSG6XL) only **TSS: TX_SPREADER_STEP_NUM** CYAT6165/8165 (TSG6L) only	Defines the number of pulses used for 1 modulation cycle. See next section for TX spreader tuning and detailed description.		
6	**TSS: TX_PUMP_VOLTAGE**	TX driver voltage if pump set to Bypass Mode.	The lower the TX driver voltage the lower emission in the whole frequencies range.	Low TX voltage decreases SNR.
7	**TSS: SHIELD_EN_SC** **TSS: SHIELD_EN_BTN_SC**	Enables/disables TX shield for self-cap scan.	All unused sensors during self-cap measurements are connected to TX shield (**TSS: SHIELD_EN_SC** = Enabled) or ground (**TSS: SHIELD_EN_SC** = Disabled). Enabled TX shield	Poor water rejection / wet finger tracking with disabled TX shield.

Tuning order	Parameter	Description	Influence on emission spectrum	Influence on the whole system
			increases emission.	
8	**TSS: REFGEN_CTL: TX SHIELD**	Specifies TX shield output voltage.	The lower the shield TX voltage the lower radiated emission.	The optimal value TX SHIELD voltage should be equal to RXDAC value (true self-cap measurements).
9	**TSS: REFGEN_CTL: RXDAC**	Specifies swing voltage for self-cap measuring mode.	The lower the swing voltage the lower radiated emission.	The higher the swing voltage the better self-cap SNR.
10	**TSS: DRV_STRENGTH_MC**	Controls the TX voltage rising and falling edges rate for mutual-cap measurements.	The lower the edge rate the lower amplitude of radiated emission harmonics, except the main harmonic.	The higher the TX edge rate, the faster the signal charge will be settled allowing a higher TX frequency.
11	**TSS: DRV_STRENGTH_SC**	Controls the TX voltage rising and falling edges rate for self-cap measurements.	↑	The higher the TX edge rate, the faster the signal charge will be settled allowing a higher TX frequency.
12	**TSS: TSS_TX_DUTY_CYCLE**	Defines the duty cycle of TX pulses.	This parameter allows spreading the energy between the odd and even harmonics.	Appling higher/lower duty cycle than 50%, the panel settling time should be considered.
13	**TSS: TX_PULSES_MC** **TSS: TX_PULSES_SC** **TSS: TX_PULSES_GLOVE_MC** **TSS: TX_PULSES_GLOVE_SC**	Number of TX pulses per conversion for each scan type.	The smaller number of TX pulses, the smaller average radiated emission (not peak).	The bigger number of TX pulses the higher SNR.
14	**TSS: SHIELD_RES_SEL_SC**	Determines value of the resistor in TX driver for stability.	This parameter is relevant only if TX shield is used. Different resistor values slightly change the TX shield waveforms that little impact	Almost no impact on touch performance.

Tuning order	Parameter	Description	Influence on emission spectrum	Influence on the whole system
			on emission spectrum.	

6.3.5.2 System tuning parameters for radiated emission reduction

This section describes some system tuning features that can have an impact on emission spectrum. The features are not applicable to all cases and solutions. Table 18 shows system tuning parameters which influence the emission spectrum.

Table 18 System tuning parameters that influence on emission

Parameter	Description	Influence on emission spectrum	Influence on the whole system
Device Setup: SENSOR_ASSIGNMENT	Assignment of TX/RX sensors to X/Y axes.	Can reduce emission in some designs.	Swapping TX/RX sensors is possible only for touchscreen panel with symmetrical structure (SSD, DSD ...).
Device Setup: SELF_Z_MODE	Determines a set of sensors which are used for self-cap scan.	↑	Usually not applicable to Manhattan panel structure.
Device Setup: MAX_MUTUAL_SCAN_INTERVAL	Maximum interval between mutual-cap scans.	The more rarely the system performs mutual-cap scan, the lower average emission.	The lower scan interval, the lower refresh rate.
Device Setup: MAX_SELF_SCAN_INTERVAL	Maximum interval between self-cap scans.	The more rarely the system performs self-cap scan, the lower average emission.	Low self-cap scan interval reduces water rejection performance.
Device Setup: LOW_POWER_ENABLE	Enabled/Disables low power mode.	Can reduce average emission if low power mode is used.	Low power mode has not any negative effect on the system.
Device Setup: ACT_INTRVL0	Initial active state refresh interval.	Average emission can be reduced if longer interval is used.	Low interval can decrease a refresh rate or increase a latency for touch/glove detection.
Device Setup: ACT_LFT_INTRVL0	Initial active look-for-touch state refresh interval.	↑	↑

Parameter	Description	Influence on emission spectrum	Influence on the whole system
Device Setup: LP_INTRVL0	Initial low power state refresh interval.	↑	↑
Fingers: ACT_LFT_EN	Method if scanning when no touches on the panel. If disabled, mutual-cap scan is used. If enabled, self-cap scan is used.	Self-cap scan produces a big emission increase, especially if the TX shield is used. Emission significantly reduced if self-cap scan is disabled.	Disabled increases power consumption.
Scan Filtering: WATER_REJ_ENABLE	Enables/disabled advanced water rejection (enables self-cap scanning).	↑	Disabled reduces immunity to water on the touchscreen.

6.3.6 Display noise spectrum evaluation

This section uses the touch controller to measure the display noise. The outcome of this section is a spectrum of display noise that can be used to find a quiet TX frequency.

1. Set **Touch Mode: TOUCHMODE_CONFIG** to "FingerOnly".
2. Double-click the "AT" button next to **TSS: TX_PERIOD_MC** (mutual-cap) or **TSS: TX_PERIOD_SC** (self-cap) to open the Assisted Tuning window.

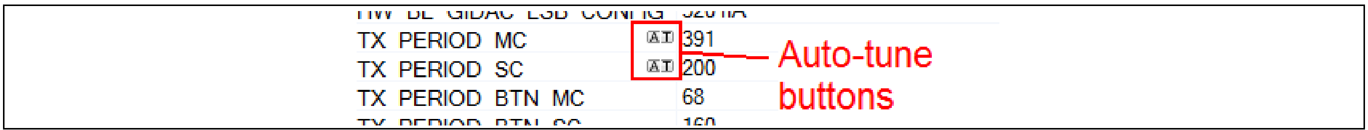


Figure 35 TX period Assisted Tuning "AT" buttons

3. Select "TX_PERIOD Sweep (noise)".
4. Select several sensors across the panel, for example in the center and near each corner.
5. Set Noise Type to Raw-count and Data Type to Peak-to-Peak.
6. Set the minimum and maximum TX periods of interest.
7. Set the step size to 1 for the finest possible readings, or to a larger value for a faster sweep.
8. Set the Number of Samples to at least 20. A higher number of samples gives a more accurate noise measurement but takes longer to test.
9. Select a display image. Different images have different levels of noise depending on the display and touch configuration. Multiple tests should be performed using at least the following images:
- Typical picture (for a baseline)
- White
- 1-pixel black/while horizontal lines
- 1-pixel black/while vertical lines
- 1-pixel chess pattern
- Sub-pixel chess pattern (usually the highest noise)

10. Click Apply, then Run TX Sweep. The test may take several minutes to complete.

11. Export the results into a spreadsheet for analysis of multiple images.
12. Evaluate the resulting spectrum to identify quiet frequencies. Display noise will be minimized when the TX frequency falls in one of the quiet ranges. Figure 36 shows a sample output. In this case, a TX period of 165 (TX frequency = 145 kHz) would be a good choice.
13. Click OK to exit the auto-tune window.

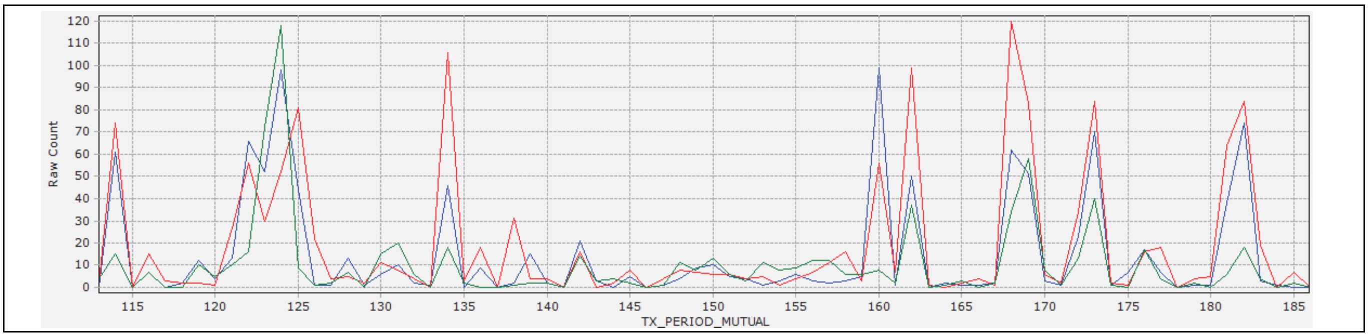


Figure 36 Sample output of display noise evaluation

If a frequency cannot be found that allows both EMI/EMC emissions to pass and display noise to be acceptable, then display synchronization can be considered. This method of display noise mitigation should only be used if it was originally planned and/or all other display noise mitigation methods have been exhausted. See Section 8.3 for details. Note that hardware support is required, so early planning is important. The display driver must provide the synchronization signal(s) (usually HSync, the display's horizontal line synchronization) to a touch controller GPIO.

6.3.7 Panel uniformity test

This section describes a test that finds the highest allowable TX frequency. The test can be used, if necessary, to confirm that the selected TX frequency is not too high for the panel.

Note that this procedure requires a low-noise operating environment. For the steps below, ensure that the display is off (or showing an all-black image). Make sure that the system is well-grounded to prevent system noise.

1. Set **TSS: TX_PERIOD_MC** to about one-half of the 3τ frequency (see Equation 15 for conversion).
2. Set **TSS: MTX_ORDER** to 1.
3. Set **TSS: TX_FREQ_METHOD_MC** to "Maximizing Integration Time".
4. Set **TSS: MC_TX_SPREADER_STEP** to 0 (**TSS: TX_SPREADER_STEP** for TSG6L).
5. Set **Charger Armor: CHARGER_ARMOR_ENABLE** to "Disabled".
6. Set TTHE to show difference counts by selecting the Heat Map view:
   - Select DiffCounts in the Data section of the Touch Display Settings window
   - Select MaxHold in the Sensors section of the Touch Display Settings window
7. Using a 10-15 mm grounded metal finger, slowly cover the panel:
   - Note the maximum counts at the corners, edges, and center of the panel (see Figure 37)
   - Values will not be equal, but the relationship among them should be noted and recorded
   - For example, the diff-count at the center might be 10% higher than at the edge

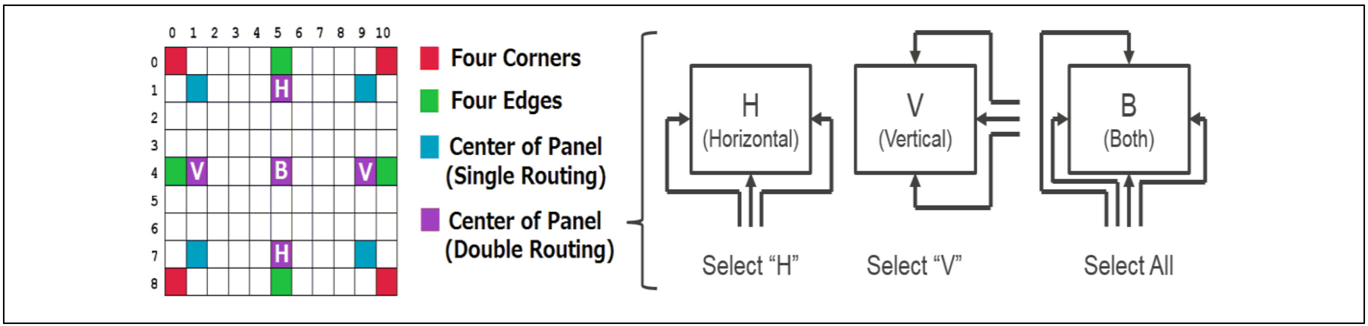


Figure 37 Location of corners, edges, and center for different routing

8. Increase the TX frequency by 10%, and repeat Step 7:
   - Reset the Max Hold display (MaxHold >Current >MaxHold), in the Touch Display Settings
   - Check if the relationship among the diff-counts have changed by more than 10%
   - If a precise TX frequency must be obtained, then smaller steps can be used
9. Continue increasing the TX frequency and checking the diff-count relationships:
   - At some point, the diff-counts in some locations will fall, while others will stay the same, changing the diff-count relationships
   - When a panel location deviates from the original relationship by more than 10%, then the TX frequency is at the highest allowed value

6.3.8 Parameters

Table 19 TX waveform parameters

Configurable Parameter	Description	Selection
TSS: REFGEN_CTL: TX SHIELD	Output voltage for the TX shield.	0 – 15 (default = 0)
TSS: REFGEN_CTL: RXDAC	Output voltage for the RX DAC.	0 – 15 (default = 0)
TSS: TX_PUMP_VOLTAGE	Target voltage for the charge pump when enabled by **TSS: VDDA_MODE**.	4.3 V / 4.6 V / 5.0 V (default = 5.0V)
TSS: DRV_STRENGTH_MC	This parameter specifies the drive strength control for mutual-cap TX scans.	0 – 15 (default = 2)
TSS: DRV_STRENGTH_SC	This parameter specifies the drive strength control for self-cap TX scans.	0 – 15 (default = 14)
TSS: SHIELD_EN_SC	This parameter specifies if the TX shield driver is enabled for self-cap scans. Enabled TX shield improves Water Rejection but increases emission.	Enabled / Disabled (default = Enabled)
TSS: SHIELD_EN_BTN_SC	This parameter specifies if the TX shield driver is enabled for self-cap button scans. Enabled TX shield improves Water Rejection but increases emissions.	Enabled / Disabled (default = Enabled)
TSS: SHIELD_RES_SEL_SC	This parameter specifies the resistor size (2 KΩ to 16 KΩ in the TX shield driver for stability. This setting has minimal impact on performance.	4 kOhm / 8 kOhm / 16 kOhm / 2 kOhm (default = 4 kOhm)
TSS: MTX_ORDER	This parameter defines the multi-phase TX order.	1 – 19

Configurable Parameter	Description	Selection
		(default = 4)
TSS: TX_PERIOD_MC	This parameter defines the number of system clocks in each TX half period for mutual-cap or self-cap scans. Where stated "48/68", the minimum value is 48 for TSG6L devices and 68 for TSG6XL devices.	68 – 1000 (default = 235)
TSS: TX_PERIOD_SC		48/68 – 1000 (default = 236)
TSS: TX_PULSES_MC	These parameters define the number of TX pulses per conversion for scans of each type for all known object types. The "BTN" parameters are for button scans, and the "GLOVE" parameters are for glove scans.	1 – 255 (default = 20)
TSS: TX_PULSES_SC		1 – 255 (default = 64)
TSS: TX_PULSES_GLOVE_MC		0 – 255 (default = 36)
TSS: TX_PULSES_GLOVE_SC		0 – 255 (default = 130)
TSS: TSS_TX_DUTY_CYCLE	This parameter defines the duty cycle of TX pulses, and allows spreading the energy between the odd and even harmonics.	30 – 50 (default = 50)
TSS: TX_SPREADER_STEP_NUM CYAT6165/8165 (TSG6L) only	Defines the number of pulses used for 1 modulation cycle. TSG6XL only, set to 1 to disable the TX spreader.	4 steps / 8 steps (default = 4 steps)
TSS: MC_TX_SPREADER_PULSES CYAT8168 (TSG6XL) only		1 – 12 (default = 1)
TSS: SC_TX_SPREADER_PULSES CYAT8168 (TSG6XL) only		
TSS: TX_SPREADER_STEP CYAT6165/8165 (TSG6L) only	Number of system clocks for stepping width applied to each TX phase. Each subsequent TX period will be elongated by twice this value. Set to zero to disable the TX spreader.	0 – 100 (default = 0)
TSS: MC_TX_SPREADER_STEP CYAT8168 (TSG6XL) only		
TSS: SC_TX_SPREADER_STEP CYAT8168 (TSG6XL) only		
Device Setup: SENSOR_ASSIGNMENT	Assignment of TX/RX sensors to X/Y axes.	RX=X; TX=Y / RX=Y; TX=X (default = RX=X; TX=Y)
Device Setup: SELF_Z_MODE	Determines a set of sensors which are used for self-cap scan.	Self Cap RX / Self Cap TX / Self Cap TX & RX (default = Self Cap RX)

Configurable Parameter	Description	Selection
Scan Filtering: WATER_REJ_ENABLE	Enables or disables the advanced water rejection feature; when enabled, both mutual-cap and self-cap scans are performed on every panel scan.	Enabled / Disabled 6165X/8165X (TSG6L) (default = Disabled) 8168X (TSG6XL) (default = Enabled)
Fingers: ACT_LFT_EN	Enables or disables self-cap scanning in look-for-touch mode. When enabled (and water rejection is disabled), self-cap scanning is only performed in LFT mode (and once every MAX_SELF_SCAN_INTERVAL).	Enabled / Disabled (default = Disabled)

6.4 Calibration verification

The calibration process was introduced in Section 2.5. Now that the TX drive has been configured, the calibration process must be verified.

The calibration procedure is designed to get average raw-counts as close to zero as possible. For most designs, the default settings will perform well. However, the results should always be checked to ensure a good calibration is possible, and there is enough scope for manufacturing variance.

There are three parts of the calibration verification:

1. IDAC Range
2. Raw-data Counts
3. Manufacturing Variance

6.4.1 IDAC range verification

The IDAC calibration results can be viewed using the TTHE software by (see Figure 38):

1. Select View > Tool Windows > IDAC Data
2. Click the Read button :

	Rx Attenuator	I_BAL
Mutual	3	18
Self Rx	6	12
Self Tx	4	12

Gain Correction		RxCh0	RxCh1	RxCh2	RxCh3	RxCh4	RxCh5	RxCh6	RxCh7	RxCh8	RxCh9	RxCh10	RxCh11	RxCh12	RxCh13	RxCh14	RxCh15	RxCh16
Mutual	(+ve phase)	32	44	34	30	32	35	31	40	4	55	30	31	49	31	-20	86	95
	(-ve phase)	-40	-41	-49	-47	-45	-45	-48	-47	-37	-63	-36	0	-38	-49	-9	-30	-36
Self	(+ve phase)	19	19	43	16	23	6	14	28	-2	29	9	7	54	23	-14	43	54
	(-ve phase)	-26	-28	-29	-26	-19	-29	-33	-5	-26	-58	-22	-16	-1	-35	-19	-16	-24

	RxCh0	RxCh1	RxCh2	RxCh3	RxCh4	RxCh5	RxCh6	RxCh7	RxCh8	RxCh9	RxCh10	RxCh11	RxCh12	RxCh13	RxCh14	RxCh15	RxCh16	RxC
Attenuator Trim	9	7	1	3	5	5	4	5	5	5	5	6	7	5	2	3	1	[illegible]

Baselining Data Mutual	
GIDAC	4
PWC	30

Baselining Data Self

GIDAC	Rx	Tx																						
	29	16																						
PWC	Rx0	Rx1	Rx2	Rx3	Rx4	Rx5	Rx6	Rx7	Rx8	Rx9	Rx10	Rx11	Rx12	Rx13	Rx14	Rx15	Rx16	Rx17	Rx18	Rx19	Rx20	Rx21	Rx22	
	53	7	4	4	3	4	4	4	4	5	5	5	6	5	7	6	7	8	8	11	11	18	107	
	Tx0	Tx1	Tx2	Tx3	Tx4	Tx5	Tx6	Tx7	Tx8	Tx9	Tx10	Tx11	Tx12	Tx13	Tx14	Tx15	Tx16	Tx17	Tx18	Tx19	Tx20	Tx21	Tx22	Tx23
	76	11	11	10	8	6	4	5	5	3	3	3	4	3	3	2	4	5	6	2	2	2	2	3

Figure 38 TTHE IDAC data window

Calibration automatically configures several settings, as shown in Figure 38. The most important settings are listed in Table 20, along with their recommended values. A value outside the recommended range does not automatically mean the panel is bad. But is does indicate that further investigation is needed.

Table 20 Calibration values and recommended ranges

Calibration value	Description	Recommended range
RX Attenuator	Part of the hardware baselining used to remove excess current while maintaining channel sensitivity. See Figure 38, circled in Red.	3, 4, 4.8, 6, 8, 12, 24
Balancing IDAC (I_BAL)	Part of the hardware baselining used to focus the channel sensitivity for manufacturability and life-cycle variance. See 0, circled in Green.	CYAT8165 (TSG6L) & GIDAC_MUL of 1x: 25 – 100
		CYAT8165 (TSG6L) & GIDAC_MUL of 2x: 13 – 100
		CYAT8168 (TSG6XL): 13 – 100
Global IDAC	Part of the hardware baselining used to control the strength of bias current added; usually calculated to be about 2/3 of the peak current on the RX channel. The Global IDAC (mutual, self, and button in separate tables) can be seen towards the bottom of the TTHE's "IDAC Data" window, above the "PWC" row. See Figure 38, circled in Blue.	0 – 100

Calibration value	Description	Recommended range
PWC	Part of the hardware baselining used to control the time the bias current is added; ideally the total bias current would equal the total current on the RX channel. Limit applies to all PWC values. See Figure 38, circled in Brown.	0 – (**TSS: TX_PERIOD_XX** * (**Calibration: XX_PWC_LIMIT_PERCENT** / 128))

If the IDAC calibration is not within the recommended range. There is a limited amount of tuning that can be made. The calibration tuning parameters and their default values are shown in Table 21:

Table 21 Default calibration parameters

Tuning Parameter	Default Value	Conversion Phase	Comment
TSS: RX_CAP_SEL CYAT6165/8165 (TSG6L) only	20 pF	All	CYAT6165/8165 (TSG6L) only Do not change from default
TSS: GIDAC_MULT	1x	HW Baselining	Increase if GIDAC >100
TSS: HW_BL_GIDAC_LSB_CONFIG	320 nA	HW Baselining	Decrease if GIDAC <5
Calibration: MC_PWC_LIMIT_PERCENT CYAT6165/8165 (TSG6L) only	115	HW Baselining	Decrease if GIDAC <3
Calibration: SC_PWC_LIMIT_PERCENT	64	HW Baselining	Decrease if GIDAC <3
Calibration: INT_VOLTAGE_MC **Calibration: INT_VOLTAGE_SC** **Calibration: INT_VOLTAGE_MC_BTN** **Calibration: INT_VOLTAGE_SC_BTN**	250	Balancing	Do not change from default
Calibration: BAL_TARGET_MC **Calibration: BAL_TARGET_MC_BTN** **Calibration: BAL_TARGET_SC_BTN**	38	Balancing	Default value corresponds to 30% Decrease if I_BAL=MIN and average raw is very large
Calibration: BAL_TARGET_SC	64	Balancing	Default value corresponds to 50% Decrease if I_BAL=MIN and average raw is very large

6.4.2 Raw-data counts

The raw-data can also show if there are problems with the calibration. Therefore, the raw-data counts should be checked: mutual-cap, self-cap, and button (as available). View the raw-count data in the TTHE by entering the Heat Map mode, and setting (in the Touch Display Settings window):

- Data Type = RawPostFilter
- Scan Type = Self (if applicable)
- SensorValue Type = Average (to make data easier to read)

The raw-count data indicates a bad calibration if the average value of the raw-count data is very far from zero. It is not necessary for the exact average value to be zero, it is usually good enough if both positive and negative raw-count data values can be seen. An example of raw-count data with a good calibration result is shown in Figure 39:

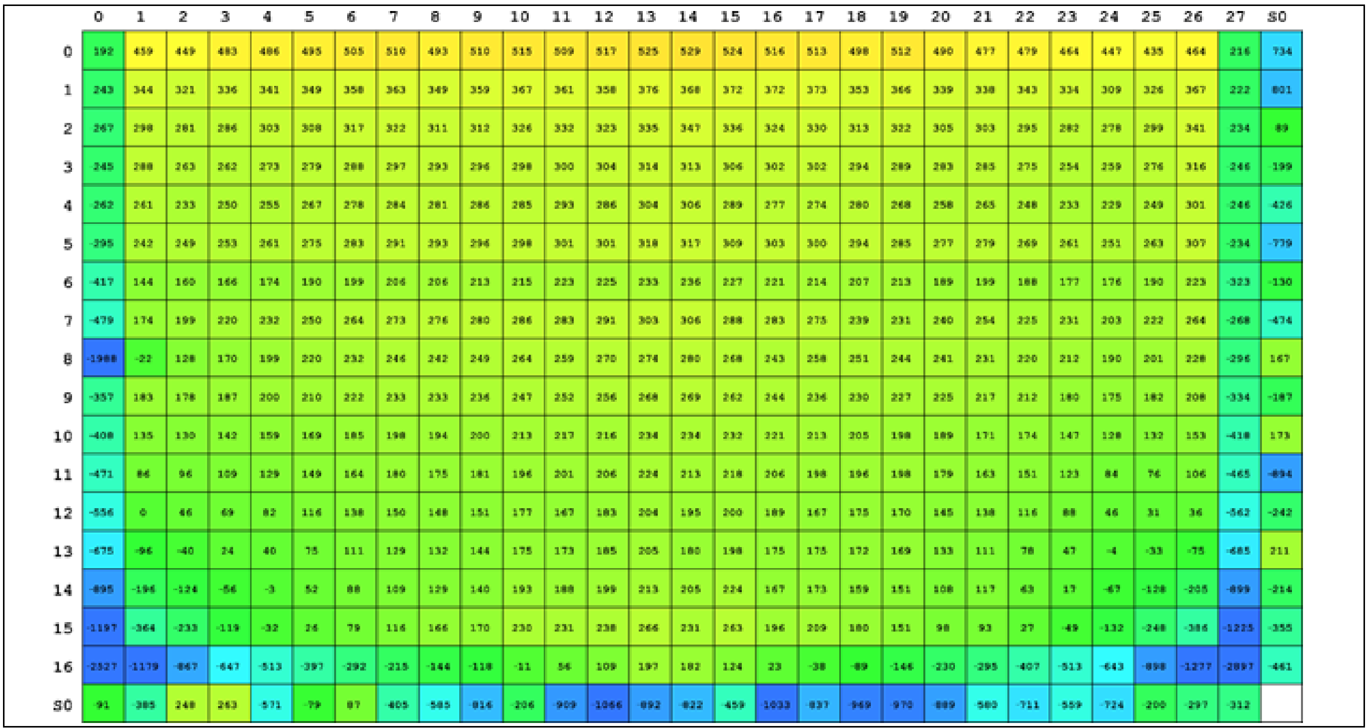


Figure 39 Raw-count data after a good calibration

Note that in the example of good calibration (Figure 39), none of the raw-count values are zero, or even close to zero. Good calibration is indicated by the average of all the raw-count data being close to zero. Contrast this with the raw-count data resulting from bad calibration, see Figure 40. Note that all the mutual-cap data is very high, and almost all the self-cap data is a very high negative value.

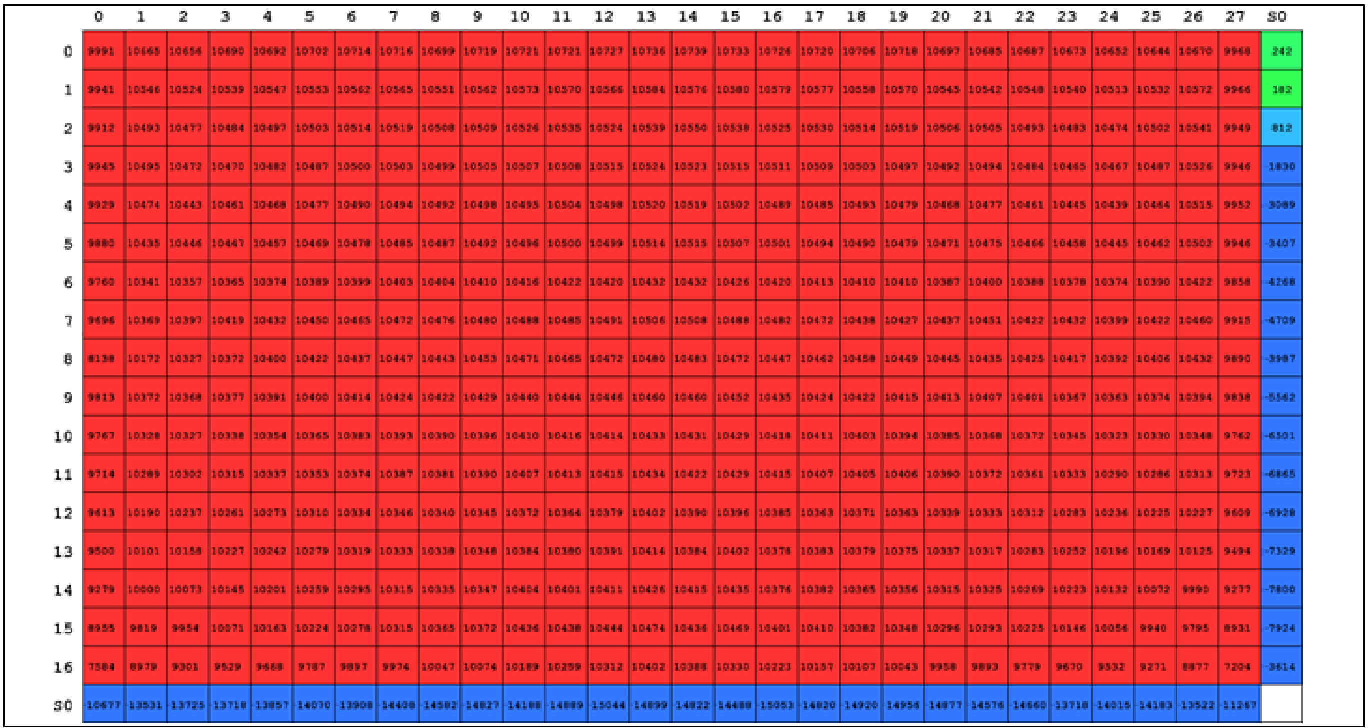


Figure 40 Raw-count data after a bad calibration

The example of bad calibration (Figure 40) is quite a typical case when calibration has failed. This data usually indicates calibration failed due to hardware limitations of the balancing IDAC and attenuator. The balancing IDAC current cannot be less than 1 µA for TSG6XL parts and 2 µA for TSG6L, as shown in Table 22:

Table 22 IDAC hardware limitations

Part Family	GIDAC_MULT	Min Balancing IDAC Code	Min IDAC current (µA)
TSG6XL	1x	12	1
	2x	12	2
TSG6L	1x	24	2
	2x	12	2

The calibration algorithm tries to decrease the balancing IDAC value to meet the balancing target, but cannot go beyond the HW limitation. As a result, the raw-count data may have some offset. An example is shown in Figure 41. This results in the IDAC code being the minimum supported by the device, which in this case (TSG6L with GIDAC_MUL of 1x) is 24, see Figure 42.

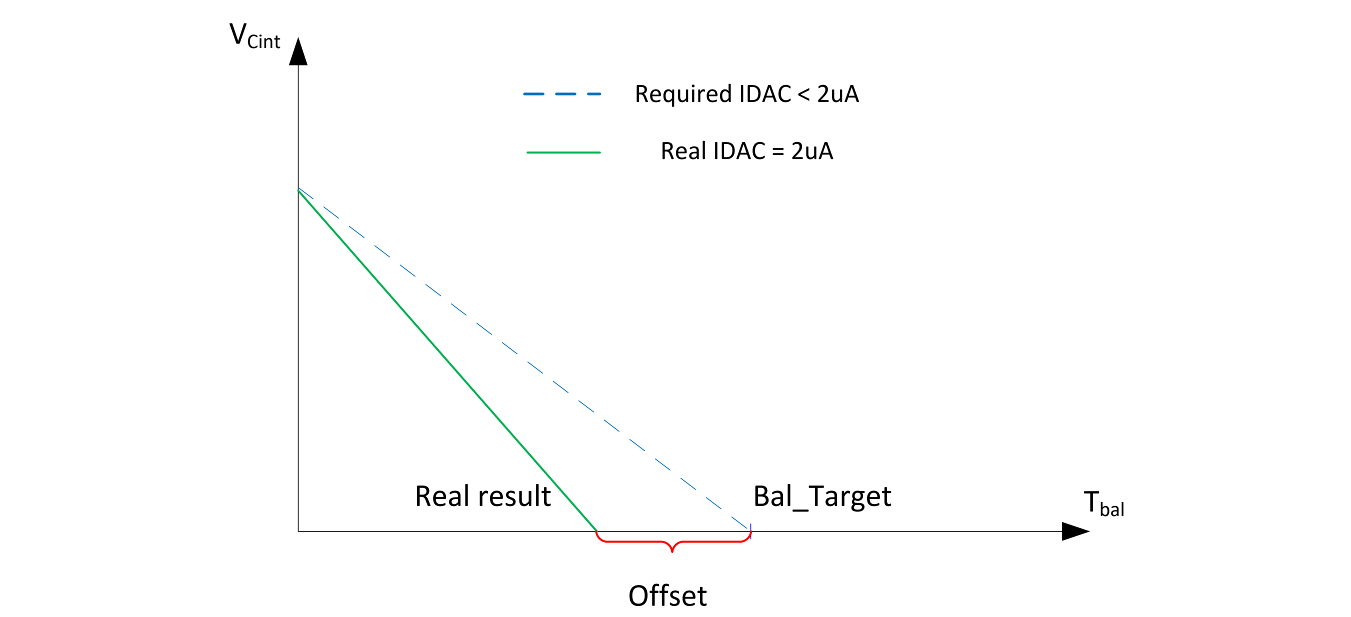


Figure 41 Raw-count data offset due to IDAC HW limitations

	Rx Attenuator	IDAC
Mutual	3	24
Self Rx	3	24
Self Tx	3	24

Figure 42 Calibration results with less than minimum current

To improve the results of calibration and decrease the offset in the raw-count data, the corresponding **Calibration: BAL_TARGET_XX** will need to be decreased until the IDAC current is greater than the min value shown in Table 22. This process can also be used if the average raw-count data is close to zero, but the IDAC is at its minimum value, to add extra safety margin for manufacturing variation.

6.4.3 Manufacturing variance

Another potential issue with calibration is inconsistency of the calibration results across a production run. This happens when the calibration error is about the same for different attenuator settings, see Figure 43:

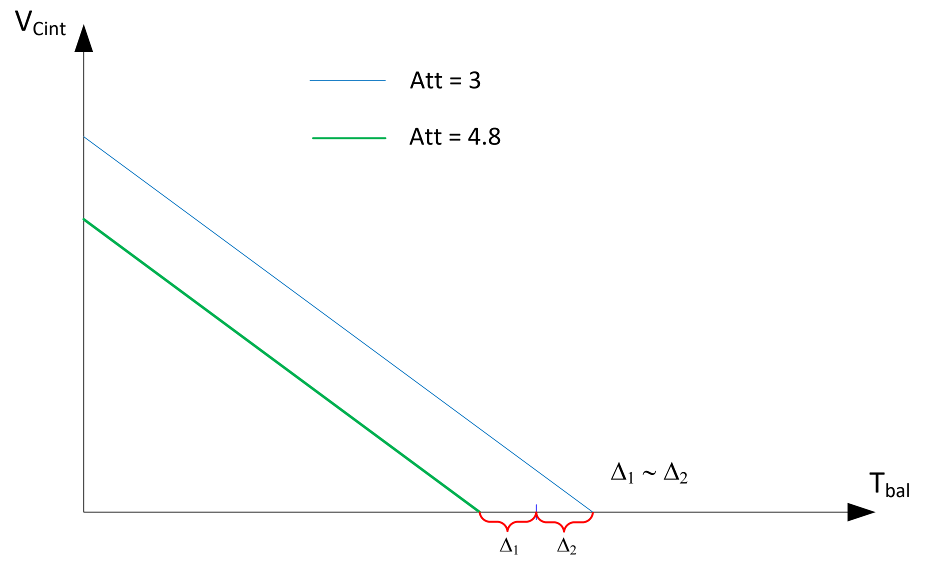


Figure 43 Attenuator calibration result midway between options

The firmware is not able to detect the situation where the manufacturing variation will cause different calibration results.

In the final production configuration, the attenuator should be set to as specific value. The default setting of “Automatic” is very useful during the development phases where hardware and tuing changes are frequent. However, in the final configuration set **Calibration: ATTENUATOR_xx** to the highest value seen through the calibration process.

If the attenuator values recorded during manufacturing testing are not consistent, try to slightly change the corresponding **Calibration: BAL_TARGET_XX** to break the equality.

The general tuning procedure for **Calibration: BAL_TARGET_XX** is as follows:

1. If the average raw-count data is close to zero (see Figure 39) and the balancing IDAC is >min (see Table 22), leave the **Calibration: BAL_TARGET_XX** parameters at their default values.
2. While the balancing IDAC remains at its minimum value, decrement the relevant **Calibration: BAL_TARGET_XX** parameter.

6.4.4 Channel saturation

It is not normally necessary to check the analog channel for saturation. However, if unexpected behavior is detected, it can be useful to verify the channel usage. There are separate values for mutual-cap and self-cap, however, the process is the same.

Have the display on with any typical image. Test with both nothing on the panel, and with as much of the panel covered as can be easily accomplished.

Use the View Variable tool to view the min and max ADC outputs (see Table 23). The minimum value should not be less than negative half the TX period parameter (e.g., **TSS: TX_PERIOD_PANEL**). The maximum value should not be more than half the TX period parameter.

Table 23 Raw ADC variable RAM addresses

Variable Name	6165X RAM Address*	8165X RAM Address*	8168X RAM Address*	Size	Signed
nm_min_sc_rx	0x200015F2	0x20004250	0x20004250	2	Yes
nm_max_sc_rx	0x200015F4	0x20004252	0x20004252	2	Yes
nm_min_sc_tx	0x200015F6	0x20004254	0x20004254	2	Yes
nm_max_sc_tx	0x200015F8	0x20004256	0x20004256	2	Yes
nm_min_mc	0x200015FA	0x20004258	0x20004258	2	Yes
nm_max_mc	0x200015FC	0x2000425A	0x2000425A	2	Yes
* RAM addresses are FW build dependent (1.5.1093522, 1.5.1093522, and 1.4.1084642 respectively)					

If the channel range is too high, increase the relevant attenuator setting. If the attenuator is set to Automatic, check its current value under **View** > **Tool Windows** > **IDAC Data** (see Figure 38).

Repeat the test with the noisiest image to be supported. If the channel range is too high, trade-off between:

- Do not support artificially high-noise images.
- Increase the attenuator setting. However, this will reduce the touch sensitivity for all images.
- Enable charger armor and review the burst noise feature (see Section 8.10). However, this can make the panel unresponsive.

6.4.5 Parameters

Table 24 Calibration verification parameters

Configurable Parameter	Description	Selection
TSS: RX_CAP_SEL CYAT6165/8165 (TSG6L) only	Note CYAT6165/8165 (TSG6L) only Size of the integration capacitor in pico-farads. Do not change from the default.	2.5 pF / 5 pF / 7.5 pF / 10 pF / 12.5 pF / 15 pF / 17.5 pF / 20 pF (default = 20 pF)
TSS: GIDAC_MULT	This parameter specifies the output current range for the HW baseline's global IDAC.	1x / 2x (default = 1x)
TSS: HW_BL_GIDAC_LSB_CONFIG	Specifies the HW baseline's global IDAC LSB.	160 nA / 320 nA (default = 320 nA)
Calibration: INT_VOLTAGE_MC **Calibration: INT_VOLTAGE_SC** **Calibration: INT_VOLTAGE_MC_BTN** **Calibration: INT_VOLTAGE_SC_BTN**	Also called **Calibration: INT_VOLTAGE_xx**. These parameter sets the target integration voltage swing for sensing. The value is specified in mV. Do not change from the default.	10 – 650 (default = 250)
Calibration: BAL_TARGET_MC **Calibration: BAL_TARGET_SC**	Also called **Calibration: BAL_TARGET_xx**. These parameters set the balancing target	6 – 121 BAL_TARGET_SC

Configurable Parameter	Description	Selection
Calibration: BAL_TARGET_MC_BTN	in terms of percentage of the TX_PERIOD. The value is specified in percentage and is scaled to the integer 128 (i.e. 50% = 64).	(default = 64) All others (default = 38)
Calibration: BAL_TARGET_SC_BTN		
Calibration: MC_PWC_LIMIT_PERCENT	This parameter limits the baseline current injection period in terms of percentage of the TX_PERIOD. The value is specified in percentage and is scaled to the integer 128 (i.e. 50% = 64).	6 – 128 (default = 115)
Calibration: SC_PWC_LIMIT_PERCENT		6 – 128 (default = 64)
Calibration: ATTENUATOR_MC	Also called **Calibration: ATTENUATOR_xx**. Attenuator can be set by the calibration process, or manually fixed. Should only be manually fixed if the manufacturing process produces inconsistent attenuator values.	Auto, 3x, 4x, 4.8x, 6x, 8x, 12x (default = Auto)
Calibration: ATTENUATOR_SC_RX		
Calibration: ATTENUATOR_SC_RX		
Calibration: ATTENUATOR_BTN_MC		
Calibration: ATTENUATOR_BTN_SC		

6.5 TX scan tuning

6.5.1 TX pulses

Sensing very small capacitive touch sensors for small fluctuations requires significant resolution and noise performance. The RX channel is specifically designed to do this by integrating charge over several pulses to improve the sensitivity of the channel, increasing the resolution as well as the SNR. The signal transfer (relative to the out of band transfer (noise)) grows significantly with the number of pulses. Likewise, with increasing pulses, the bandwidth of the channel narrows, focusing sensitivity around the sample rate.

Pulses	Relative Signal to Noise Transfer (db)	Bandwidth at Different Sample Rates (kHz)			
		120kHz	200kHz	250kHz	320kHz
4	16.27	30.00	50.00	62.50	80.00
8	22.40	15.00	25.00	31.25	40.00
12	25.94	10.00	16.67	20.83	26.67
16	28.45	7.50	12.50	15.63	20.00
24	31.97	5.00	8.33	10.42	13.33
32	34.48	3.75	6.25	7.81	10.00
48	38.00	2.50	4.17	5.21	6.67
64	40.50	1.88	3.13	3.91	5.00

Figure 44 Transfer characteristics

Note that increasing the number of pulses is, however, not without some tradeoff in performance. Integrating more pulses requires more time and more power. When setting the TX pulses, it is recommended to also consider the project's refresh rate (Section 6.5.2) and Power (Section 8.4) requirements.

For initial bring up of the touch solution, it is recommended to integrate with no less than 16 pulses. This can be refined later when noise is introduced.

Also for initial bring up, it is recommended to disable glove support by setting the parameter **Touch Mode: TOUCHMODE_CONFIG** to "FingerOnly". Otherwise, the controller will switch between normal and glove scanning parameters in a way that may seem unpredictable. It is best to tune for fingers first, then enable gloves and optimize the glove tuning after.

6.5.2 Refresh rate

There are three different refresh rates:

- Active – when there is a touch on the panel.
- Look-for-touch (LFT) – no touch detected and either low-power mode is disabled or the touch timeout has not expired (after the most recent liftoff).
- Low-power – low power mode enabled and no touch detected for greater than the touch timeout.

The LFT and low-power refresh rates do not need tuning. Simply set them to the required refresh period.

6.5.2.1 Defining active refresh rate

In the active state, scanning and data processing are pipelined (performed in parallel). Therefore, the refresh rate is the longest of:

- Scan Time (see Figure 45)
- Processing Time
- Minimum Refresh Interval (**Device Setup: ACT_INTRVL0**) (see Figure 46)

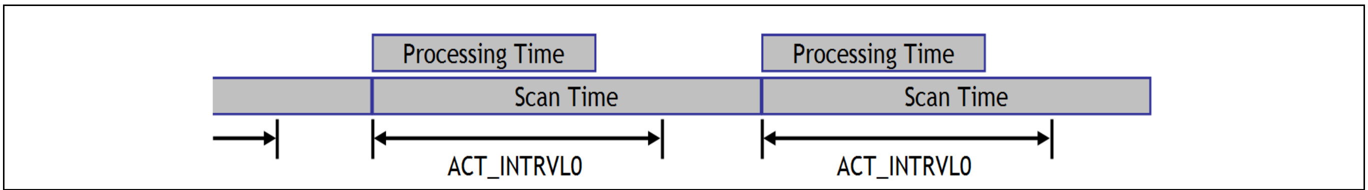


Figure 45 Active mode mutual-cap scanning operation (refresh rate = 1/scan time)

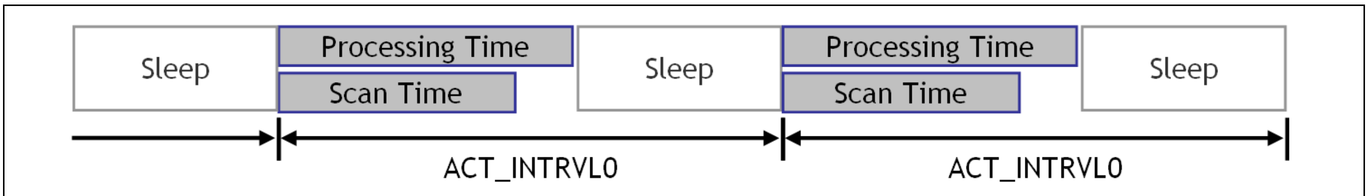


Figure 46 Active mode mutual-cap scanning operation (refresh rate = 1/ACT_INTRVL0)

The scan time, similar to all touchscreen tuning, is heavily configuration dependent. First identify the types of scan that constitute the full scan time:

Table 25 Active mode scan time

Fingers: ACT_LFT_EN	Scan Filtering: WATER_REJ_ENABLED	Scan Time (ST)
Disabled	Disabled	Mutual-cap ST
Disabled	Enabled	Mutual-cap + Self-cap ST
Enabled	Disabled	Approx. Mutual-cap ST (small occasional increase from self-cap baseline)
Enabled	Enabled	Mutual-cap + Self-cap ST

For each scan method, the scan time is the product of the number of RX slots, the TX's to scan, the TX pulses, and the pulse time. For mutual-cap, if the MTX feature is enabled (see Equation 9 in Section 3.3.5), the number of TX scanned is the number of MTX-patterns, which may be larger.

- RX_Slots: View > Project Configuration Wizard... > Next > “Slot Assignment” tab
- SelfCap_Slots: **Device Setup: SELF_Z_MODE** defines the slots used in the “Slot Assignment” tab
- TX_Patterns: See Equation 9 in Section 3.3.6 (Multi-phase TX order)
- TX_Pulses_x: **TSS: TX_PULSES_MC** and **TSS: TX_PULSES_SC**
- Pulse_Time_x:
  - If spreader disabled: **TSS: TX_PERIOD_MC** and **TSS: TX_PERIOD_SC**
  - If spreader enabled (for TSG6XL mutual-cap): **TSS: TX_PERIOD_MC** + Avg_Sprd
    - Avg_Sprd = ((**TSS: MC_TX_SPREADER_PULSES** – 1) * (**TSS: MC_TX_SPREADER_STEP** * 2) / 2)

$$\text{Scan Time}_{MC} = \text{RX\_Slots} * \text{TX\_Patterns} * \text{TX\_Pulses\_MC} * \text{Pulse\_Time\_MC}$$

Equation 22

$$\text{Scan Time}_{SC} = \text{SelfCap\_Slots} * \text{TX\_Pulses\_SC} * \text{Pulse\_Time\_SC}$$

Equation 23

Note that if charger armor is active (**Charger Armor: CHARGER_ARMOR_ENABLE** = Enabled), additional noise-detection scans will be made. This can have a small effect on the overall panel scan rate.

6.5.2.2 Tuning active refresh rate

There are many parameters that control the active refresh rate for a given scan type. However, most of these are usually fixed or have very limited scope, leaving only one tunable parameter. The main parameters are described here:

- Number of TX-patterns – FIXED
  - Initially the number of TX-sensors
  - This value could increase if MTX is used and the number of TX is not an exact divisor
- Refresh Rate – FIXED
  - A project normally has a fixed target refresh rate that must be met
- TX Period – LIMITED
  - Minimum period is set by the hardware sensor RC value
  - EMI/EMC emissions and display noise will severely limit the available selections
- TX Pulses – CONFIGURABLE
  - Set as high as possible to maximize SNR
  - Increasing TX pulses will increase the refresh time
  - Increasing TX pulses will increase the power consumption

The only consideration when tuning the number of TX pulses, is whether the project has a power requirement. Most automotive projects do not, so simply set the TX pulses as high as possible without violating the refresh rate requirement. If power is a project requirement, then set the TX pulses as high as possible without violating either the power or refresh rate requirements.

6.5.3 Response time

The response time should also be considered when defining the refresh rate. Typically, the response time can be defined as a minimum of 1x the active refresh rate, and a maximum of 3x the active refresh rate.

Precise calculations depend on the configuration of ACT_LFT, MTX_ORDER, and the location of the touch. The simplest case is with ACT_LFT disabled, MTX_ORDER=1, and the touch occurring in the middle of the panel. In this case, the response time would be a linear distribution from 1.5x to 2.5x the active refresh rate (see rows 3 and 5 in Table 26), with the variable being when the touch occurred in relation to the current TX being scanned.

Slightly more complex (but keeping MTX-Order=1) would result in the following minimum and maximum response time estimations:

Table 26 Response time timings

Min/ Max	ACT_LFT	Touch Location	Total Time*	Before Detected	Detect Scan	Locate Scan	Process Scan
Min	Disabled	Bottom	11ms	0	0	1	Active
Min	Enabled	Bottom	21ms	0	1	Active	Active
Min	Disabled	Middle	15ms	0	0	Active/2	Active
Min	Enabled	Middle	22ms	0	LFT/2	Active	Active
Max	Disabled	Middle	25ms	Active/2	0	Active	Active
Max	Enabled	Middle	26ms	LFT/2	LFT	Active	Active
Max	Disabled	Top	30ms	Active	0	Active	Active
Max	Enabled	Top	28ms	LFT	LFT	Active	Active

* The “Total Time” is an estimate and assumes:

- LFT scan rate of 4 ms
- Active scan rate of 10 ms
- Minimum time to complete a scan detecting a touch at the bottom of the panel is 1 ms

6.5.4 Discard time

For touch panels with larger capacitive load, a longer settling time may be required when the RX channel is first connected. This is because the RX channel only resets at the beginning of each conversion and is not reset between TX pulses. Between pulses, the settling error is accumulated and can behave similarly to random noise in the self-cap scan.

TSS: DISCARD_TIME specifies the time in micro-seconds at the beginning of each conversion to ignore the measured data. The device intentionally ignores the initial measurement to prevent accumulating the RX channel settling error. This value applies to both mutual-cap and self-cap scan.

$$\text{Ignored Pulses} = \text{rounddown}\left(\frac{24 \cdot \text{TSS: DISCARD\_TIME}}{\text{TSS: TX\_PERIOD\_MC (or SC)}}\right)$$

Equation 24

For example, with **TSS: TX_PERIOD_SC** set to 270, the TX Period is about 11.25 µs. If the discard time is set to 130 µs, the first 11 TX pulses will be ignored in each conversion. If **TSS: TX_PULSES_ SC** is set to 64, only the measurements from the last 53 self-cap TX pulses are used.

If the RX channel is set up as described in Section 6.3.2, then the panel should be settled within one pulse. The time for each pulse can be calculated as follows:

$$\text{MC (or SC) TX Period Time } (\mu s) = \text{rounddown}\left(\frac{\text{TSS: TX\_PERIOD\_MC (or SC)}}{24}\right)$$

Equation 25

The discard time can be calculated more precisely using the following process:

1. Remove all noise sources (display and charger noise)
2. Set **Charger Armor: CHARGER_ARMOR_ENABLE** to Enabled
3. Prevent charger armor being activated by setting the thresholds artificially high, for example:

- **Charger Armor: WB_THRESH** = 1000
- **Charger Armor: NMF_DETECT_THRESH** = 255
- **Charger Armor: NMI_THRESH** = 32767

4. Setup analog for noise capture:

- **Charger Armor: WB_CMF_ENABLE** = Disabled
- **TSS: GIDAC_MULT** = 1x
- **TSS: NM_WB_IDAC** = 12

5. Enable noise capture:

- **Device Setup: REPORT_CFG: Metrics Output Mode** = Noise
- **Device Setup: REPORT_CFG: Noise Water metrics output enable** = Enabled

6. Open the listener window and observe the wideband noise. In TTHE, select View, Tool Windows, Listener. Make sure that the Wideband Noise checkbox is selected.
7. Initialize **TSS: DISCARD_TIME** to 0
8. Increment discard time starting at 1½ the pulse time (see Equation 25), and then in steps equal to the pulse time
9. Observe the wideband noise as the discard time is increased
10. Optimal discard time is when the wideband noise almost stops decreasing

6.5.5 Parameters

Table 27 TX scan parameters

Configurable Parameter	Description	Selection
TSS: DISCARD_TIME	This parameter sets the number of discarded TX periods DISCARD_TIME = 24 / TX_PERIOD. This is to allow for panel settling. The value is specified in µs.	0 – 255 (default = 30)
TSS: TX_PERIOD_MC	This parameter defines the number of system clocks in each TX half period for mutual-cap or self-cap scans. The "BTN" parameters are for button scans.	68 – 1000 (default = 235)
TSS: TX_PERIOD_SC		48/68 – 1000 (default = 236)

Configurable Parameter	Description	Selection
TSS: TX_PERIOD_BTN_MC	Where stated "48/68", the minimum value is 48 for TSG6L devices and 68 for TSG6XL devices.	48/68 – 1000 (default = 235)
TSS: TX_PERIOD_BTN_SC		48/68 – 1000 (default = 236)
TSS: TX_PULSES_MC	This parameter defines the number of TX pulses per conversion for scans of each type for all known object types. The "BTN" parameters are for button scans, and the "GLOVE" parameters are for glove scans.	1 – 255 (default = 20)
TSS: TX_PULSES_SC		1 – 255 (default = 64)
TSS: TX_PULSES_BTN_MC		1 – 255 (default = 64)
TSS: TX_PULSES_BTN_SC		1 – 255 (default = 32)
TSS: TX_PULSES_GLOVE_MC		0 – 255 (default = 36)
TSS: TX_PULSES_GLOVE_SC		0 – 255 (default = 130)
TSS: TX_PULSES_BTN_GLOVE_MC		0 – 255 (default = 64)
TSS: TX_PULSES_BTN_GLOVE_SC		0 – 255 (default = 32)
TSS: TX_SPREADER_STEP_NUM CYAT6165/8165 (TSG6L) only	Defines the number of pulses used for 1 modulation cycle. TSG6XL only, set to 1 to disable the TX spreader.	4 steps / 8 steps (default = 4 steps)
TSS: MC_TX_SPREADER_PULSES CYAT8168 (TSG6XL) only		1 – 12 (default = 1)
TSS: SC_TX_SPREADER_PULSES CYAT8168 (TSG6XL) only		
TSS: TX_SPREADER_STEP CYAT6165/8165 (TSG6L) only	Number of system clocks for stepping width applied to each TX phase. Each subsequent TX period will be elongated by **twice** this value. Set to zero to disable the TX spreader.	0 – 100 (default = 0)
TSS: MC_TX_SPREADER_STEP CYAT8168 (TSG6XL) only		
TSS: SC_TX_SPREADER_STEP CYAT8168 (TSG6XL) only		
Device Setup: SELF_Z_MODE	Determines a set of sensors which are used for self-cap scan.	Self Cap RX / Self Cap TX / Self Cap TX & RX (default = Self Cap RX)

Configurable Parameter	Description	Selection
Device Setup: ACT_INTRVL0	Enables or disables self-cap scanning in look-for-touch mode. When enabled (and water rejection is disabled), self-cap scanning is only performed in LFT mode (and once every MAX_SELF_SCAN_INTERVAL).	Enabled / Disabled (default = Disabled)

6.6 Signal scaling

The touch processing system is designed to comfortably support signal resolutions up to 12-bit. With no thick glove requirement, it is recommended to tune all mutual-cap and self-cap scaling factors to achieve approximately a 2000 count change (also known as diff-count) with a typical 9 mm finger.

1. In TTHE Heat Map display mode, set DataType to "DiffCounts".
2. Center a typical-sized grounded 9 mm metal finger over the highest touch intensity (HTI) location, see Figure 47.
3. Configure the parameter **TSS: SCALE_FACT_MC** such that the mutual-cap diff-count is about 2000 counts. Repeat for **TSS: SCALE_FACT_BTN_MC** if buttons are used.
4. Configure the parameter **TSS: SCALE_FACT_SC** such that the self-cap diff-count is about 2000 counts. Repeat for **TSS: SCALE_FACT_BTN_SC** if buttons are used.
5. If thick glove is to be supported, follow the instructions in Section 9.3.3.2 to achieve diff-counts of at least 100 with the thickest supported glove.

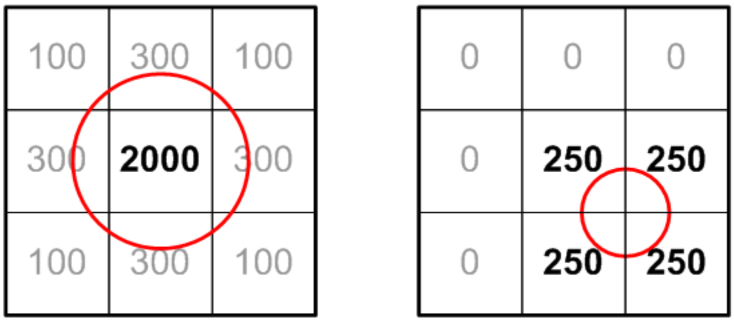


Figure 47 HTI (left) and LTI (right) regions

6.6.1 Parameters

Table 28 Signal scaling parameters

Configurable Parameter	Description	Selection
TSS: SCALE_FACT_MC	This parameter scales mutual-cap signals. This scaling is applied just prior to touch processing.	0 – 1000 (default = 300)
TSS: SCALE_FACT_SC	This parameter scales self-cap signals. This scaling is applied just prior to touch processing.	0 – 1000 (default = 300)
TSS: SCALE_FACT_BTN_MC	This parameter scales mutual-cap signals for buttons. This scaling is applied just prior to button processing.	0 – 1000 (default = 200)
TSS: SCALE_FACT_BTN_SC	This parameter scales self-cap signals for buttons. This scaling is applied just prior to button processing.	0 – 1000 (default = 200)

7 Basic touch processing tuning

Basic raw and difference data should now be generated, ready for detecting fingers. This section discusses tuning the touch solution to report some fingers.

7.1 Mutual-cap finger detection criteria

Touch detection requires a couple of parameter definitions. The algorithms use values called the Peak, Z9-sum, and Z8-sum. The peak is the highest value in a touch zone. The Z9-sum is the sum of the peak and the 8 surrounding sensors. The Z8-sum is the sum of the peak's 8 surrounding sensors, but not including the peak. In Figure 48 the peaks are 2000 and 250, the Z9-sums are 3600 and 1000, and the Z8-sums are 1600 and 750.

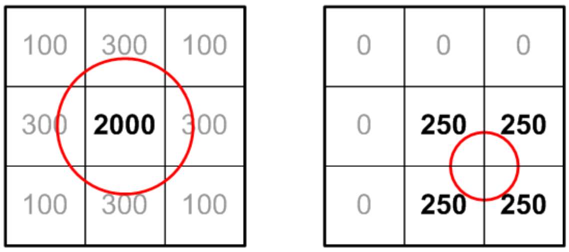


Figure 48 Defining Peak, Z9-sum, and Z8-sum

A finger touch is reported to the host when the following conditions are met:

- The touch threshold is met:
  - For a new finger touch on the panel, use **Fingers: FINGER_THRESH_MUT_HI** (or **Gloves: GLOVES_THRESH_MUT_HI**).
  - For an ongoing finger touch on the panel, use **Fingers: FINGER_THRESH_MUT_LO** (or **Gloves: GLOVES_THRESH_MUT_LO**).

$$Z_{peak} > \text{xxx\_THRESH\_MUT\_xx}$$

Equation 26

- The absolute mass of the finger is sufficiently large to be recognized as a finger:
  - For a new finger touch on the panel, use **Fingers: FINGER_THRESH_MUT_HI** (or **Gloves: GLOVES_THRESH_MUT_HI**).
  - For an ongoing finger touch on the panel, use **Fingers: FINGER_THRESH_MUT_LO** (or **Gloves: GLOVES_THRESH_MUT_LO**).

$$\text{Z9-sum} > \text{xxx\_THRESH\_MUT\_xx} * \text{xxx\_Z9\_FILT\_SCALE (panel core)}$$

Equation 27

$$\text{Z9-sum} > \text{xxx\_THRESH\_MUT\_xx} * \text{xxx\_Z9\_FILT\_SCALE} /2 \text{ (panel edge)}$$

Equation 28

$$\text{Z9-sum} > \text{xxx\_THRESH\_MUT\_xx} * \text{xxx\_Z9\_FILT\_SCALE} / 4 \text{ (panel corner)}$$

Equation 29

- The relative mass of the finger is sufficiently large to be recognized as a finger.

$$\text{Z8-sum} > Z_{peak} * \text{xxx\_Z8\_FILT\_SCALE (panel core)}$$

Equation 30

$$\text{Z8-sum} > Z_{peak} * \text{xxx\_Z8\_FILT\_SCALE} / 2 \text{ (panel edge)}$$

Equation 31

$$\text{Z8-sum} > Z_{peak} * \text{xx\_Z8\_FILT\_SCALE} / 4 \text{ (panel corner)}$$

Equation 32

Note that for a slider, the finger is never considered to be in the panel core. The Z9 sum is a Z3 sum. However, the **Fingers: FINGER_Z9_FILT_SCALE** can still be tuned the same as for a full touchscreen. The first equation (for the core area) is just never used.

For finger touches at the panel edge, a 2x3 grid is used to determine mass. The mass of a touch at the corner of the touch screen will be calculated using a 2x2 grid of sensors. When a touch is at the edge or corner of a panel, the result is scaled according to Equation 28 and Equation 29.

7.2 Mutual-cap finger threshold tuning

The finger threshold is set using the diff-count at the least touch intensity (LTI) location, see Figure 47. And the mass (often referred to as the Z9 sum) is the sum of the diff-counts of all nine sensors in a 3x3 grid. Note that the finger threshold should be tuned using the smallest supported finger or the lightest touch. Use the following procedure to determine the finger threshold.

1. Use TTHE Heat Map display mode, set DataType to “DiffCounts”.
2. Disable the wet finger tracking by setting **Wet Finger: WF_ENABLE** to Disabled to ensure that the small finger will not be misinterpreted as a water droplet and get rejected.
3. Place the smallest supported grounded metal finger over the least touch intensity (LTI) location as shown in Figure 47.
4. Set **Fingers: FINGER_THRESH_MUT_HI** to 80% of the LTI peak.
5. Set **Fingers: FINGER_THRESH_MUT_LO** to 70% of the LTI peak.

7.3 Mutual-cap Zx_Sum

The **Fingers: FINGER_Z9_FILT_SCALE** (or **Gloves: GLOVES_Z9_FILT_SCALE**) and **Fingers: FINGER_Z8_FILT_SCALE** (or **Gloves: GLOVES_Z8_FILT_SCALE**) values ensure that the detected object is of the correct proportions. For most projects, these values can be left at their defaults. Compare these values (using the smallest supported grounded metal finger over the least touch intensity (LTI) location) with the following equations (confirm these values are acceptable through empirical testing):

$$\text{xxx\_Z9\_FILT\_SCALE} = \frac{0.8 \cdot \text{xxx\_Z9\_Sum}}{\text{xxx\_THRESH\_MUT\_HI}}$$

Equation 33

$$\text{xxx\_Z8\_FILT\_SCALE} = \frac{0.8 \cdot \text{xxx\_Z8\_Sum}}{\text{xxx\_THRESH\_MUT\_HI}}$$

Equation 34

7.4 Self-cap finger threshold

There are two self-cap finger thresholds.

Firstly, if the active look-for-touch (LFT) feature is enabled (**Fingers: ACT_LFT_EN**), then the parameter **Touch Mode: TOUCHMODE_LFT_SELF_THRSH** determines when the touch controller transitions from the self-cap LFT state into the mutual-cap LFT state. If the active-LFLT feature is not enabled, the LFT state is always the mutual-cap LFT state. Note that if the periodic mutual-cap scan (for baseline maintenance) detects a valid touch, this will also trigger the state to change from self-cap LFT to mutual-cap LFT.

Secondly, when in the mutual-cap LFT state and water rejection is enabled (**Scan Filtering: WATER_REJ_ENABLE**), a touch is only reported to the host if many conditions are true. One of these conditions is that the self-cap signalpeak sensor value must be greater than the threshold **Fingers: FINGER_THRESH_SELF** (or **Gloves: GLOVES_THRESH_SELF**). Self-cap signal is summed across the active sensors (see Section 8.11.1 for more details). For example, if the water rejection sensor width (**Scan Filtering: WATER_REJ_SNS_WIDTH**) is 0, only the self-cap sensor on the same row/column as the mutual-cap peak is used.

$$\mathtt{Z_{peak}} > xxx\mathtt{\_THRESH\_SELF} > \mathtt{Touch\ \ Mode{:}\,TOUCHMODE\_LFT\_SELF\_THRSH}$$

Equation 35

The finger thresholds are found by measuring the diff-count values at the least touch intensity (LTI) location, see Figure 47. Use the following procedure to determine initial values for the finger thresholds.

1. In Heat Map display mode, set DataType to “DiffCounts” and ScanType to “Self”.
2. Center the smallest supported grounded metal finger over the least touch intensity (LTI) location, as shown in Figure 47.
3. Take 80% of the average of the two adjacent self-cap sensor values. If self-cap data is acquired on both axes, then use the minimum for setting the TX or RX threshold.

$$\text{xxx\_THRESH\_SELF} = 0.8 \cdot \frac{\text{X[n]+X[n+1]}}{2}$$

Equation 36

$$\text{Touch Mode: TOUCHMODE\_LFT\_SELF\_THRSH} = 0.8 \cdot \text{xxx\_THRESH\_SELF}$$

Equation 37

7.5 Z-value scaling

Use the largest finger supported by the project being tuned. If there is no requirement, then a 24mm to 30mm finger is reasonable. Tune the parameter **Fingers: FINGER_Z_SCALE** (or **Gloves: GLOVES_Z_SCALE**) such that a Z-value of approximately 200 is reported.

7.6 Parameters

Table 29 Signal scaling parameters

Configurable Parameter	Description	Selection
Fingers: FINGER_THRESH_MUT_HI	This parameter is the threshold for mutual-cap data used to identify a finger touchdown.	0 – 32767 (default = 490)
Fingers: FINGER_THRESH_MUT_LO	This parameter is the threshold for mutual-cap data used to identify a finger liftoff.	0 – 32767 (default = 400)
Fingers: FINGER_THRESH_SELF	This parameter is the threshold for self-cap data used to identify a finger touchdown.	0 – 32767 (default = 124)
Fingers: FINGER_Z9_FILT_SCALE	Noise rejection 3x3 signal sum filter scale for detecting a finger touch. The 3x3 signal sum must be larger than the product of FINGER_THRESH_MUTUAL_ON and this parameter.	0 – 4 (default = 2)
Fingers: FINGER_Z_SCALE	Factor to scale the Z value for normal finger. Adjust this value to achieve 255 Z counts for the maximum finger size. Rarely changed from default value.	0 – 65535 (default = 450)
Buttons: BTN_LS_ON_THRSH_MUT_N	Button mutual-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 80)
Buttons: BTN_LS_OFF_THRSH_MUT_N	Button mutual-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 70)
Buttons: BTN_LS_ON_THRSH_SELF_N	Button self-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 65)
Buttons: BTN_LS_OFF_THRSH_SELF_N	Button self-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 40)
Buttons: BTN_HS_ON_THRSH_MUT_N	Button mutual-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 20)
Buttons: BTN_HS_OFF_THRSH_MUT_N	Button mutual-cap scan liftoff threshold for High Sensitivity Mode (Glove)for button N.	5 – 32767 (default = 10)
Buttons: BTN_HS_ON_THRSH_SELF_N	Button self-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 30)
Buttons: BTN_HS_OFF_THRSH_SELF_N	Button self-cap scan liftoff threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 10)

Configurable Parameter	Description	Selection
Gloves: GLOVES_THRESH_MUT_HI	This parameter is the threshold for mutual-cap data used to identify a glove touchdown.	0 – 32767 (default = 180)
Gloves: GLOVES_THRESH_MUT_LO	This parameter is the threshold for mutual-cap data used to identify a glove liftoff.	0 – 32767 (default = 150)
Gloves: GLOVES_THRESH_SELF	Self-cap Gloves Threshold.	0 - 32767 (default = 65)
Gloves: GLOVES_Z9_FILT_SCALE	Z9 filter scale for normal gloves when Charger Armor is not active. Peak with Z9<(scale*GLOVES_THRESH_MUT_LO) will be removed.	0 - 4 (default = 4)
Gloves: GLOVES_Z8_FILT_SCALE	Z8 filter scale for gloves detection. A valid touch must satisfy this requirement: Z8 Sum > Glove Peak Diff-count * GLOVES_Z8_FILT_SCALE.	0 - 127 (default = 2)
Gloves: GLOVES_Z_SCALE	Factor to scale the Z value for glove. Adjust this value to achieve 255 Z counts for the maximum glove size.	0 - 65535 (default = 150)

8 Advanced touch processing tuning

8.1 Background: common noisy displays

Display noise can interfere with the capacitive touchscreen sensing. AC-VCOM is one of the most common types of displays that cause interference. AC stands for alternating current. VCOM refers to a common conductive layer in the display that is held at a particular voltage to bias the liquid crystal material. Putting the terms together, AC-VCOM means an AC voltage waveform is applied to the common conductive layer. AC-VCOM displays have two common characteristics:

- Strong fundamental period (square-wave-like) in the 10 kHz to 30 kHz frequency range.
- Constant (or almost constant) amplitude/period with changes in the display image.

Figure 49 shows a captured waveform of a real AC-VCOM display. The measurement is performed by covering the display surface with a conductive material (copper tape or plate) and probing the conductive material with an oscilloscope.

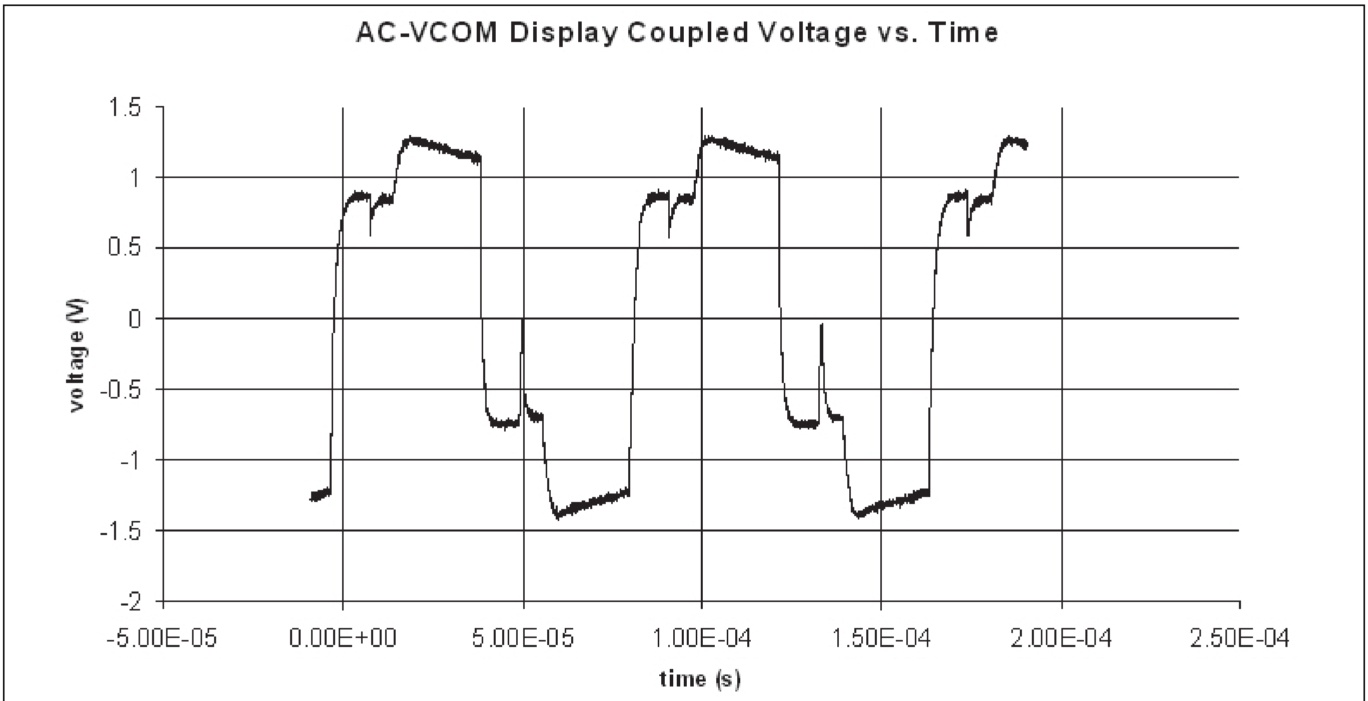


Figure 49 Example AC-VCOM noise waveform captured from a real display

Dot inversion is another common type of display that can cause interference. Dot inversion displays have several common characteristics:

- Series of short noise pulses that represent display RGB pixel signals and repeat in a pattern between 30 kHz and 80 kHz (typical).
- Short pulses with changing magnitude and/or polarity depending on display image.

Figure 50 shows the noise waveform of a real dot inversion display measured at the display surface.

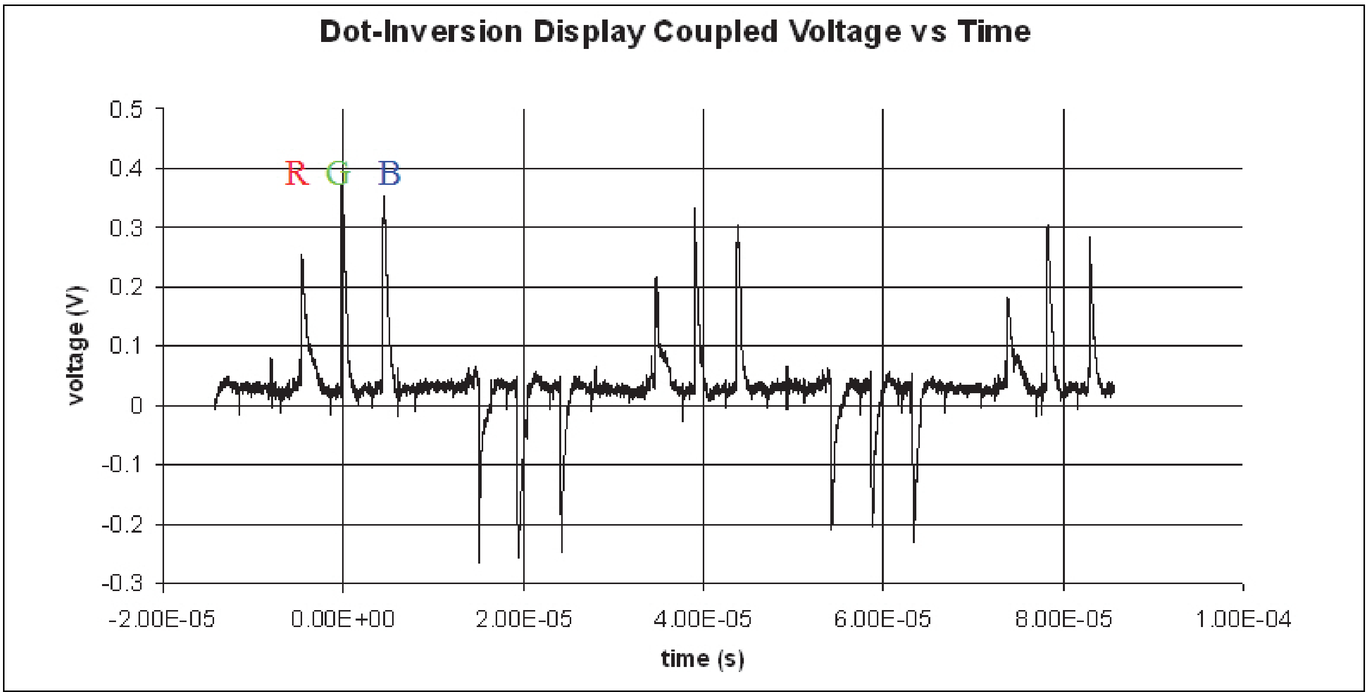


Figure 50 Example dot-inversion display noise waveform captured from a real display

8.2 Noise filter tuning

8.2.1 Display noise

Display noise is the switching noise capacitively coupled from the surface of the display to the conductive touchscreen material. There are four ways to mitigate the display noise:

- Use longer integration time to average out the noise impact and to reduce the width of the channel passband frequency.
- Configure the TX operating frequency such that the receiver channel passband is outside the display noise spectrum.
- Enable the common mode digital filter (CMF) to dynamically remove the display noise that is correlated to multiple channels.
- Enable the on-chip filters, most effective when the display noise frequency is less than 1/5 of the TX operating frequency.

8.2.2 Ground sensors between scans (CYAT8168X (TSG6XL) only)

In the CYAT8168X (TSG6XL) devices only, there is a parameter that allows the sensors to be grounded between scans (**TSS: TSS_GROUND_SENSORS**). This is only applicable if the touch controller is not scanning at the fastest possible rate. Enabling this feature will reduce the display emissions as the grounded sensors will act as a shield. However, to accomplish this, the touch controller must stay awake, increasing the power consumption.

It is recommended to leave this feature disabled. The feature should only be enabled if display emissions are a significant problem.

8.2.3 Display noise tuning procedure

Follow these steps to tune for display noise. In all cases, make sure that the panel is calibrated in a quiet environment – for example, with the display disabled or displaying an all-black image. Attempting to tune noise with a calibration that is already degraded with noise may not work well.

Before starting, disable all advanced features like charger armor, wet finger tracking, or dynamic calibration.

The test images to use should contain the noisiest images available. Examples are: White, 1-pixel chess, sub-pixel-chess, 1-pixel vertical stripes, and 1-pixel horizontal stripes.

8.2.3.1 Display noise tuning – Common Mode Filter (CMF)

Significant display noise can usually be mitigated by using the common mode filter (CMF). The CMF looks at the RX channel results and subtracts the common value from them. It is especially useful in cases where a large amount of noise is present along an entire TX axis (typical for display noise).

There are 2 filters. The basic filter uses all data to calculate the CMF's "common value". The advanced filter only uses values not part of a touch, which is determined by a diff-count threshold compared to the last scan's diff-counts. The advanced filter is supported in SW. CYAT8168 (TSG6XL) also support the basic filter in HW (CYAT8165 (TSG6L) devices have no HW CMF filter).

For low levels of display noise, use the basic CMF filter (or advanced filter with "diff" threshold maximized). Set the CMF "raw" threshold to less than 25% of the maximum finger touch diff-count (or slightly higher if this is close to the detected display noise). A larger threshold can result in multiple touches skewing the common value enough for valid touches to be attenuated.

For higher levels of display noise, use the advanced CMF filter. Set the CMF "raw" threshold to the maximum (32,767). Initially, set the CMF "diff" threshold to 25% of the maximum finger touch diff-count (it may need to be changed, depending on empirical testing).

CMF parameters are configured independently for the scan object (panel and button), scan type (finger and glove), the scan method (mutual-cap and self-cap), and the presence of noise (charger armor). The combinations are shown in the following table:

Table 30 CMF parameters

Scan Object	Scan Method	Filter Enable	Filter Thresholds
Panel	Mutual	**MC_RAW_FILTER_MASK: CMF Filter**	**MC_RAW_CMF_THRESH** **MC_DIFF_CMF_THRESH**
Panel	Self	**SC_RAW_FILTER_MASK: CMF Filter**	**SC_RAW_CMF_THRESH** **SC_DIFF_CMF_THRESH**
Button	Mutual	**BTN_MC_RAW_FILTER_MASK: CMF Filter**	**BTN_MC_RAW_CMF_THRESH** **BTN_MC_DIFF_CMF_THRESH**
Button	Self	**BTN_SC_RAW_FILTER_MASK: CMF Filter**	**BTN_SC_RAW_CMF_THRESH** **BTN_SC_DIFF_CMF_THRESH**
Glove	Mutual	**GLOVE_MC_RAW_FILTER_MASK: CMF Filter**	**GLOVE_MC_RAW_CMF_THRESH** **GLOVE_MC_DIFF_CMF_THRESH**
Glove	Self	**GLOVE_SC_RAW_FILTER_MASK: CMF Filter**	**GLOVE_SC_RAW_CMF_THRESH** **GLOVE_SC_DIFF_CMF_THRESH**

Scan Object	Scan Method	Filter Enable	Filter Thresholds
Btn+Glv	Mutual	**BTN_GLOVE_MC_RAW_FILTER_MASK: CMF Filter**	**BTN_GLOVE_MC_RAW_CMF_THRESH** **BTN_GLOVE_MC_DIFF_CMF_THRESH**
Btn+Glv	Self	**BTN_GLOVE_SC_RAW_FILTER_MASK: CMF Filter**	**BTN_GLOVE_SC_RAW_CMF_THRESH** **BTN_GLOVE_SC_DIFF_CMF_THRESH**
CA+Pan	Mutual	**CA_MC_RAW_FILTER_MASK: CMF Filter**	**CA_MC_RAW_CMF_THRESH** **CA_MC_DIFF_CMF_THRESH**
CA+Btn	Mutual	**CA_BTN_MC_RAW_FILTER_MASK: CMF Filter**	**CA_BTN_MC_RAW_CMF_THRESH** **CA_BTN_MC_DIFF_CMF_THRESH**

The following process uses mutual-cap parameters; however, the same process is applicable to the parameters for all other modes.

1. Disable all filters **Raw Processing: MC_RAW_FILTER_MASK: xxx** (glove may need disabling).
2. Set **Raw Processing: CMF_TYPE** to "Default" (CYAT8168X (TSG6XL) only).
3. Set Display Mode to "Heat Map" and Display Settings DataType to RawPostFilter and SensorValueType to "Max - Min".

- Inject the display noise.
- Record the maximum value for both MC and SC after about 10 seconds.

4. Repeat the experiment in step-3 with all test images:

- Try different images, including high-noise images (such as sub-pixel chess, vertical 1-pixel bars, and horizontal 1-pixel bars).

5. Select the CMF filter to use, by comparing the maximum noise measured to 25% of the maximum touch diff-count (for CYAT8165 (TSG6L), always use advanced CMF):

- If max noise is less, use the basic CMF (CYAT8168X (TSG6XL) only):
  a) Set **Raw Processing: CMF_TYPE** to "Default"
  b) Set the "raw" CMF threshold to about 20% more than the maximum noise measured. If max noise is greater, use the advanced CMF:
- If max noise is greater, use the advanced CMF:
  a) (CYAT8168X (TSG6XL) only) Set **Raw Processing: CMF_TYPE** to "Software"
  b) Set the "raw" CMF threshold to the maximum (32,767)
  c) Initially, set the "diff" CMF threshold to 25% of the maximum finger touch diff-count (may need to be changed, depending on empirical testing)

6. Re-enable the features disabled in step-1.

If the common mode filter is able to reach the target SNR in Section 2.7, display noise tuning is complete. Otherwise proceed to the next section to evaluate the display noise frequency spectrum and to determine an alternate TX frequency to avoid the display noise impact.

8.2.3.2 Random noise tuning - IIR

The raw data IIR filter can also be very helpful in removing noise. IIR parameters are configured independently for the scan object (finger, glove, button, and glove button), the scan method (mutual-cap and self-cap), and the presence of noise (charger armor).

Use the following process to set the IIR threshold. The process is scan type independent. Use this process for mutual-cap, self-cap (unless using "standard" SC), combined with the panel, slider, and glove scans.

1. In the TTHE:
- Disable the raw data IIR filter parameters (CMF should be enabled and tuned first).
- Select Heat Map display mode.
- In Touch Display, set DataType to RawPostFilter and SensorValueType to "Max - Min".
2. Inject the display noise:
- Try different images, including high-noise images (such as sub-pixel chess, vertical 1-pixel bars, and horizontal 1-pixel bars).
- Record the maximum value for both MC and SC after at least 10 seconds.
3. Repeat the test for multiple display images.
4. Set the IIR threshold to about 20% more than the maximum noise measured.

The IIR coefficient can usually be left at the default. However, for high noise systems, reducing the IIR coefficient can be a useful method of increasing the SNR. However, there are significant tradeoffs for glove and small fingers. Increased IIR filtering reduces fast response for touch detection and swiping.

8.2.3.3 Noise tuning – advanced tuning options

If the CMF filter is unable to achieve the required SNR, more aggressive methods must be used. Evaluate the following suggestions to extend the integration time. Longer integration time reduces the RX channel passband in frequency domain, minimizing sensitivity to the injected noise. The goal is to increase the integration time as much as possible while still meeting the target refresh rate.

- Enable Multi-TX for mutual-cap panel scan (**TSS: MTX_ORDER**), see Section 3.3.5.
- Increase the number of TX pulses per conversion (**TSS: TX_PULSES_MC** and **TSS: TX_PULSES_SC**).
- Review the TX frequency selection to avoid the display's noise frequencies:
  - Refer to Section 6.3.6 to measure the display's noise spectrum.
  - Select a TX frequency that is in a quiet part of the spectrum.
  - Note that changing the TX frequency affects SNR, panel scan time, and EMI/EMC emissions, so take care in modifying it.

8.2.4 Parameters

Table 31 Noise filter tuning parameters

Configurable Parameter	Description	Selection
TSS: MTX_ORDER	This parameter defines the multi-phase TX order.	1 – 19 (default = 4)
TSS: TX_PULSES_MC	This parameter defines the number of TX pulses per conversion for scans of each type for all known object types.	1 – 255 (default = 20)
TSS: TX_PULSES_SC		1 – 255 (default = 64)

Configurable Parameter	Description	Selection
TSS: TSS_GROUND_SENSORS CYAT8168X (TSG6XL) only	Note: CYAT8168X (TSG6XL) ONLY. When enabled, all sensing pins are grounded between scans. Helps reduce display emissions at the cost of higher power-consumption.	Enabled / Disabled (default = Disabled)
Raw Processing: xxx_RAW_FILTER_MASK: CMF Filter	CMF filter for raw-counts enabled/disabled. Applies only to mutual-cap raw-counts.	Enabled / Disabled (default = Enabled)
Raw Processing: xxx_RAW_FILTER_MASK: IIR Filter	IIR filter for raw-counts enabled/disabled. Applies only to mutual-cap raw-counts.	Enabled / Disabled (default = Enabled)
Raw Processing: xxx_RAW_IIR_COEF	Determines IIR filter order for mutual-cap in base state.	1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Raw Processing: xxx_RAW_IIR_THRESH	IIR threshold for mutual-cap in base state	0 – 32767 (default = 100)
Raw Processing: xxx_RAW_CMF_THRESH	CMF threshold for mutual-cap to determine whether a sensor should be included in the CMF base value calculation.	0 – 32767 (default = 200)
Raw Processing: xxx_DIFF_CMF_THRESH CYAT8168X (TSG6XL) only	CYAT8168X (TSG6XL) only, and only when the software CMF is selected. CMF diff count threshold (from the previous scan) to determine whether a sensor should be included in the CMF base value calculation (also needs to pass RAW threshold criteria).	0 – 32767 (default = 32767)
Raw Processing: CMF_TYPE CYAT8168X (TSG6XL) only	CYAT8168X (TSG6XL) only. Selection between the HW (Default) and SW (Software) CMF algorithm.	Default / Software (default = Default)

8.3 Display synchronization

This method of display noise mitigation should only be used if it was originally planned and/or all other display noise mitigation methods have been exhausted. If a frequency cannot be found using standard tuning techniques (see Section 6) that allows both EMI/EMC emissions to pass and display noise to be acceptable, then display synchronization can be considered.

Note that hardware support is required, so early planning is important. The display driver must provide the synchronization signal(s) (usually HSync and/or VSync) to a touch controller GPIO.

Note that the spreader should be disabled when display synchronization is used.

8.3.1 Supported methods

The touch controller supports three methods of display synchronization (see Figure 51). The method is selected using the **TSS: SYNC_CONTROL** parameter. The scan-time taken for each synchronization method is dependent on the TX waveform parameters, however, rough estimates are given for guidance. The three synchronization methods are:

1. Sync TX Period
   a) Sync each TX-pulse
   b) Scans about 10 µs – 30 µs (TX-period)
   c) Sync signal is usually the pixel-clock, HSync, or a custom sync signal
   d) Number of sync-signals to scan entire panel is usually: TX-pulses * TX-sensors * RX-slots
2. Sync TX Pattern
- Sync one full TX waveform
- Scans about 200 µs – 500 µs (usually TX-period * TX-pulses)
- Sync signal is usually either the HSync or a custom sync signal
- Number of sync-signals to scan entire panel is usually: TX-sensors * RX-slots
3. Sync Scan Start:
- Sync the whole panel scan
- Scans about 10 ms (usually TX-period * TX-pulses * TX-sensors * RX-slots)
- Sync signal is usually either the VSync or a custom sync signal
- Number of sync-signals to scan entire panel is: 1

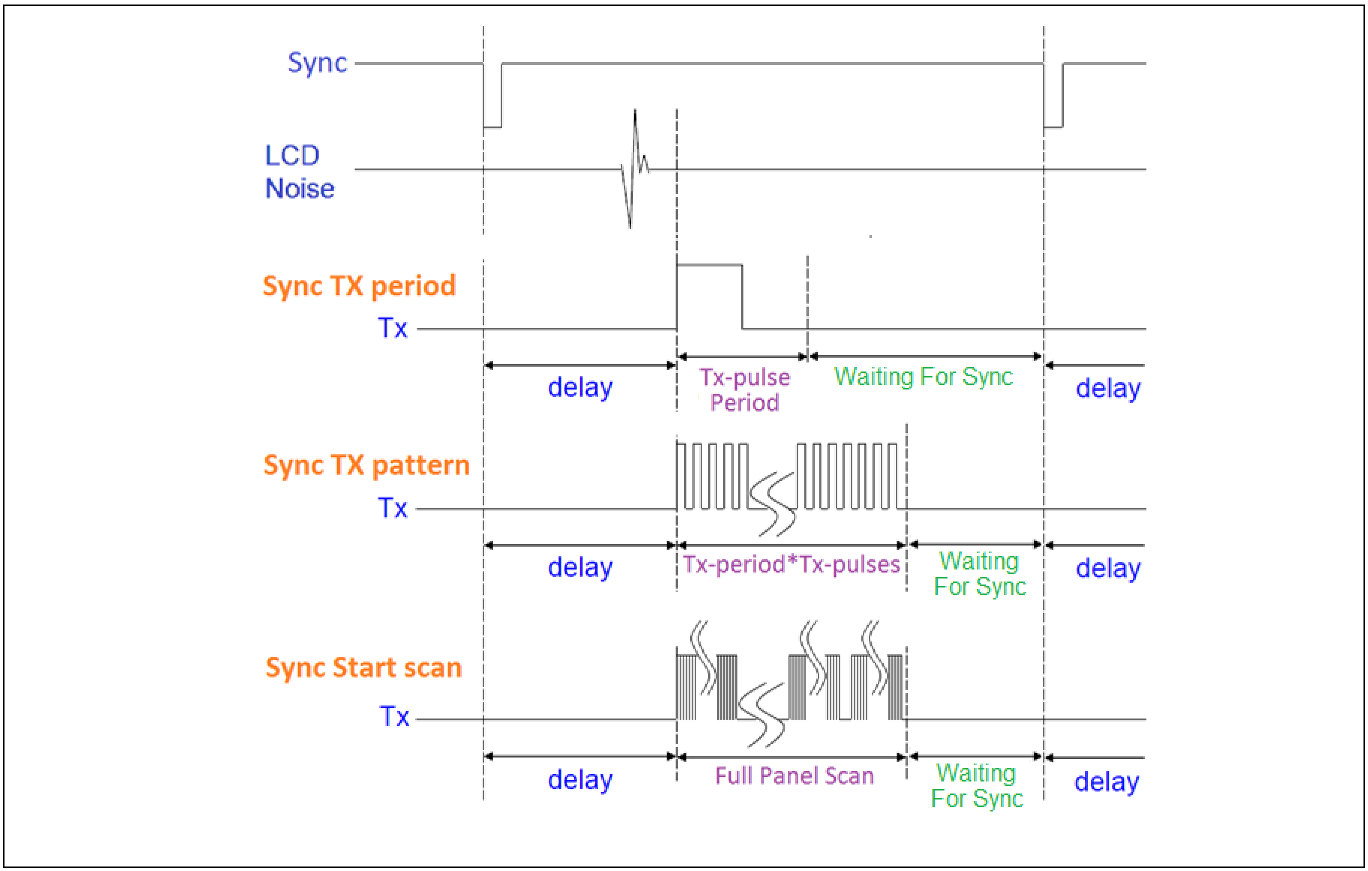


Figure 51 Display synchronization methods

8.3.2 Measuring display noise

Use the following procedure to measure display noise:

1. Use a display module that is not laminated to the touchscreen so that you can directly access the surface of the display.
2. Turn on the display module and display a white image.
3. Cover the display with a copper plate to capacitively couple the noise waveform to the copper plate (see Figure 52).
4. Connect an oscilloscope (10x probe) to the copper plate and capture the noise generated by the display (see Figure 53). The oscilloscope ground should be connected to the display ground.
5. It is also helpful to capture the display timing signals (for example, VSync, HSync, Pixel-clock)

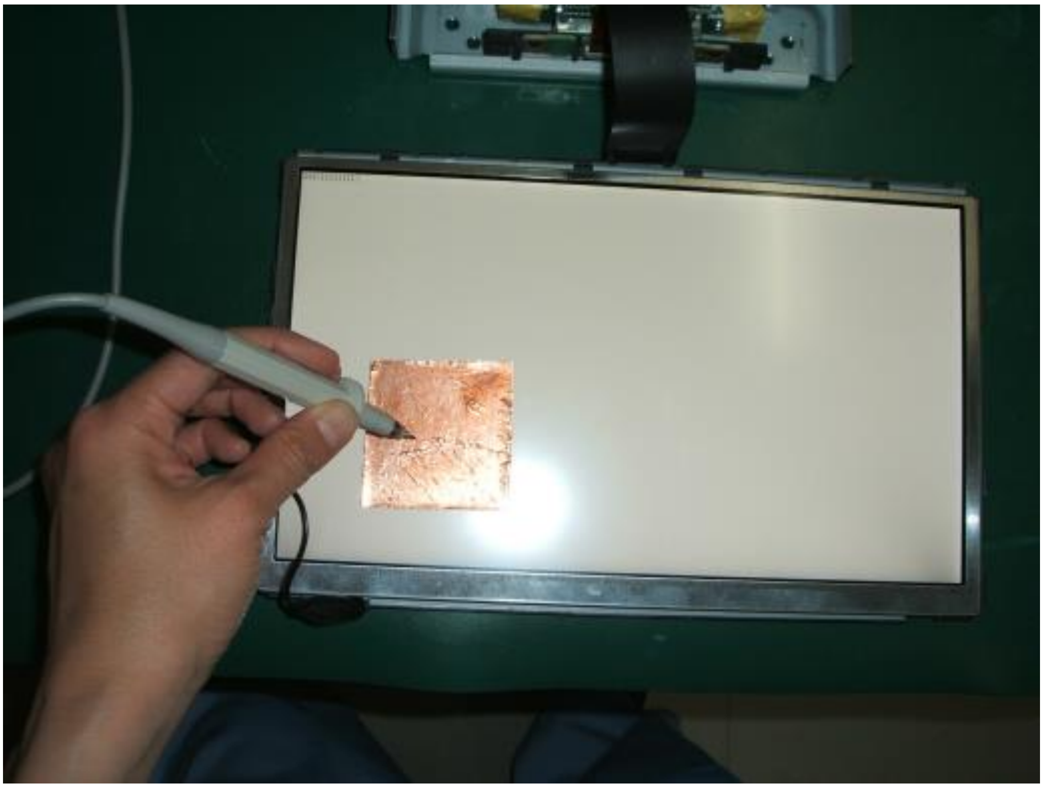


Figure 52 Measuring display noise

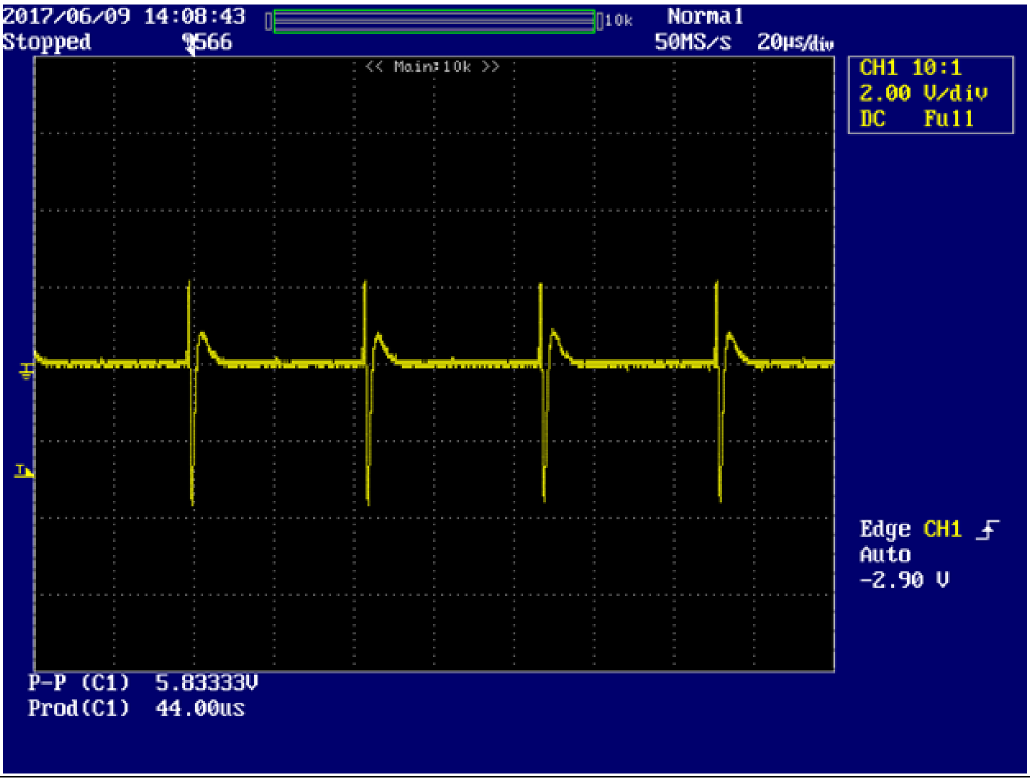


Figure 53 Display noise measurement result

8.3.3 Analyze display synchronization

Firstly, analyze the display noise in the time domain (see Figure 53). Identify the quiet periods, their duration, and relationship to the display timing signals. It is not always necessary to complete the full touch panel scan in the quiet periods, making the noise “regular” can make it easier for the touch controller to filter out.

For example, the following screenshot (Figure 54) shows similar display noise to Figure 53, but with the addition of the display-driver’s HSync signal. The HSync signal has a period of 13.4 µs (~75 kHz). To achieve one TX-pulse every HSync period, the TX frequency must be more than 75 kHz, but cannot exceed the maximum frequency supported by the panel (see the discussion on panel speed in Section 6.3.2.

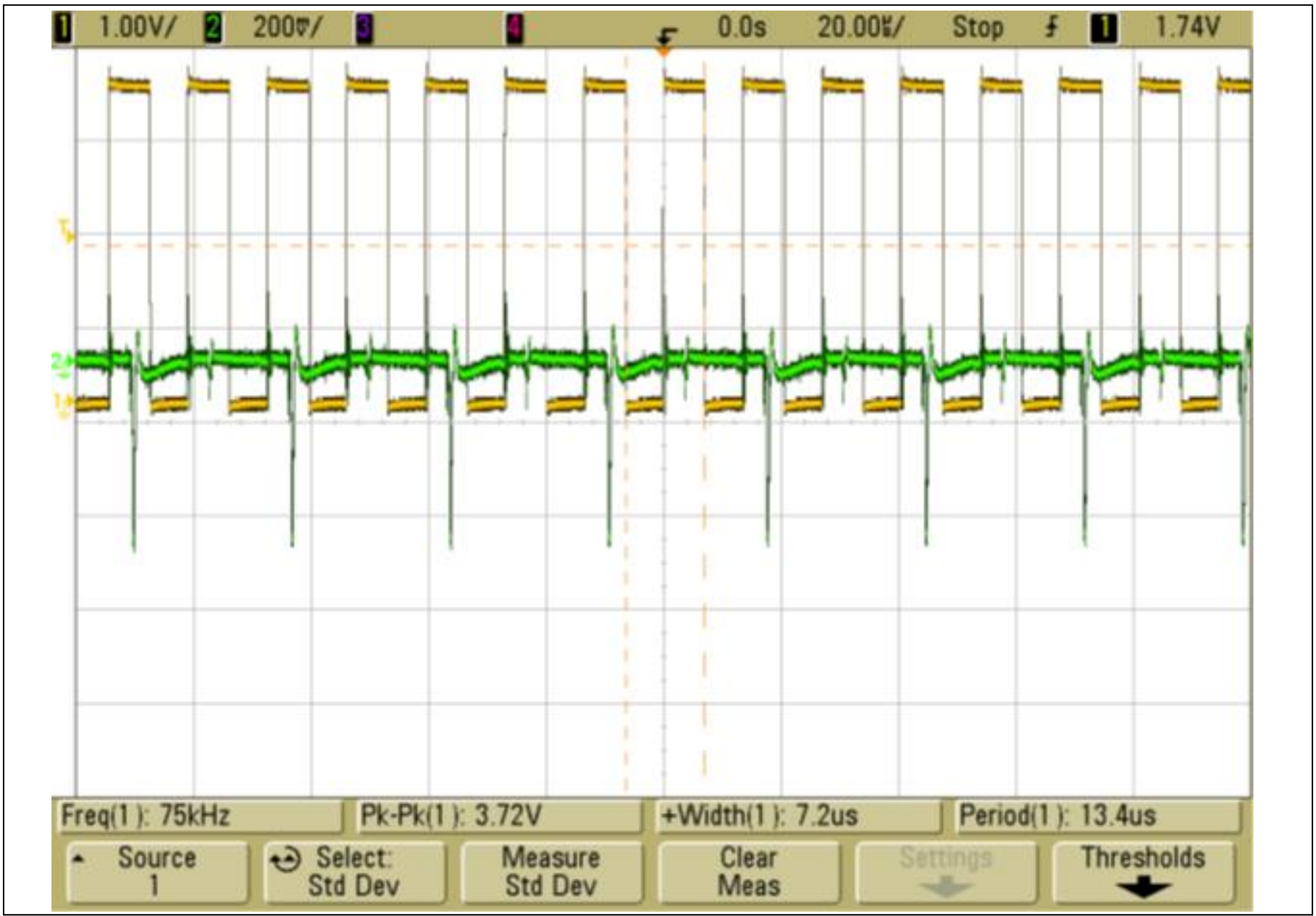


Figure 54 Display noise analysis - noise with sync signal

Figure 54 shows two big noise spikes; these should be avoided. There is a quiet zone of about 9 µs starting just before the falling-edge of the sync signal. Therefore, starting each TX pulse on the sync signal's falling edge would be a good configuration. The TX-period should be less than 6.7 µs (half-sync-period or 149 kHz) so both rising and falling edges would occur within the low-period of the sync signal (quiet zone).

The display synchronization is shown in Figure 55 and Figure 56. The TX pulses are synchronized to the falling-edge of the sync signal (**TSS: PQ_CTRL: POLARITY** = INVERTED). Note that there is a short delay (about 2.5 µs) between the synchronization trigger and the TX falling-edge. This is defined by the parameter **TSS: PQ_CTRL3**. The TX falling-edges can be clearly seen, but the ends of the TX pulses are less clear. A TX pulse is clearly shown by the red "TX period" delimiters in Figure 56. The idle period before the next sync signal is shown in Figure 56 in green.

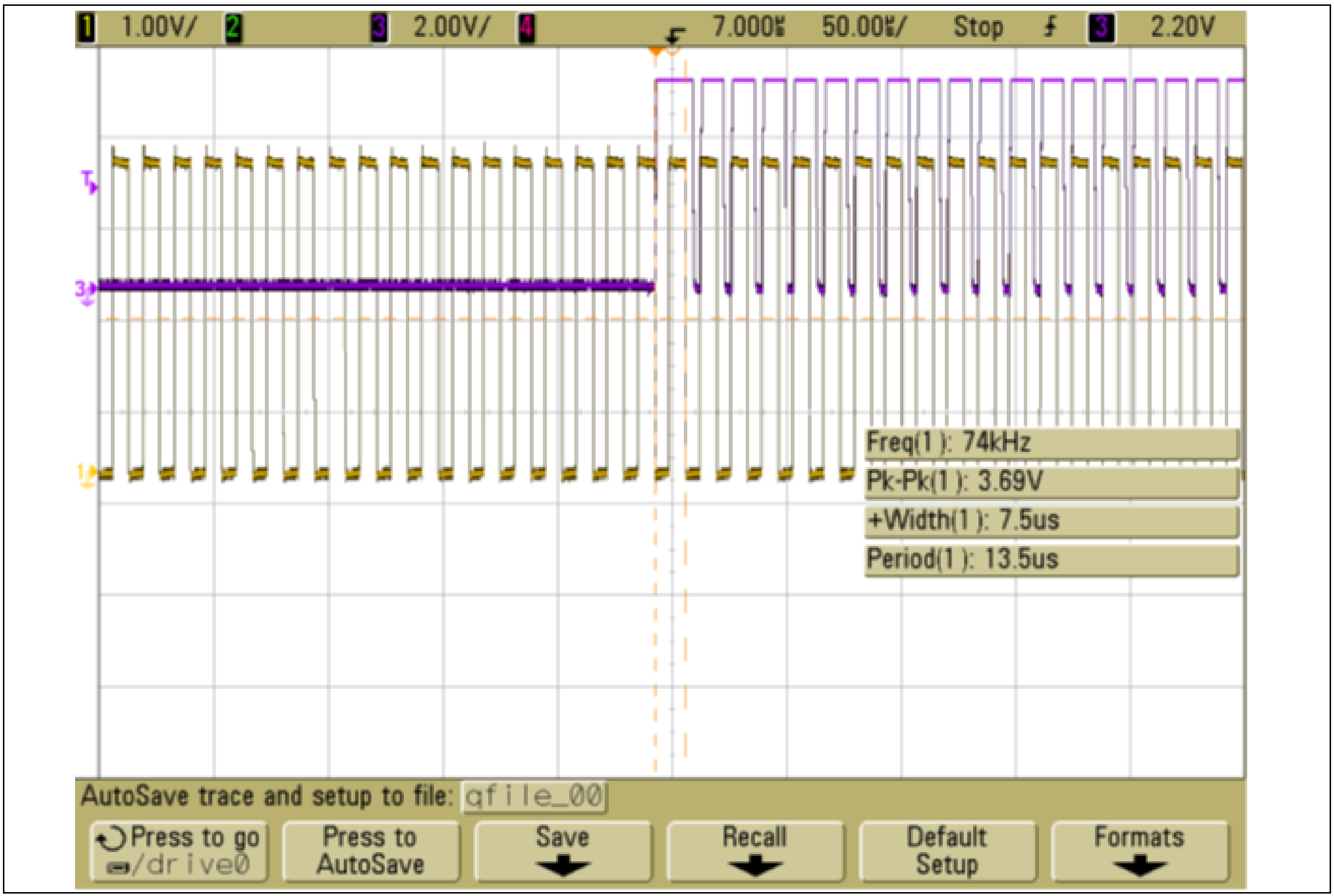


Figure 55 Display noise analysis - sync signal with TX

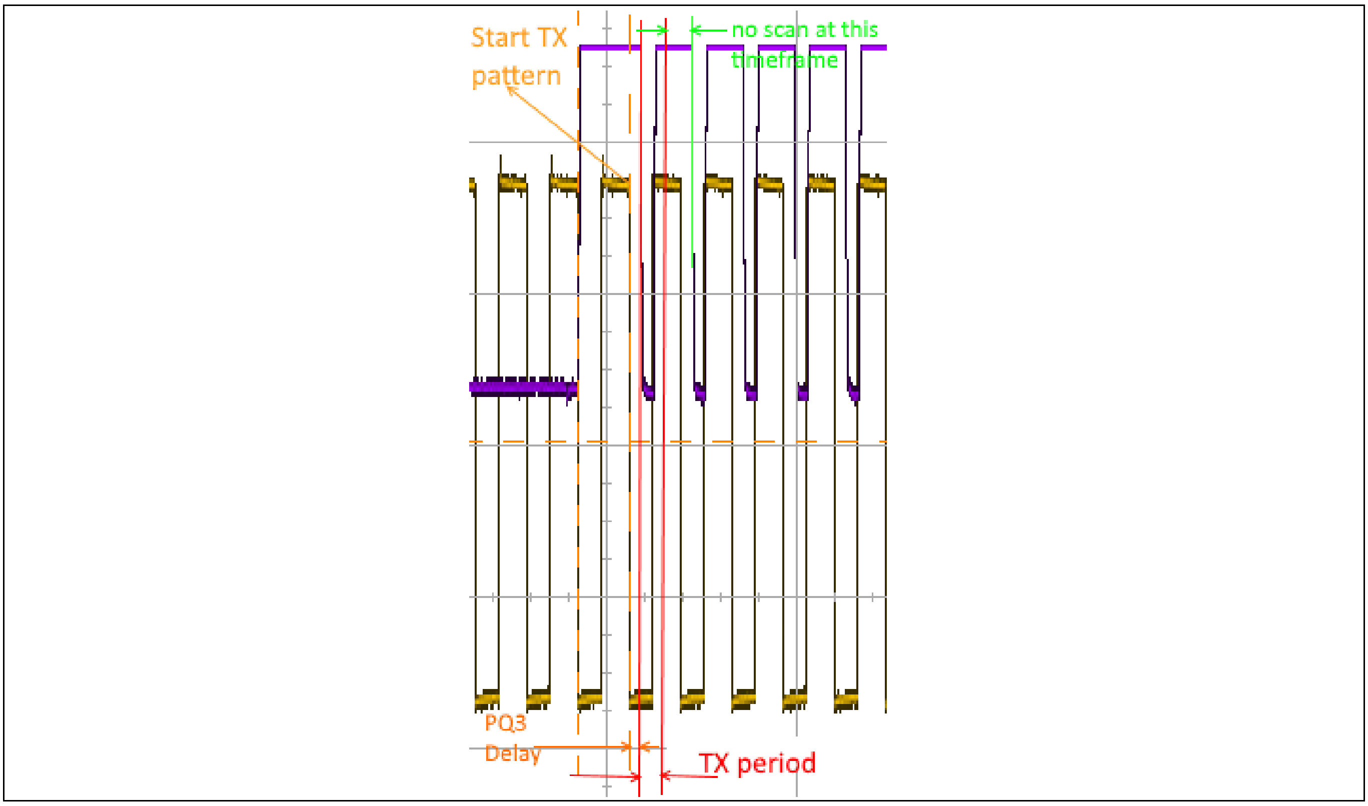


Figure 56 Display noise analysis – TX period sync start

8.3.4 Configuring display synchronization

First capture the display noise (in the time domain) and the available synchronization signals, as described in Section 8.3.2. Next analyze the display noise and based on the quiet periods identified, select a display synchronization method to use (see Section 8.3.3 and Figure 51).

Enable display synchronization with default settings:

- Ensure the spreader is disabled by setting the number of spread-steps to 1:
  - **TSS: TX_SPREADER_STEP_NUM** in CYAT8165 (TSG6L)
  - **TSS: MC_TX_SPREADER_PULSES / SC_TX_SPREADER_PULSES** in CYAT8168 (TSG6XL)
- Set the incoming synchronization source GPIO:
  - **Device Setup: EXT_SYNC** is set through the "Project Configuration Wizard"
- Set the polarity of the incoming synchronization pulse:
  - **TSS: PQ_CTRL: POLARITY** = NORMAL (active high)
- Enable the internal synchronization logic:
  - **TSS: PQ_CTRL: SRC_SEL** = Ext_Start_In
  - **TSS: PQ_CTRL: START_ENABLED** = ENABLED
  - **TSS: PQ_CTRL: PQ_ENABLED** = ENABLED
- Filter the incoming synchronization signal if it is noisy (default = 0 (no filtering)):
  - **TSS: PQ_CTRL2: ACT_RANGE** = 0
  - **TSS: PQ_CTRL2: DEACT_RANGE** = 0
- Set any delay from the sync trigger to the start of the TX waveform:
  - **TSS: PQ_CTRL3** = delay in units of the system clock (usually 48 MHz)

8.3.5 Triggering display synchronization

The touch controller will not always recognize a TX waveform synchronization event. If a synchronization trigger occurs when the touch controller is not waiting for it, the trigger is ignored. The next TX waveform will not start until a synchronization trigger is detected.

- A synchronization trigger will be ignored if it occurs while the touch controller is processing:
  - Delay period (defined by **TSS: PQ_CTRL3**) – see blue text in Figure 51
  - TX-waveform, depending on the synchronization method (single-TX-pulse, single-TX-pattern, or full-panel-scan) – see purple text in Figure 51
- A synchronization trigger will be processed if it occurs while the touch controller is processing:
  - Waiting for the next trigger event – see green text in Figure 51.

8.3.6 Examples

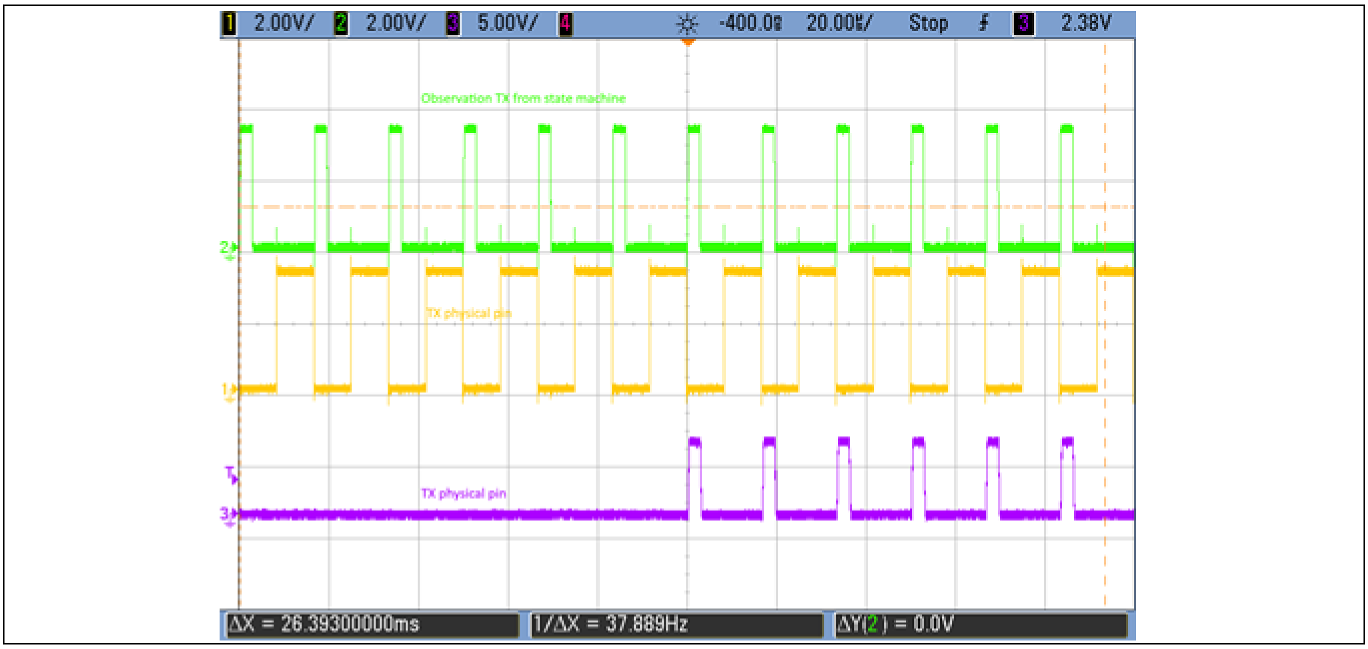


Figure 57 TSS: PQ_CTRL: POLARITY = INVERTED, TSS: PQ_CTRL3 = 0, MTX order positive phase

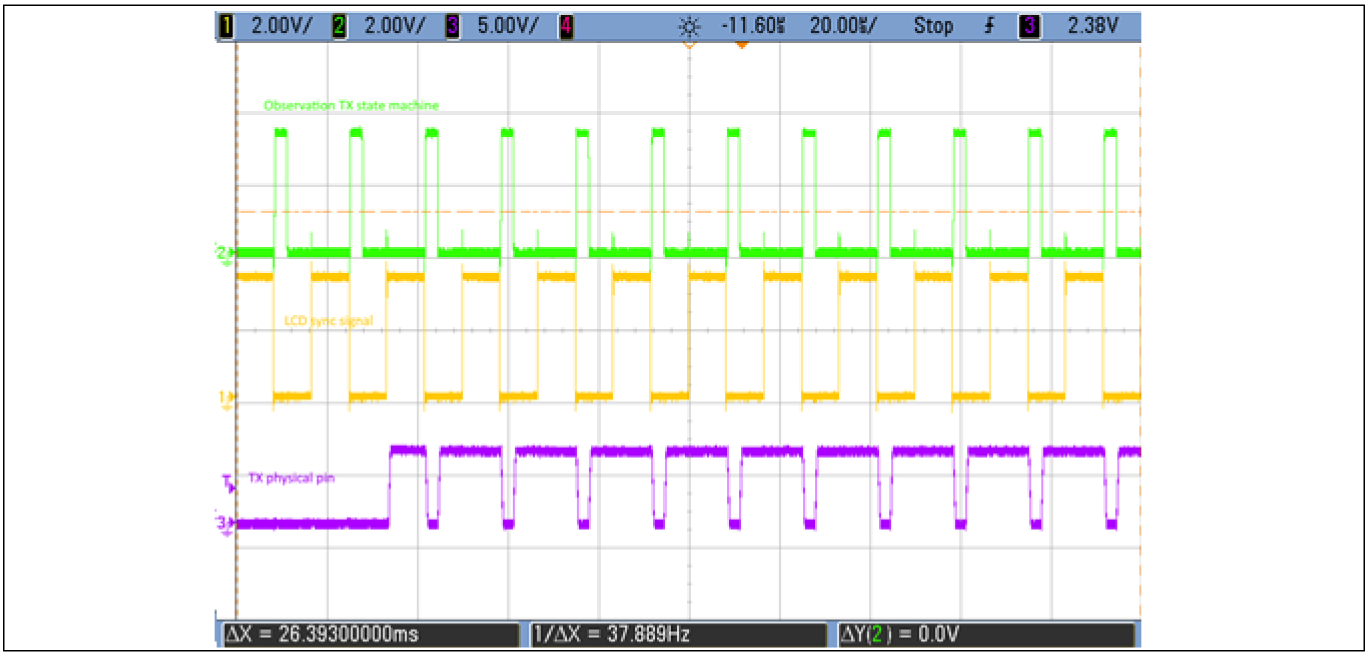


Figure 58 TSS: PQ_CTRL: POLARITY = INVERTED, TSS: PQ_CTRL3 = 0, MTX order negative phase

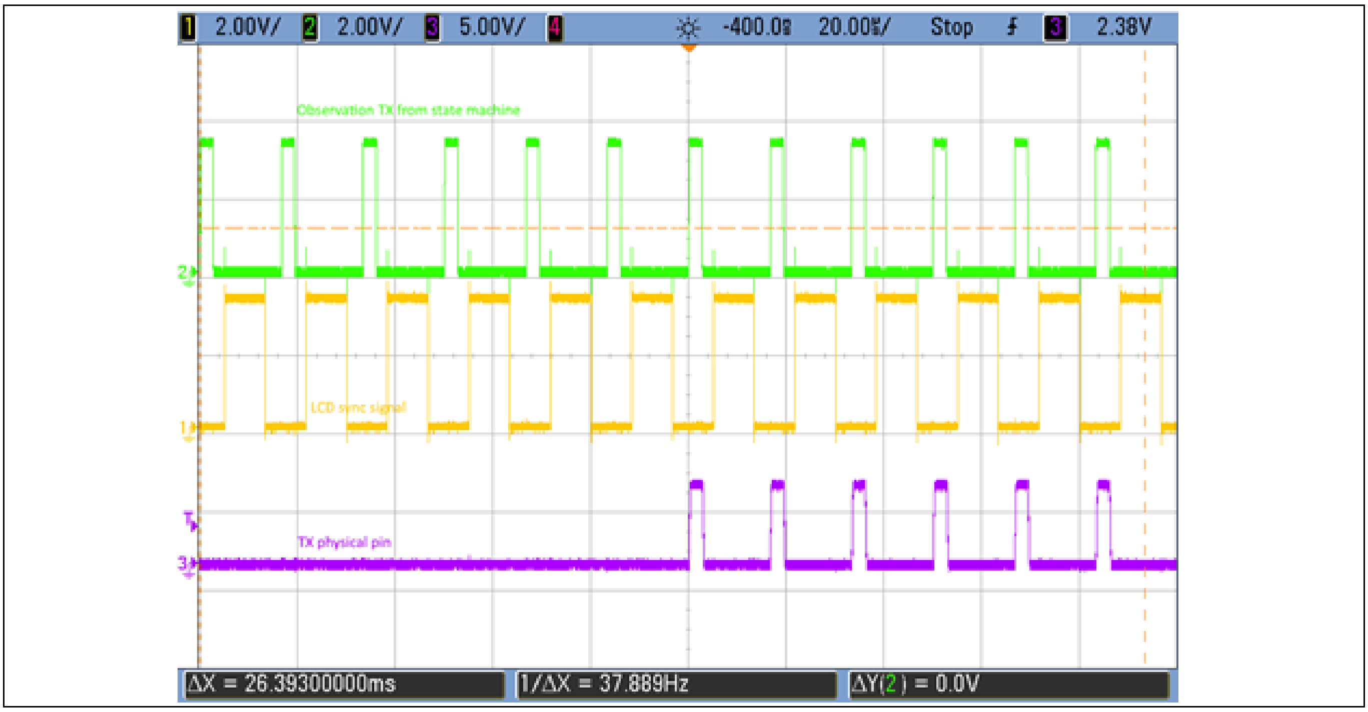


Figure 59 TSS: PQ_CTRL: POLARITY = INVERTED, TSS: PQ_CTRL3 = 150, MTX order positive phase

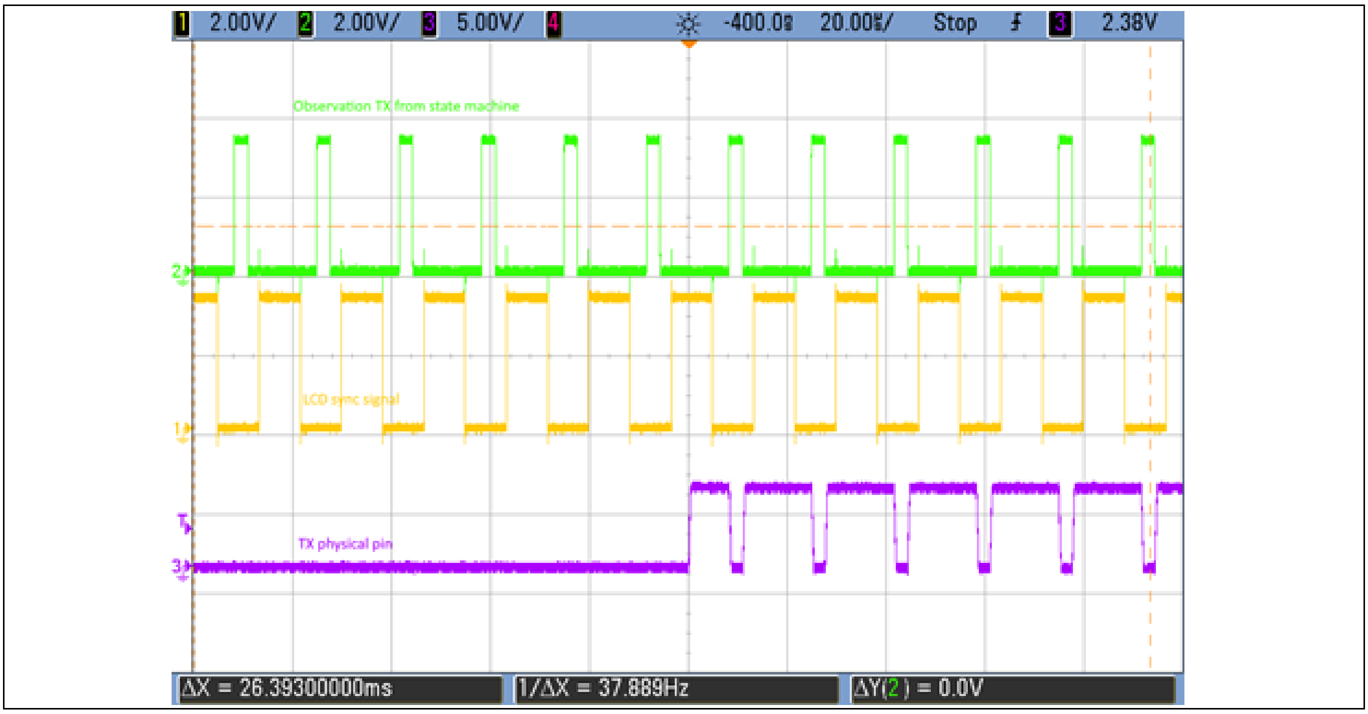


Figure 60 TSS: PQ_CTRL: POLARITY = INVERTED, TSS: PQ_CTRL3 = 150, MTX order negative phase

8.3.7 Parameters

Table 32 Display synchronization tuning parameters

Configurable Parameter	Description	Selection
Device Setup: EXT_SYNC	Note that P1[0] is only an option on TSG6L devices. GPIO to connect the display synchronization signal. This parameter can only be set using the "Project Configuration Wizard".	Disabled, P1[0] (CYAT6165/8165 (TSG6L) only), P1[1], P1[2], P1[3] (default – Disabled)
TSS: PQ_CTRL: POLARITY	Define whether the synchronization signal's rising-edge (NORMAL) or falling-edge (INVERTED) is used as the trigger event.	NORMAL / INVERTED (default = NORMAL)
TSS: PQ_CTRL: SRC_SEL	Select where the TX waveform's start trigger is sourced. Only used if START_ENABLED and PQ_ENABLED are ENABLED. Should always be set to "Ext_Start_In", other options not supported by this tuning guide.	Ext_Start_In, MUX_DATA.DATA_IN[0:6] (default = Ext_Start_In)
TSS: PQ_CTRL: START_ENABLED	With PQ_ENABLE, enables the generation of the "pq_start" signal to trigger the TX waveform. Enable if using display synchronization, disable otherwise.	ENABLED / DISABLED (default = DISABLED)
TSS: PQ_CTRL: PQ_ENABLED	With START_ENABLE, enables the generation of the "pq_start" signal to trigger the TX waveform. Enable if using display synchronization, disable otherwise.	ENABLED / DISABLED (default = DISABLED)
TSS: PQ_CTRL2: ACT_RANGE	Filter for the active edge of the synchronization signal. It is the number of additional cycles (units of the system clock – usually 48 MHz) the signal must be active for the synchronization trigger to be recognized. For example, a value of 2 will require the signal to be active for 3 consecutive system clocks. If the POLARITY parameter is set to NORMAL, this active edge is the rising-edge, if it is set to INVERTED it is the falling-edge.	0 – 65535 (default = 0)
TSS: PQ_CTRL2: DEACT_RANGE	Same as ACT_RANGE, but for the opposite edge of the synchronization signal.	0 – 255 (default = 0)
TSS: PQ_CTRL3:	Delay from a valid synchronization trigger to the start of the TX waveform. The delay is in units of the system clock (usually 48 MHz).	0 – 16777215 (default = 0)
TSS: SYNC_CONTROL	Selects the display synchronization method, unless set to "Disabled".	Disabled / Sync Scan Start / Sync TX pattern / Sync TX period (default = Disabled)

8.4 General optimization suggestions

8.4.1 Power consumption in active state

Power consumption in the active state is proportional to the ratio between the Active mode duration (which is the longer of the scan time and the data processing time) and the active interval (**Device Setup: ACT_INTRVL0**). If the ratio is less than 1, the controller is in Sleep mode saving power for part of the time. If the ratio is 1 or higher, the controller is always in Active mode. The total power consumption is calculated using Equation 38.

$$\text{Power} = (\text{VDDD} \cdot \text{IDDD}) + (\text{VDDA} \cdot \text{IDDA})$$

Equation 38

Consider the following steps to reduce power:

Increase the active interval: This is the most effective way to reduce power. The touch controller sleep for longer, without trading off SNR. The trade-off is a reduced refresh rate.

Reduce the VDDA level: Power consumption is linearly scaled with the supply voltage level. The digital supply is regulated by default, so the VDDD voltage level has little effect. Although the analog supply has no regulator, the analog supply current stays relatively flat with higher VDDA.

Reduce the scan time: If the refresh rate is dominated by either the active interval or the process time, use a reduced number of TX pulses to decrease the scan time to improve the analog supply current. If the refresh rate is dominated by the scan time, reducing the number of TX pulses increases the refresh rate. But the power consumption will remain roughly constant.

Reduce the TX voltage level: This will only impact the analog supply current. SNR is reduced proportionally.

8.4.2 Optimization trade-offs

SNR, refresh rate, and power consumption are interrelated. Improving the performance of one may degrade the performance of the other two. In addition, because each design has different priorities, there is no single way of optimizing the SNR, refresh rate, and power. Follow the optimization procedure, and/or modify it based on the project requirements. The optimization is illustrated in Figure 61.

1. Enable Multi-phase TX for mutual-cap panel scan (**TSS: MTX_ORDER**), see Section 3.3.5.
2. Set the active interval to get the required refresh rate.
3. Sweep the mutual-cap TX pulses to find the refresh rate transition point ("TX_PULSES_1" in Figure 61).
4. Sweep the mutual-cap TX pulses from 2 to the transition point and measure the power. Find another point "TX_PULSES_2" where it is just below the target power consumption.
5. Evaluate the SNR (see Section 2.7) with mutual-cap TX Pulses from 2 to TX_PULSES_2.

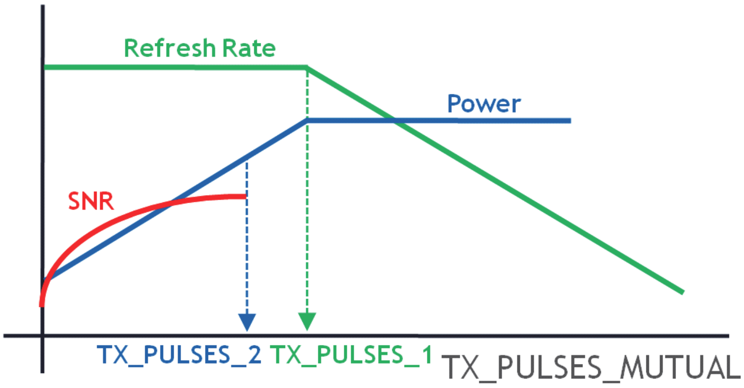


Figure 61 General SNR, refresh rate, and power consumption optimization illustration

8.4.3 Parameters

Table 33 summarizes the parameters discussed for optimizing SNR, refresh rate, and power.

Table 33 SNR, refresh rate and power tradeoff summary

Parameter or Item	Description
Device Setup: ACT_INTRVL0	Increasing this parameter decreases refresh rate and power consumption. There is no effect if the setting is refresh rate is dominated by scan time.
VDDD, VDDA voltages	Increasing voltage can increase SNR and increase power consumption.
TSS: TX_PUMP_VOLTAGE	When **TSS: VDDA_MODE** is set to Pump Mode, increasing this parameter can increase SNR and will increase power consumption.
TSS: MTX_ORDER	Increasing this parameter increases SNR and increases power consumption.
TSS: TX_PULSES_MC (SC)	Increasing this parameter increases SNR, decreases refresh rate, and increases power consumption.
Raw data filters	Enabling raw data filters increases SNR.

8.5 Baselining

The baseline update algorithms must be tuned so that environmental factors (for example, temperature, humidity, water, etc.) are baselined out, but noise and slowly approaching touches do not significantly affect the baseline. The baseline is updated by both the baselining algorithm and the temperature compensation algorithm. These two processes are briefly introduced in the following sections, and the tuning procedures are detailed.

8.5.1 Baseline background

The baseline should track changes in the raw-counts without the influence of a touch. Therefore, the noise and any touch information must not influence the baseline update process. The noise is removed using a low-pass filter, which is only tunable through the frequency at which it is applied (update rate). The touch information is removed by setting a threshold above which the baseline should no longer track the current raw-count values.

The baseline update process is therefore a priority list of when to increment/decrement the baseline. The process acts upon each slot independently. A simplified priority list is shown below (the process is the same for both mutual-cap and self-cap):

1. If MAX(diff-count) > baseline update threshold (parameters described below):
   a) Baselines not updated in this scan
2. If MIN(diff-count) < -10:
   b) Baselines are updated in this scan (ignores the baseline update speed)
3. If the scan is only performed for baseline maintenance (such as mutual-cap scan when in self-cap LFT mode):
   c) Baselines are updated in this scan (ignores the baseline update speed)
4. To get here, all other cases are false, so:
   d) Update the baselines at an interval specified by the baseline update speed (parameters described below).

There are separate baseline update speed parameters for finger, glove, and button (**Raw Processing: FINGER_BL_UPDATE_SPEED**, **Raw Processing: GLOVE_BL_UPDATE_SPEED**, and **Raw Processing: BTN_BL_UPDATE_SPEED**). These parameters control the number of scans to skip the baseline update. For example, an update speed of 10 will update the baseline once every 11 panel scans. It is recommended to update the baseline 50 to 100 times per second. For example, it is reasonable for a panel running at 120 Hz to set the update speed to 1 (baseline updated 60 times/sec).

8.5.2 Tuning baseline threshold

There are separate baseline threshold parameters for mutual-cap and self-cap (**Raw Processing: FINGER_BL_THRESH_MC** and **Raw Processing: FINGER_BL_THRESH_SC**). There are also parameters for gloves (**Raw Processing: GLOVE_BL_THRESH_MC** and **Raw Processing: GLOVE_BL_THRESH_SC**) and buttons (**Raw Processing: BL_THRESH_BTN_MC** and **Raw Processing: BL_THRESH_BTN_SC**). However, the tuning method is the same:

1. Fully tune the module for noise mitigation.
2. Put the module in the noisiest environment that must be supported.
3. Put the TTHE in “Heat Map” display mode, select diff-counts, max-hold, and self-cap (if applicable).
4. Wait for the values in the heat map to stabilize.
5. Set the baseline threshold midway between the worst-case noise and the finger liftoff threshold (**Fingers: FINGER_THRESH_MUT_LO** for MC, **Fingers: FINGER_THRESH_SELF** for SC):
   - If the worst-case noise is higher than this threshold, then trade-offs must be made.
6. Fine-tune the baseline threshold according to the project-specific requirements within the available range noting the following trade-offs:
   - A lower threshold will better detect very slowly moving/approaching touches.
   - A higher threshold will have better rejection/recovery from noise and water.

8.5.3 Tuning baseline sensor locking

A finger touch provides a Gaussian-like distribution of difference counts around the touched area. The middle of the touch will result in signal that is far above the baseline update threshold. However, the “tails” on either side would normally fall under the baseline update threshold. These small signals provide essential information for an accurate centroid. If unprotected, the small signals would quickly be baselined out, resulting if a far less accurate position report.

The **Raw Processing: FINGER_BL_SNS_WIDTH** and **Raw Processing: GLOVE_BL_SNS_WIDTH** parameters provide this protection by “locking” baseline updating for a configurable number of sensors on each side of the sensors above the baseline threshold.

It is recommended to lock about 10 mm for fingers and 15 mm for gloves on either side of the locked sensors to allow for slow moving objects. Therefore, set the sensor width by the following equation (the pitch is the size of the sensor in mm):

$$\text{Sensor Width} = \text{Roundup}\left(\frac{10\ (or\ 15)}{\text{Min(Xpitch,Ypitch)}}\right)$$

Equation 39

8.5.4 Baseline reset conditions

Baseline reset conditions are verified by the parameter (**Raw Processing: BL_RESET_DEBOUNCE**). It is recommended to leave this parameter at its default value. The baseline reset debounce parameter is used for different purposes by different algorithms.

There are different algorithms controlling a baseline reset depending on the configuration, and for the touch panel and buttons. The four baseline reset conditions are listed and then detailed:

1. Panel scan when self-cap data is available (**Scan Filtering: WATER_REJ_ENABLE** enabled)
2. Panel scan when only mutual-cap data is available (**Scan Filtering: WATER_REJ_ENABLE** disabled)
3. Button scan when self-cap data is available (Button Mode is "Hybrid" and **Scan Filtering: WATER_REJ_ENABLE** enabled)
4. Button scan when only mutual-cap data is available (Button Mode not "Hybrid" or **Scan Filtering: WATER_REJ_ENABLE** disabled)

The 1st condition checks whether a baseline reset is needed when there is both mutual-cap and self-cap data available (**Scan Filtering: WATER_REJ_ENABLE** enabled). The baselines are reset for the columns/rows when the negative touch threshold (**Fingers: FINGER_THRESH_MUT_HI** (mutual-cap) and **Fingers: FINGER_THRESH_SELF** (self-cap)) is passed for BOTH mutual-cap and self-cap for longer than the baseline reset debounce.

The 2nd condition checks for whether a baseline reset is needed when only mutual-cap data is available (**Scan Filtering: WATER_REJ_ENABLE** disabled). This condition is detailed in the palm-on-startup algorithm described in Section 8.14.

The 3rd condition checks whether a button baseline reset is needed during a startup period, when there is both mutual-cap and self-cap data available (Button Mode is "Hybrid" and **Scan Filtering: WATER_REJ_ENABLE** enabled). The algorithm only checks during a configurable startup period (defined by **Raw Processing: PALM_STARTUP_INTERVAL**). The baselines are reset for ALL buttons when the negative button touch threshold (**Buttons: BTN_LS_ON_THRSH_MUT_x** (mutual-cap) and **Buttons: BTN_LS_ON_THRSH_SELF_x** (self-cap)) is passed. The baselines continue to be reset for the baseline reset debounce period.

The 4th condition checks whether a button baseline reset is needed during a startup period, when there is only one scan method (either mutual-cap or self-cap) of data available (Button Mode not "Hybrid" or **Scan Filtering: WATER_REJ_ENABLE** disabled). The scan method used is mutual-cap if either the Button Mode is "Mutual Capacitance" or **Scan Filtering: WATER_REJ_ENABLE** disabled; the scan method is self-cap otherwise. The algorithm only checks during a configurable startup period (defined by **Raw Processing: PALM_STARTUP_INTERVAL**). The baselines are reset for ALL buttons when the negative button touch threshold (**Buttons: BTN_LS_ON_THRSH_MUT_x** (mutual-cap) or **Buttons: BTN_LS_ON_THRSH_SELF_x** (self-cap)) is passed. The baselines continue to be reset for the baseline reset debounce period.

8.5.5 Temperature compensation

8.5.5.1 Basic temperature compensation

Basic temperature compensation is provided by the standard baseline update algorithm. It is recommended to disable aggressive temperature compensation (set **Raw Processing: xxx_CMF_DELTA_TEMPCO_INC** = 0). Instead temperature compensation should be tuned using the baseline update rate (for example **Raw Processing: FINGER_BL_UPDATE_SPEED**).

The standard temperature compensation tuning procedure is as follows:

1. Disable aggressive temperature compensation (**Raw Processing: xxx_CMF_DELTA_TEMPCO_INC** = 0).
2. Disable all raw-data CMF filters so the raw-data can be accurately examined.
3. Put the TTHE in “Heat Map” display mode and log the raw-data values.
4. Return the TTHE to the “Touch Reporting” display mode.
5. Simulate the worst-case environmental changes that the temperature compensation algorithm is expected to compensate for.
6. Put the TTHE in “Heat Map” display mode and log the raw-data values again.
7. Find the maximum raw-data change.
8. Note the minimum scans per change:
9. roundDown((test time in secs * panel scan rate Hz) / MaxRawChange).
10. If less than 1, either reduce the temperature compensation requirement, or use aggressive temperature compensation.
11. Set **Raw Processing: FINGER_BL_UPDATE_SPEED** and **Raw Processing: GLOVE_BL_UPDATE_SPEED** to the MinScansPerChange, do not set this value lower than necessary

8.5.5.2 Aggressive temperature compensation overview

It is recommended to use the standard baseline update algorithm for temperature compensation. However, if the required temperature compensation rate cannot be achieved using the standard baseline update method alone, then aggressive temperature compensation can be used. However, note the side-effects detailed later in this section.

Aggressive temperature compensation is enabled when the “TemperatureThreshold” (**Raw Processing: xxx_CMF_DELTA_TEMPCO_INC** parameter) is non-zero. There are independent TemperatureThreshold settings for each scan type and noise state (the abbreviation **Raw Processing: xxx_CMF_DELTA_TEMPCO_INC** is used elsewhere):

- Mutual-cap panel scan for fingers: **Raw Processing: MC_CMF_DELTA_TEMPCO_INC**
- Self-cap panel scan for fingers: **Raw Processing: SC_CMF_DELTA_TEMPCO_INC**
- Mutual-cap button scan for fingers: **Raw Processing: BTN_MC_CMF_DELTA_TEMPCO_INC**
- Self-cap button scan for fingers: **Raw Processing: BTN_SC_CMF_DELTA_TEMPCO_INC**
- Mutual-cap panel scan for gloves: **Raw Processing: GLOVE_MC_CMF_DELTA_TEMPCO_INC**
- Self-cap panel scan for gloves: **Raw Processing: GLOVE_SC_CMF_DELTA_TEMPCO_INC**
- Mutual-cap button scan for gloves: **Raw Processing: BTN_GLOVE_MC_CMF_DELTA_TEMPCO_INC**
- Self-cap button scan for gloves: **Raw Processing: BTN_GLOVE_SC_CMF_DELTA_TEMPCO_INC**
- Mutual-cap scan (charger armor enabled): **Raw Processing: CA_MC_CMF_DELTA_TEMPCO_INC**
- Mutual-cap button scan (charger armor): **Raw Processing: CA_BTN_MC_CMF_DELTA_TEMPCO_INC**

The aggressive temperature compensation algorithm is part of the standard CMF filter and works as follows (when the "TemperatureThreshold" (**Raw Processing: xxx_DELTA_TEMPCO_INC**) is non-zero):

1. Standard CMF filter algorithm:

- Calculate the raw-count data "Drift" in a slot (may be positive or negative).

2. Update the raw-count scan data by subtracting the "Drift" from each sensor in the slot.
3. Detect and compensate for potential temperature change by updating all baselines in the slot according to the "Drift":

- ABS(Drift) < TemperatureThreshold:
  a) If positive Drift, increment baselines by 1.
  b) If negative Drift, decrement baselines by 1.
- TemperatureThreshold <= ABS(Drift) < (4*TemperatureThreshold):
  a) If positive Drift, increment baselines by TemperatureThreshold.
  b) If negative Drift, decrement baselines by TemperatureThreshold.
- (4*TemperatureThreshold) <= ABS(Drift):
  a) If positive Drift, increment baselines by 1.
  b) If negative Drift, decrement baselines by 1.

The aggressive temperature compensation algorithm is designed for the baseline to track the raw-count data changes. In case of big display noise, the "drift" (average noise in the scan slot) should be bigger than the upper threshold (4*TemperatureThreshold), and so the baseline will only change by 1 count per panel scan. If the raw-count data is indeed changing due to environmental changes (such as temperature and humidity), the drift is relatively small, and therefore the baseline's ability to be updated by TemperatureThreshold each panel scan would be sufficient.

The extreme case should be when the absolute maximum temperature change causes a raw-count data drift as large as the TemperatureThreshold value on successive panel scans, as this would be the limit that the temperature compensation algorithm can completely compensate.

8.5.5.3 Aggressive temperature compensation tuning

Tuning of the aggressive temperature compensation requires setting the TemperatureThreshold parameter (see previous section). The procedure requires a method of simulating the worst-case environmental changes that the temperature compensation algorithm is expected to compensate for. This is usually a temperature chamber.

The procedure is as follows:

1. Disable temperature compensation by setting TemperatureThreshold to 0
2. Disable all raw-data CMF filters so the raw-data can be accurately examined
3. Put the TTHE in "Heat Map" display mode and log the raw-data values
4. Return the TTHE to the "Touch Reporting" display mode
5. Simulate the worst-case environmental changes that the temperature compensation algorithm is expected to compensate for
6. Put the TTHE in "Heat Map" display mode and log the raw-data values again
7. Find the maximum raw-data change
8. Max change per scan: roundup(MaxRawChange / (test time in secs * panel scan rate Hz))
9. Set TemperatureThreshold to the MaxChangePerScan, do not set higher than necessary

As an example of the above procedure:

- System definition:
  - The sensor with the larges change starts with a raw-count of 8,000 and ends with a raw-count of 10,100
  - The test time is 10 seconds
  - The scans rate is 100 Hz
- Tuning process:
  - The MaxRawChange = 10,100 – 8,000 = 2,100
  - The MaxChangePerScan = roundup(2100 / (10 * 100)) = 3

8.5.5.4 Aggressive temperature compensation side-effects

An example of a problem that may occur with a fast updating baseline, is a stuck baseline when a large object is very slowly moved. The aggressive temperature compensation algorithm continues to update the baselines, even when a touch is present. Unfortunately, if a large object slowly moves onto (or off) a whole slot, the raw-data change can appear to be a slow drift in the raw-data. The baseline is therefore updated more aggressively (as if it were a temperature compensation) using the TemperatureThreshold adjustment. If the large object continues to slowly increase the signal in the whole slot, and therefore masquerade as a continued temperature change, the baseline can be updated significantly enough to be "locked" when the large object is removed. This process is illustrated in Figure 62:

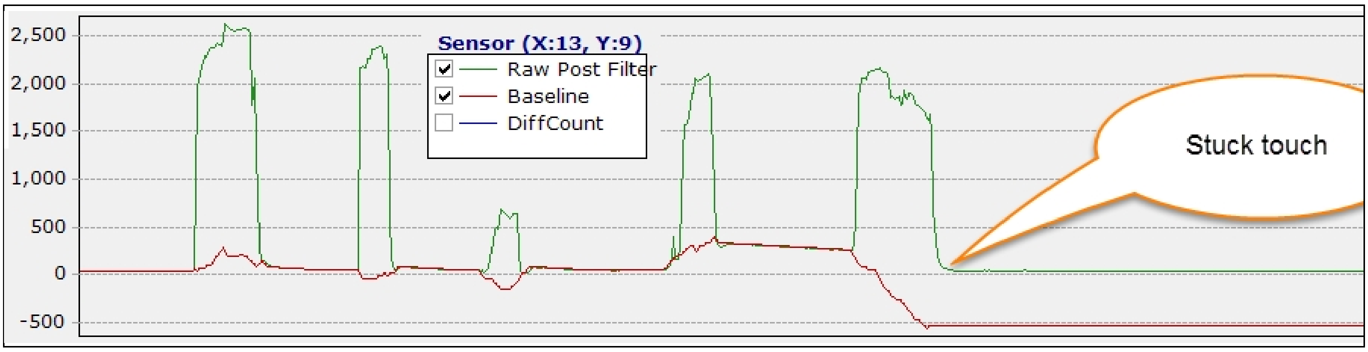


Figure 62 Side-effect of temperature compensation

In Figure 62, the large object touch in question is the right-most touch (shown in green). The intensity of the touch on this sensor slowly decreases. If the TemperatureThreshold is set too high, this decrease in the raw-data is seen as a continuing temperature change (assuming the decrease is consistent across an entire slot). Normally, raw-count changes due to touches are far more significant that the temperature compensation upper threshold (4 * TemperatureThreshold). However, in this hypothetical case, the temperature compensation algorithm erroneously reduces the baselines. If this occurred inconsistently, the baseline would recover. However, in the extreme case when the TemperatureThreshold is set too high, and a large object is very slowly moved onto (or off) a whole slot for many cycles, then a stuck baseline (as illustrated in Figure 62) can occur. Therefore, as mentioned in the tuning method, the TemperatureThreshold must be set as low as possible for the environmental changes that must be compensated for.

8.5.6 Parameters

Table 34 Baseline tuning parameters

Configurable Parameter	Description	Selection
Charger Armor: CHARGER_ARMOR_ENABLE	Enable/Disable HOST Controlled Charger Armor. Initializes the CHARGER_STATUS RAM value at device power-up. Rarely changed from default value.	Enabled / Disabled (default = Disabled)
Raw Processing: MC_CMF_DELTA_TEMPCO_INC **Raw Processing: SC_CMF_DELTA_TEMPCO_INC** **Raw Processing: BTN_MC_CMF_DELTA_TEMPCO_INC** **Raw Processing: BTN_SC_CMF_DELTA_TEMPCO_INC** **Raw Processing: GLOVE_MC_CMF_DELTA_TEMPCO_INC** **Raw Processing: GLOVE_SC_CMF_DELTA_TEMPCO_INC** **Raw Processing: BTN_GLOVE_MC_CMF_DELTA_TEMPCO_INC** **Raw Processing: BTN_GLOVE_SC_CMF_DELTA_TEMPCO_INC** **Raw Processing: CA_MC_CMF_DELTA_TEMPCO_INC**	CMF Temperature compensation speed. Higher values cause fast temperature compensation, lower values slow down temperature compensation. Zero value disables temperature compensation. CMF average larger than 4x the value will not be considered a temperature change.	0 – 127 (default = 1)
Raw Processing: FINGER_BL_SNS_WIDTH	This parameter defines how many sensors must be locked on each side of the sensor with baseline hold condition.	0 – 7 (default = 3)
Raw Processing: FINGER_BL_UPDATE_SPEED	This parameter defines the baseline updating speed. The higher the number, the slower the baseline updating is.	0 – 31 (default = 1)
Raw Processing: FINGER_BL_THRESH_MC	Baseline threshold for mutual-cap. If the maximum signal in the column is above this threshold, baseline updating will be locked. Only used if water rejection is disabled	0 – 32767 (default = 179)
Raw Processing: FINGER_BL_THRESH_SC	Baseline threshold for self-cap. If the self-cap signal is above this threshold, baseline updating will be locked. Only used if water rejection is disabled	0 – 32767 (default = 200)
Raw Processing: GLOVE_BL_SNS_WIDTH	This parameter defines how many sensors must be locked on each side of the sensor with baseline hold condition.	0 – 7 (default = 3)
Raw Processing: GLOVE_BL_UPDATE_SPEED	This parameter defines the baseline updating speed. The higher the number, the slower the	0 – 31

Configurable Parameter	Description	Selection
	baseline updating is. It is recommended to set this parameter to 1.	(default = 1)
Raw Processing: GLOVE_BL_THRESH_MC	Baseline threshold for mutual-cap. If the maximum signal in the column is above this threshold, baseline updating will be locked.	0 – 32767 (default = 30)
Raw Processing: GLOVE_BL_THRESH_SC	Baseline threshold for self-cap. If the self-cap signal is above this threshold, baseline updating will be locked.	0 – 32767 (default = 15)
Raw Processing: BL_UPDATE_SPEED_BTN	This parameter defines the baseline updating speed. The higher the number, the slower the baseline updating is.	0 – 31 (default = 15)
Raw Processing: BL_THRESH_BTN_MC	Baseline threshold for mutual-cap buttons. If the maximum signal in the column is above this threshold, baseline updating will be locked.	0 – 32767 (default = 50)
Raw Processing: BL_THRESH_BTN_SC	Baseline threshold for self-cap buttons. If the self-cap signal is above this threshold, baseline updating will be locked.	0 – 32767 (default = 50)
Raw Processing: BL_RESET_DEBOUNCE	This parameter defines the negative baseline reset debounce time in ms.	0 – 255 (default = 100)

8.6 Touch zone tuning

The process of defining a touch zone is as follows:

1. Identify the initial candidate touch zones (see Section 8.6.1).
2. For each initial touch zone, find the valid peaks (see Section 8.6.2).
3. If there is >1 peak in a touch zone:
   a) If the number of peaks is unchanged, then number of touches does not change.
   b) If the number of peaks has changed, then apply the finger separation algorithms (see Section 8.6.4).

8.6.1 Initial touch zone

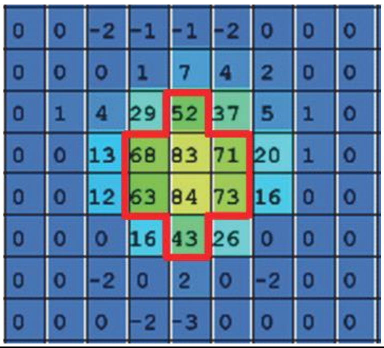
The active sensors of each touch object are grouped into an area referred to as a touch zone. After the touch signal peak (local max) is found, a signal threshold is used to determine the active sensors surrounding the peak sensor. Any sensor surrounding the peak sensor with a signal larger than this threshold is considered part of the touch zone.

The signal threshold is the product of the local maximum signal (X_{peak}) and parameterized “Signal Threshold” multiplier. There are separate multipliers for each scan type**: Fingers: FINGER_SIG_THRESH_MULT** and **Fingers: GLOVES_SIG_THRESH_MULT**. The signal threshold is represented in factions of 128, see Equation 40.

$$X_{threshold} = \frac{X_{peak} \cdot \text{Signal Threshold}}{128}$$

Equation 40

For the example shown in Figure 63, the touch zone peak is 84 counts. With a signal threshold value of 64, the touch zone threshold is 42 counts. Any sensor around the peak sensor with a touch signal above 42 is considered part of the touch zone. The touch zone size is the number of sensors including the peak sensor; in Figure 63, the touch zone size is 8.



0	0	-2	-1	-1	-2	0	0	0
0	0	0	1	7	4	2	0	0
0	1	4	29	52	37	5	1	0
0	0	13	68	83	71	20	1	0
0	0	12	63	84	73	16	0	0
0	0	0	16	43	26	0	0	0
0	0	-2	0	2	0	-2	0	0
0	0	0	-2	-3	0	0	0	0

Figure 63 Touch zone example (diff-count)

8.6.2 Confirm touch zone peaks

The number of peaks should not jitter, with peak being “only just” added or removed.

The peak ignore algorithm (**Fingers: PEAK_IGNORE_COEFF**) is used to remove peaks that are significantly less than the biggest peak in the touch zone. It is recommended to leave this value at its default.

The virtual peak algirithm (**Fingers: VP_DLT_THRESH** and **Fingers: VP_DLT_RST_THRESH**) is used to add a peak if it has just dipped below the detection threshold. It is recommended to leave both these values at their defaults.

8.6.3 Finger size tuning

Finger touches are divided into the following three categories based on the touch zone size: normal finger, fat finger (FF), and large object (LO). Different centroid algorithms are employed based on the detected finger size and signal value. The reporting of fingers and large objects can also be complex, refer to Section 3.3.3 and the device TRM.

During touchdown, a touch is considered a regular finger if the touch zone size is less than the minimum fat finger size, as shown in Figure 64. A touch is considered a fat finger if the touch zone size is between the minimum fat finger size and the maximum fat finger size. Otherwise, it is considered a large object if the number of active sensors is equal to or larger than the maximum fat finger size.

To avoid oscillation between finger sizes, hysteresis is used. When a fat finger is detected, the minimum fat finger threshold is reduced preventing toggling between two categories. Note that unintended switching between centroid algorithms could affect accuracy and linearity. Similarly, hysteresis is applied to the maximum fat finger threshold when a large object is detected.

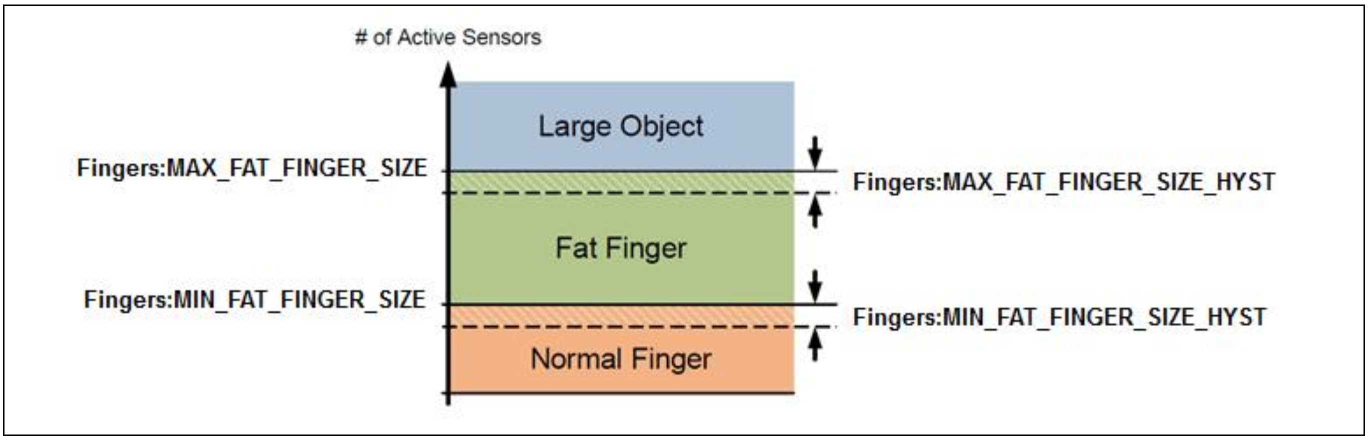


Figure 64 Finger size categories for centroid algorithm based on number of active sensors

8.6.3.1 Tuning procedure for maximum fat finger size

Use the following procedure to tune the maximum fat finger threshold and hysteresis:

1. Prepare a metal finger with the minimum large-object size.
2. Set the max fat finger size hysteresis to zero, for size tuning.
3. In the Heat Map display mode, set DataType to "DiffCounts".
4. Place the grounded large object on the panel and count the number of sensors that have a higher diff-count than the mutual-cap finger threshold.
5. Set the maximum fat finger size parameter to the measured sensor count.
6. In the Touch Reporting display mode, move the large object on the panel ensuring it is fully within the panel borders.
7. If a finger touch is reported, decrement the maximum fat finger size parameter, and retest.
8. The hysteresis is typically set to about 20% of the threshold value.

8.6.3.2 Tuning procedure for minimum fat finger size

Use the following procedure to tune the minimum fat finger threshold and hysteresis:

1. Prepare a metal finger with the minimum fat-finger size (usually about 12 mm).
2. In the Heat Map display mode, set DataType to "DiffCounts".
3. Center the grounded fat finger on a sensor intersection (left of Figure 65).
4. Record the number of active sensors.
5. Center the grounded fat finger on a single sensor (right of Figure 65).
6. Record the number of active sensors.
7. Set the minimum fat finger size parameter to the lowest value from steps 4 and 6.
8. The minimum fat finger size hysteresis is typically set to about 50% of the threshold value.

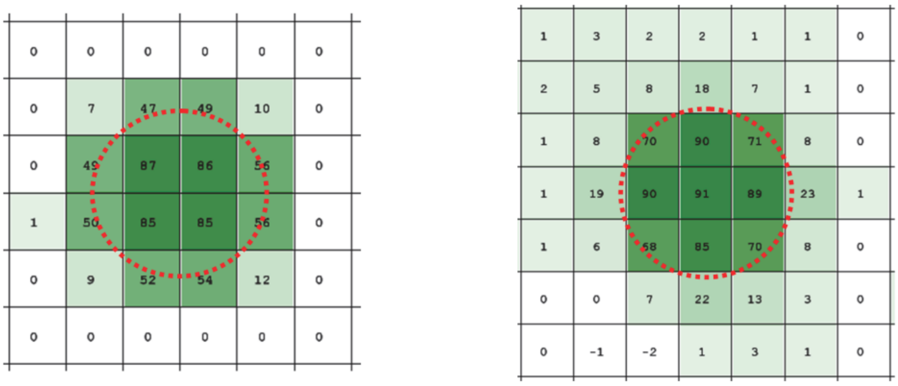


Figure 65 Finger positioning for tuning fat finger size

8.6.4 Fat-finger vs. finger separation tuning

When two or more signal peaks are detected in a touch zone, the touch controller can report it as a single fat finger or as multiple finger touches, depending on the parameter **Fingers: FINGER_OBJECT_FEATURES** (or **Fingers: GLOVES_OBJECT_FEATURES**).

There are four levels of processing covered in the following three subsections. Once the object type is decided (fat finger or multiple touches), there is no further processing. The process is summarized as:

1. Object features setting
2. Max center 3x3 signal sum algorithm (only performed if the touch is in the panel core)
   a) Slider: Touch is on the "panel edge", so skip this algorithm
   b) Glove: If touch is in the panel core, and the algorithm is tuned for fingers (recommended), always reports multiple touches
3. Center signal to min peak ratio algorithm (panel core touch classification completed here)
4. Total signal sum algorithm (only performed if the touch is on the panel edge)

8.6.4.1 Object features setting

If the "fat-finger object" feature is disabled, the touch controller will always report multiple signal peaks as multiple finger touches.

If the "fat-finger object" feature is enabled and the "multi-finger object" feature is disabled, the touch controller will always report multiple signal peaks as a single finger touch.

If both features are enabled, the touch controller will perform further analysis to determine the object type.

Note that once a touch zone with multiple peaks is classified, it will not change classification until the number of peaks in the zone changes.

Table 35 Finger touch interpretation for 2 or more signal peaks in a touch zone

xxx_OBJECT_FEATURES		When two or More Signal Peaks Detected in a Touch Zone
multi-finger object	**fat-finger object**	
Disabled	Disabled	Always report as multiple finger touches.
Enabled		
Disabled	Enabled	Always report as a single fat finger touch.
Enabled		Determine based on the Max Center 3x3 Signal Sum algorithm or the Center Signal to Min Peak Signal algorithm.

8.6.4.2 Max center 3x3 signal sum algorithm

If the fat-finger feature and the multi-finger feature are both enabled, and the touch is in the panel core, the device will first evaluate the touch zone using the “Max Center 3x3 Signal Sum” algorithm. Touches on the panel edge (including all touches on a slider) skip this algorithm, and proceed directly to the “Center Signal to Min Peak Ratio” algorithm. Multi-peak glove touches in the panel core will be reported as multiple fingers, as the tuning parameters are optimized for finger touches.

The algorithm picks the center 3x3 sensors of the touch zone, calculates the signal sum. If there are two or more choices (see Figure 67), the maximum of all the possibilities is used.

The maximum center Z9 is compared with the parameters **Fingers: MIN_FF_Z9** and **Fingers: MAX_MF_Z9**. If the signal peaks are due to multiple close fingers, the signal count is usually lower for those sensors in between the fingers. Therefore, if the finger gap is captured in the selected 3x3 segment, the total signal sum should be lower than that of a single fat finger. Figure 66 shows 4 examples. The 3x3 signal sum of a fat finger on the right is consistently larger than that of the two close fingers on the left.

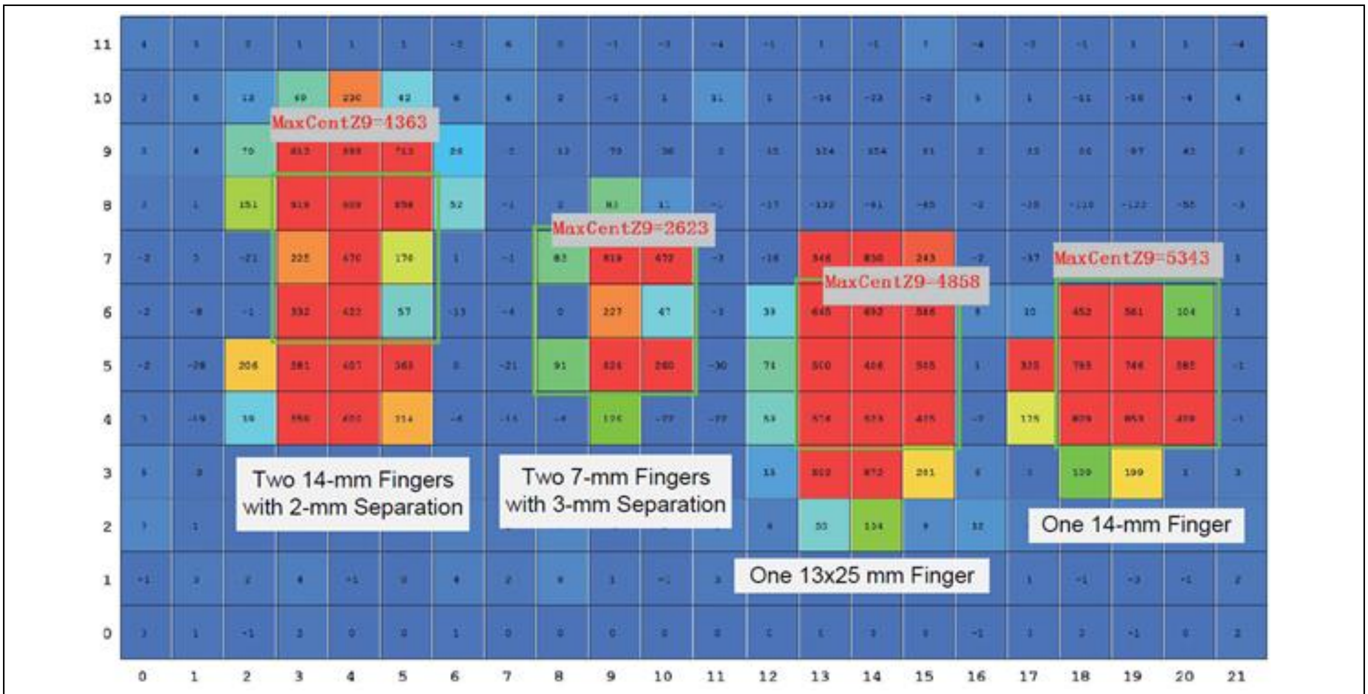


Figure 66 Center 3x3 signal sum examples

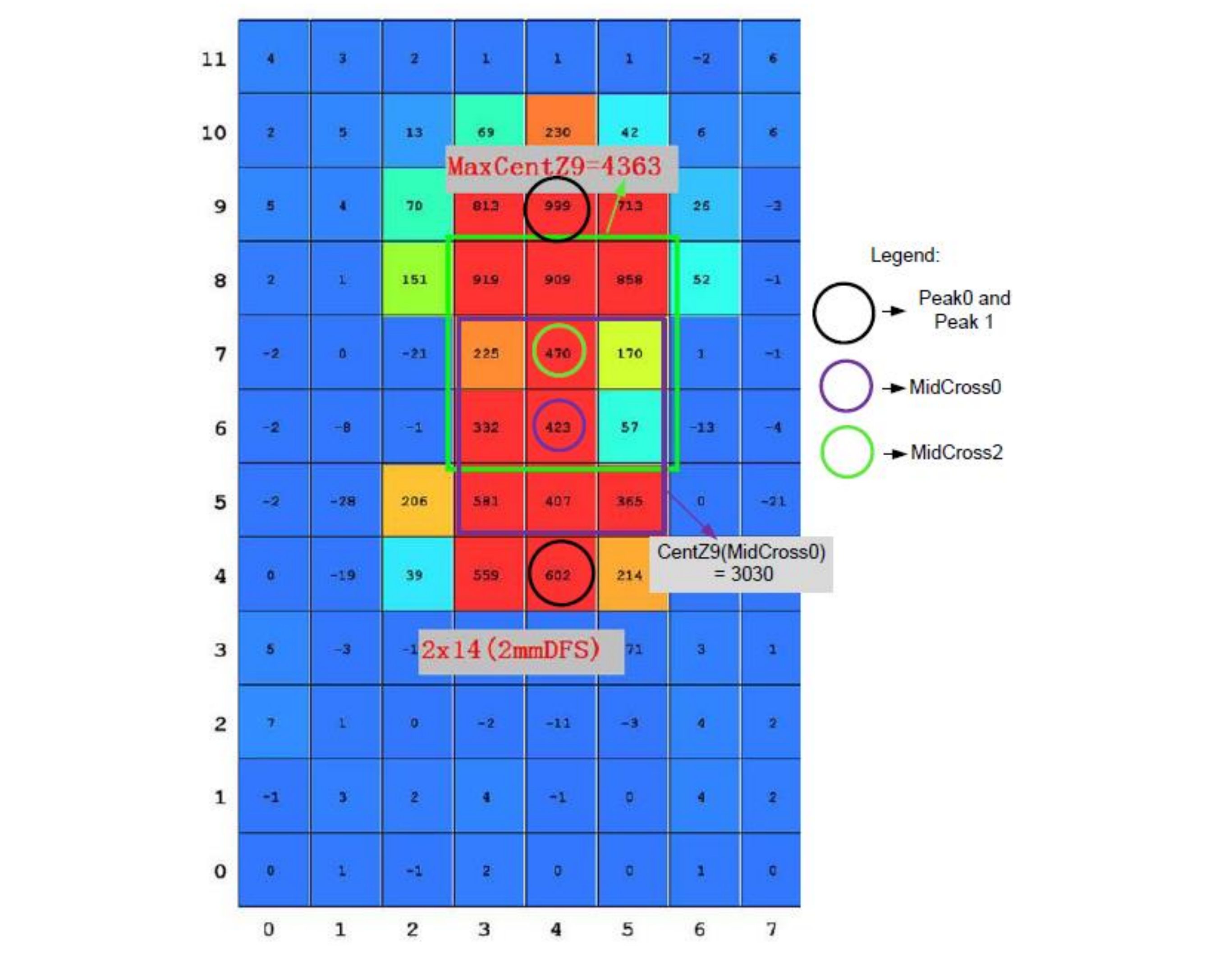


Figure 67 Center 3x3 signal sum detail

The algorithm compares the calculated signal sum against the **Fingers: MIN_FF_Z9** and **Fingers: MAX_MF_Z9**. If the signal sum is less than **Fingers: MIN_FF_Z9**, the device reports multiple finger touches in the touch zone. If the signal sum is larger than **Fingers: MAX_MF_Z9**, the device reports a single fat finger in the touch zone. If the signal sum is in between the two thresholds, the device will proceed to the next algorithm.

It is recommended setting **Fingers: MIN_FF_Z9** to four times the 12 mm finger diff-count at the HTI position, and setting **Fingers: MAX_MF_Z9** to six times the 12 mm finger diff-count at the HTI position.

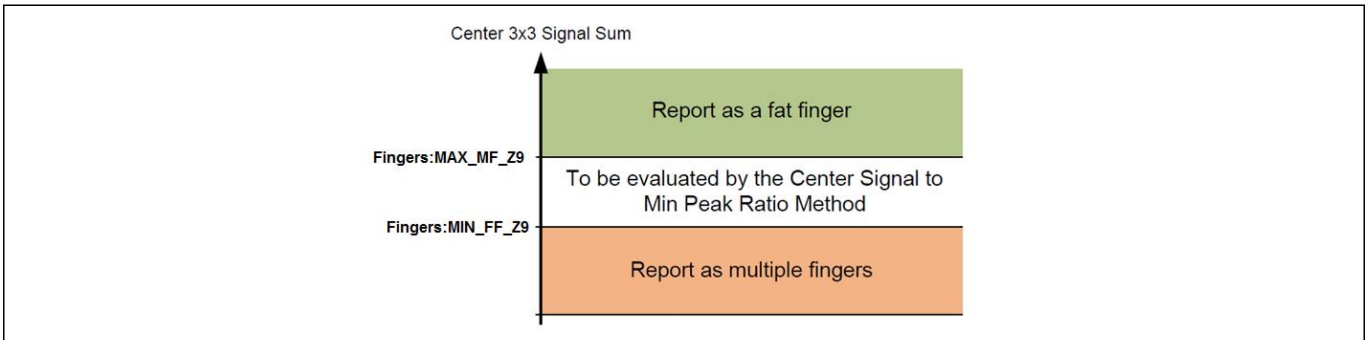


Figure 68 Center 3x3 signal sum evaluation

8.6.4.3 Center signal to min peak ratio algorithm

The touch controller will use this algorithm if it is unable to conclude whether the touch zone consists of a single fat finger or multiple finger touches, based on the Center 3x3 Signal Sum algorithm.

This algorithm will select the sensors in the middle of the two peaks, calculate the average signal count, and compare it with the lower of the two signal peaks. See Figure 69 for examples of the center signal selection method.

The idea is similar to the "Center 3x3 Signal Sum" method. If the signal peaks are due to multiple close fingers, the signal count of the sensors between the peaks is usually much lower than that of a fat finger.

$$\text{Signal\_Ratio} = 64 \cdot \frac{\text{Avg\_ Center\_Signal}}{\text{Min\_Peak\_Signl}}$$

Equation 41

If the calculated Signal_Ratio from Equation 41 is lower than the parameter **Fingers: MF_CENTERSIG_RATIO**, the device will report multiple touches. Otherwise, the classification depends on whether the touch is on the panel edge (including slider). If the touch is in the panel core, the touch is classified as a single fat finger. However, if the touch is on the panel edge (or anywhere on a slider), classification is passed to the "Total Signal Sum" algorithm.

There are six examples shown in Figure 69. Note that the ratio is shown as a percentage, while the TTHE parameter is base 64. There are four examples of a single finger, in all four cases, the Signal_Ratio is about 90% (TTHE parameter of 57). There are two examples of multiple close fingers, in both cases, the Signal_Ratio is about 65% (TTHE parameter of 42).

Figure 69 Center signal ratio examples

In most cases, the default parameter setting of 48 (75%) works well (almost midway between the examples above of 42-57). If not, follow this procedure to evaluate the **Fingers: MF_CENTERSIG_RATIO**.

1. Disable the center 3x3 signal sum algorithm:
   - Note the current values of **Fingers: MIN_FF_Z9** and **Fingers: MAX_MF_Z9**.
   - Set **Fingers: MIN_FF_Z9** to 0 and **Fingers: MAX_MF_Z9** to the max value (500,000).
2. Evaluate with different sizes and shapes of fingers.
3. Update the tuning parameter as needed:
   - Increase if a single finger is reported when there are 2 close fingers on the panel.
   - Decrease if multiple fingers are reported when there is only one fat finger.
4. Return **Fingers: MIN_FF_Z9** and **Fingers: MAX_MF_Z9** to the original values.

8.6.4.4 Total signal sum algorithm

The touch controller will use this algorithm only if the touch is on the edge, and it is unable to conclude whether the touch zone consists of a single fat finger or multiple finger touches based on the previous algorithms.

When touch object enters the panel from the edge, a touch zone can contain more than one peak. Examples are shown in Figure 70 where a touch zone at the edge of the panel contains 2 peaks. The total signal sum is the sum of all cells in the touch zone (red, orange, and yellow cells in Figure 70). In the examples, it can be seen that the total signal sum of a fat-finger is much larger than that of multiple close touches.

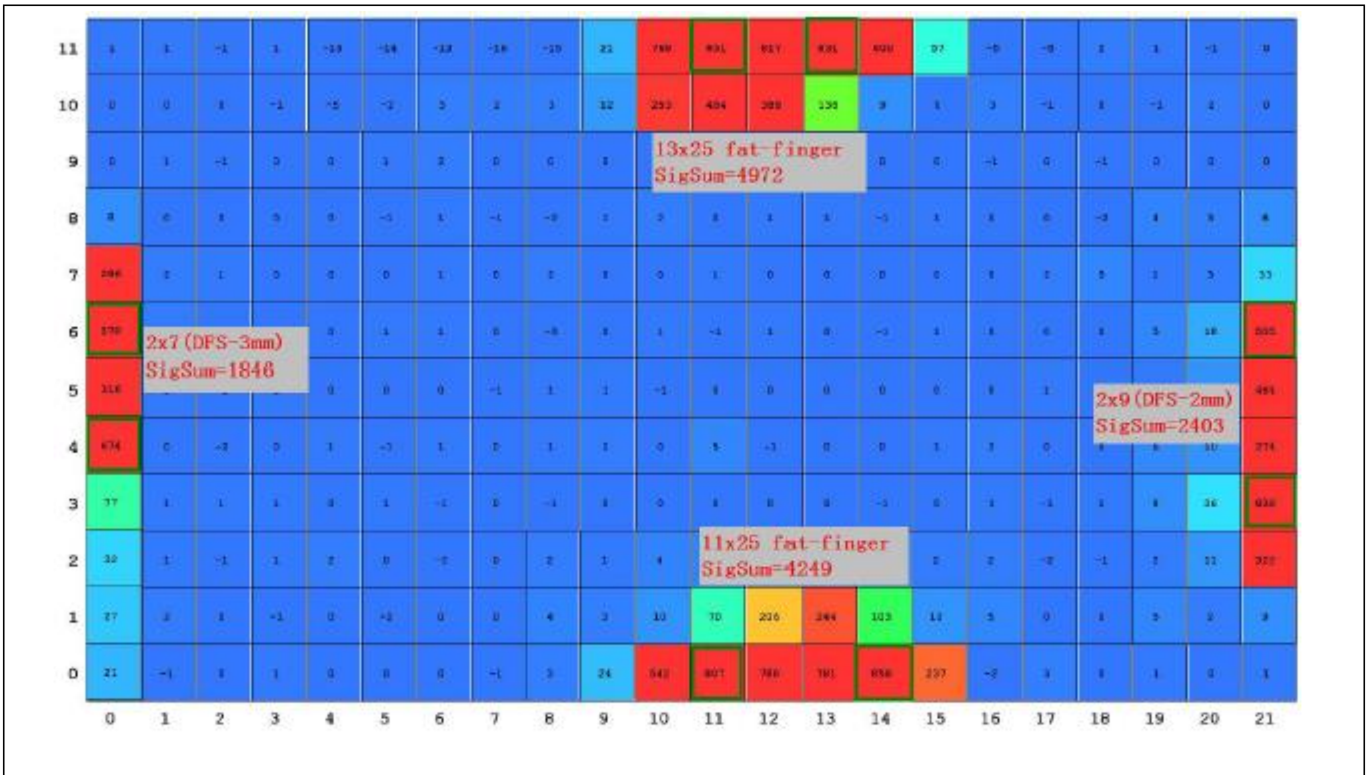


Figure 70 Center signal ratio examples

The tuning of the TTHE parameter (**Fingers: MIN_FF_SIG_SUM_EDGE**) is dependent on the unit cell size and signal levels. Tune the thresholds by reviewing the heatmap data (use average diff counts to make the data more readable) using various fat finger sizes, and multiple touch finger sizes.

8.6.5 Parameters

Table 36 Touch zone tuning parameters

Configurable Parameter	Description	Selection
Fingers: MIN_FAT_FINGER_SIZE	Minimum number of contiguous activated panel sensors that define a fat finger during touchdown.	0 – 255 (default = 10)
Fingers: MIN_FAT_FINGER_SIZE_HYST	Hysteresis applied to the minimum number of contiguous activated panel sensors that define a fat finger after touchdown.	0 – 255 (default = 1)
Fingers: MAX_FAT_FINGER_SIZE	Minimum number of contiguous activated panel sensors that define a large object during touchdown.	0 – 255 (default = 32)
Fingers: MAX_FAT_FINGER_SIZE_HYST	Hysteresis applied to the minimum number of contiguous activated panel sensors that define a large object after touchdown.	0 – 255 (default = 8)
Fingers: FINGER_SIG_THRESH_MULT	Proportion of the peak sensor value (in 1/128 increments) above which a neighbor is included in a touch zone.	0 – 128 (default = 64)
Fingers: FINGER_OBJECT_FEATURES: multi-finger object	Enables detection of multi-finger objects.	Enabled / Disabled (default = Enabled)
Fingers: FINGER_OBJECT_FEATURES: fat-finger object	Enables detection of fat-finger objects.	Enabled / Disabled (default = Enabled)
Fingers: MIN_FF_Z9	Minimum fat finger threshold used in the center 3x3 signal sum algorithm.	0 – 500000 (default = 5400)
Fingers: MAX_MF_Z9	Maximum multi-finger threshold used in the center 3x3 signal sum algorithm.	0 – 500000 (default = 8100)
Fingers: MIN_FF_SIG_SUM_EDGE	Minimum signal sum of fat-finger when 2 or more peaks detected on the panel edge.	0 – 500000 (default = 5400)
Fingers: MF_CENTERSIG_RATIO	Normalized ratio threshold of the valley to peak signal. Part of touch zone algorithms.	0 – 63 (default = 48)
Gloves: GLOVES_OBJECT_FEATURES: multi-finger object	Switches to enable or disable touch object features.	Enabled / Disabled (default = Enabled)
Gloves: GLOVES_OBJECT_FEATURES: fat-finger object	Switches to enable or disable touch object features.	Enabled / Disabled (default = Enabled)
Gloves: GLOVES_SIG_THRESH_MULT	Proportion of the peak sensor value (in 1/128 increments) above which a neighbor is included in a touch zone.	0 - 128 (default = 64)

8.7 Signal disparity detection tuning

Signal Disparity (SD) is the comparison of the finger signal between a grounded finger and an ungrounded finger. In this case, “grounded” does not necessarily mean directly connected to system ground. If the capacitance between the user’s body and system ground is much larger than the capacitance change from a finger touch, then the finger is effectively grounded. In automotive applications, there is generally a large capacitance from the user’s body to system ground (vehicle chassis), so SD issues are very rare.

The effects of SD are described below. By default, the SD parameters are set to disable SD detection, because the SD algorithm can have an adverse effect on finger separation. Do not enable the SD algorithm unless SD is confirmed to be a problem.

If present, the typical issues that result from SD are:

- Peak signal count reduction.
- A touch disappears when more than one finger is on the panel.
- A larger finger has lower signal count (or negative diff-count) in the center:
  - The donut-shaped signal map could result in multiple touches being reported.
  - See Figure 71

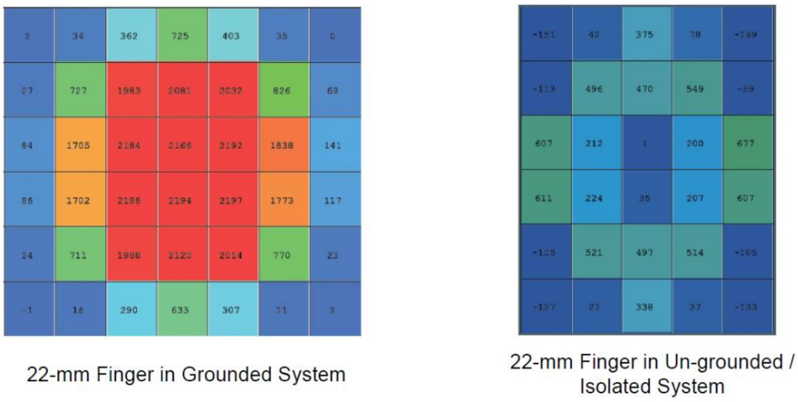


Figure 71 22 mm signal disparity heat map example

When the system is ungrounded or insolated, the sensor detects less capacitance change due to the finger touch. Figure 72 shows a simplified SD circuit representation for predicting the SD effect. Cbody is the human body capacitance coupling to the earth ground, which is in the range of 100 pF to 150 pF. The $C_{free\text{-}space}$ is the system coupling to earth ground, which is usually about 3 pF to 5 pF.

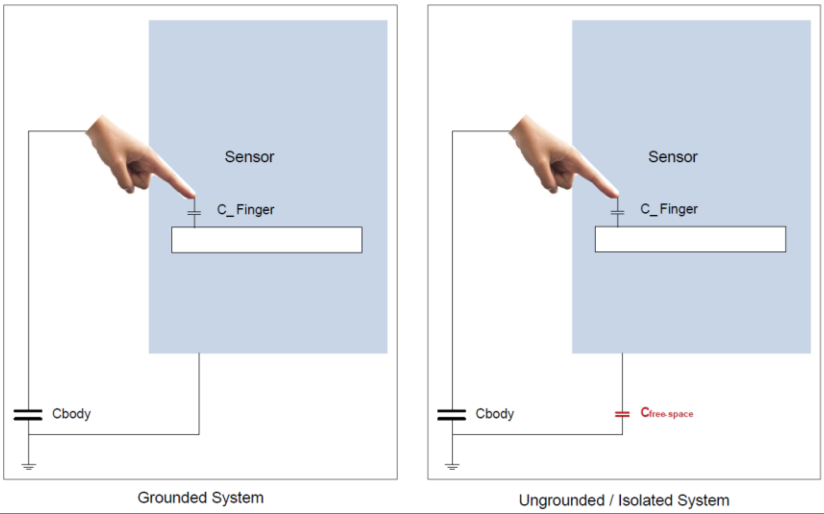


Figure 72 Simplified circuit representation between grounded and ungrounded system

When the system is grounded, the finger is coupled to earth ground through C_{body}. When the system is ungrounded, the finger is now coupled to earth ground through the series of C_{body} and C_{free_space}, which result in significantly lower capacitance.

The SD detection is specifically designed to prevent a larger finger from being reported as multiple finger touches. When the SD event is detected, only one fat finger touch is reported in that touch zone. There are two requirements to register a SD event:

- The size of the touch zone (in number of active sensors) must be equal to or larger than the parameter **SD Effect: SD_SIZE_THRESH** (or **Fingers: SD_SIZE_THRESH** in TSG6L).
- The maximum touch zone peak signal count must be less than the parameter **SD Effect: SD_SIG_THRESH_ON** (or **Fingers: SD_SIG_THRESH_ON** in TSG6L).

The touch zone will continue to be registered as a SD event until the finger is removed, or the maximum peak signal of any finger touch on the panel exceeds the value in parameter **SD Effect: SD_SIG_THRESH_OFF** (or **Fingers: SD_SIG_THRESH_OFF** in TSG6L).

Follow this procedure to tune the SD detection:

1. Set the SD size parameter to the number of sensors a 12 mm finger covered in HTI position. 12 mm finger is used because, in most cases, it is the common minimum fat finger size.
2. Set the SD "on threshold" to 5/8 of the diff-count of a 12 mm finger at HTI location.
3. Set the SD "off threshold" to 7/8 of the diff-count of a 12 mm finger at HTI location.
4. Select a number of different finger sizes larger than 12 mm.
5. Evaluate the touch performance with the finger grounded through a 150 pF cap and a 4 pF cap connected in series.
6. Increase the SD size parameter if multiple fingers are reported with a fat finger touch on panel.

8.7.1 Parameters

Table 37 SD detection parameters

Configurable Parameter	Description	Selection
SD Effect: SD_SIZE_THRESH CYAT8168 (TSG6XL) **Fingers: SD_SIZE_THRESH** CYAT6165/8165 (TSG6L)	Threshold (in number of active sensors) for the size of a touch zone to enable Signal Disparity detection.	0 – 128 (default = 9)
SD Effect: SD_SIG_THRESH_ON CYAT8168 (TSG6XL) **Fingers: SD_SIG_THRESH_ON** CYAT6165/8165 (TSG6L)	Max touch zone peak signal count threshold to enable the SD detection.	0 – 65535 (default = 1562)
SD Effect: SD_SIG_THRESH_OFF CYAT8168 (TSG6XL) **Fingers: SD_SIG_THRESH_OFF** CYAT6165/8165 (TSG6L)	Max touch zone peak signal count threshold to exit the SD detection.	0 – 65535 (default = 2187)

8.8 Finger tracking

8.8.1 Position filtering

Position filtering, or XY filtering, can be a very project specific requirement. The panel typically has consistent values; but may still require change due to project-dependent panel characteristics. Values for the slider and charger armor are project specific. The recommended initial panel values are listed below:

- **Scan Filtering: PANEL_XY_FILTER_MASK** all enabled (default all enabled)
- **Scan Filtering: PANEL_XY_FILT_IIR_COEF_SLOW** One Eighth (default One Quarter)
- **Scan Filtering: PANEL_XY_FILT_IIR_COEF_FAST** One (default One Half)
- **Scan Filtering: PANEL_XY_FILT_IIR_THR_SLOW** 5 (default 100)
- **Scan Filtering: PANEL_XY_FILT_IIR_THR_FAST** 50 (default 200)

8.8.1.1 Parameters

Table 38 XY filter parameters

Configurable Parameter	Description	Selection
Scan Filtering: XY_FILTER_MASK: Z Jitter Filter Enable	Enables or disables the Z jitter filter.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILTER_MASK: Z IIR Filter Enable	Enables or disables the Z IIR filter.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILTER_MASK: XY Jitter Filter Enable	Enables or disables the XY jitter filter.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILTER_MASK: XY IIR Filter Enable	Enables or disables the XY IIR filter.	Enabled / Disabled (default = Enabled)

Configurable Parameter	Description	Selection
Scan Filtering: XY_FILT_IIR_COEF_SLOW	Weighting of the input X and Y values in the IIR filter for displacements less than XY_FILT_SLOW_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Quarter)
Scan Filtering: XY_FILT_IIR_COEF_FAST	Weighting of the input X and Y values in the IIR filter for displacements greater than XY_FILT_FAST_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Scan Filtering: XY_FILT_XY_THR_SLOW	Displacement (in pixels) along X or Y axis below which the IIR filter input weight is set to XY_FILT_IIR_COEFF for filtering along the corresponding axis.	0 – 255 (default = 100)
Scan Filtering: XY_FILT_XY_THR_FAST	Displacement (in pixels) along X or Y axis above which the IIR filter input weight is set to XY_FILT_IIR_FAST_COEFF for filtering along the corresponding axis.	0 – 255 (default = 200)
Scan Filtering: XY_FILT_Z_IIR_COEFF	Weighting of the input Z value in the IIR filter.	1, 1/2, 1/4 (default = One Half)
Scan Filtering: XY_FILT_PREDICTION_COEF	Weight of the predicted position that is based on the velocity.	0 – 128 (default = 0)
Scan Filtering: XY_FILTER_MASK_CA: Z Jitter Filter Enable CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILTER_MASK: Z Jitter Filter Enable** 8168X (TSG6XL)	Enables or disables the Z jitter filter when Charger Armor is active.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILTER_MASK_CA: Z IIR Filter Enable CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILTER_MASK: Z IIR Filter Enable** 8168X (TSG6XL)	Enables or disables the Z IIR filter when Charger Armor is active.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILTER_MASK_CA: XY Jitter Filter Enable CYAT6165/8165 (TSG6L)	Enables or disables the XY jitter filter when Charger Armor is active.	Enabled / Disabled (default = Enabled)

Configurable Parameter	Description	Selection
Charger Armor: CA_XY_FILTER_MASK: XY Jitter Filter Enable 8168X (TSG6XL)		
Scan Filtering: XY_FILTER_MASK_CA: XY IIR Filter Enable CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILTER_MASK: XY IIR Filter Enable** 8168X (TSG6XL)	Enables or disables the XY IIR filter when Charger Armor is active.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILT_IIR_COEF_SLOW_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_IIR_COEF_SLOW** 8168X (TSG6XL)	Weighting of the input X and Y values in the IIR filter for displacements less than XY_FILT_SLOW_THR when Charger Armor is active.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Sixteenth)
Scan Filtering: XY_FILT_IIR_COEF_FAST_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_IIR_COEF_FAST** 8168X (TSG6XL)	Weighting of the input X and Y values in the IIR filter for displacements greater than XY_FILT_FAST_THR when Charger Armor is active.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Eighth)
Scan Filtering: XY_FILT_IIR_THR_SLOW_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_IIR_THR_SLOW** 8168X (TSG6XL)	Displacement (in pixels) along X or Y axis below which the IIR filter input weight is set to XY_FILT_IIR_COEFF for filtering along the corresponding axis when Charger Armor is active.	0 – 255 (default = 25)
Scan Filtering: XY_FILT_IIR_THR_FAST_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_IIR_THR_FAST** 8168X (TSG6XL)	Displacement (in pixels) along X or Y axis above which the IIR filter input weight is set to XY_FILT_IIR_FAST_COEFF for filtering along the corresponding axis when Charger Armor is active.	0 – 255 (default = 200)
Scan Filtering: XY_FILT_Z_IIR_COEFF_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_Z_IIR_COEFF** 8168X (TSG6XL)	Weighting of the input Z value in the IIR filter when Charger Armor is active.	1, 1/2, 1/4 (default = One Half)
Scan Filtering: XY_FILT_PREDICTION_COEF_CA CYAT6165/8165 (TSG6L) **Charger Armor: CA_XY_FILT_PREDICTION_COEF** 8168X (TSG6XL)	Weight of the predicted position that is based on the velocity when Charger Armor is active.	0 – 128 (default = 0)

8.8.2 Fast finger tracking

The touch controller can have difficulties differentiating a fast-moving finger from two fingers that are alternately tapping on and off the panel. The touch controller just sees the touch being removed from one position and then appearing somewhere else. The touch controller does not know if this is a single touch moving, or one touch being removed and a new touch added.

Figure 73 shows examples of errors. The left image is two fingers tapping, but the touch controller thinks it is a single fast-moving touch. The middle and right images show a single fast-moving finger, but the touch controller thinks it is two separate fingers.

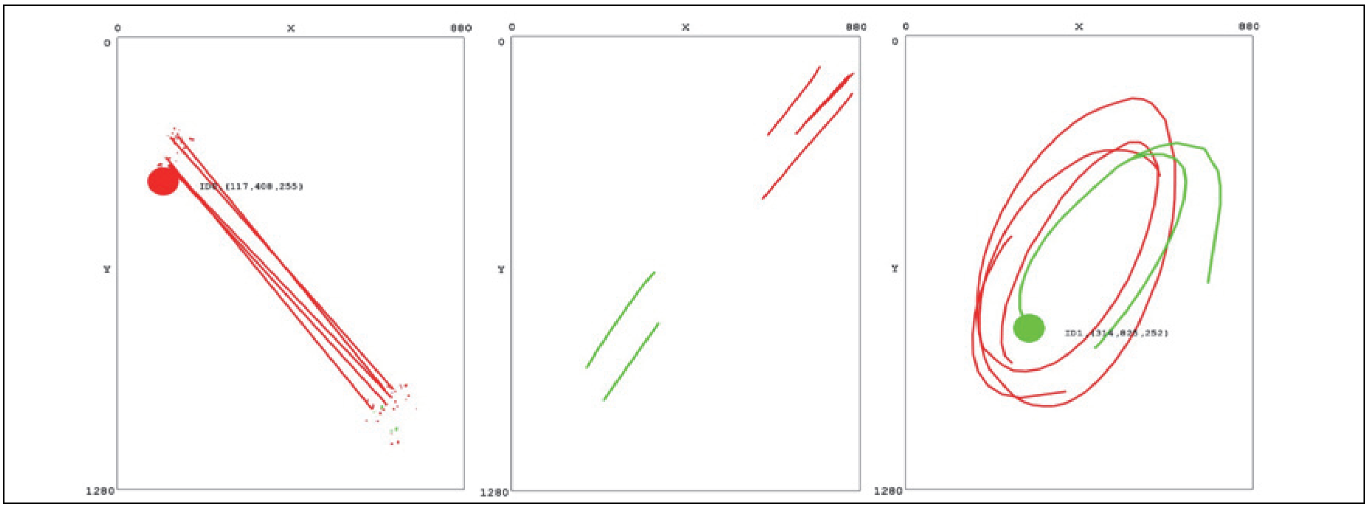


Figure 73 Fast single finger tapping and finger movement recognized as multiple touches

When the errors illustrated by Figure 73 occur, adjustments can be made using the parameters **Fingers: MAX_VELOCITY_SQR**, **Fingers: FINGER_ID_MAX_FINGER_ACCELERATION2**, **Fingers: FAST_SWIPE_LINE_PROTECT_DISTANCE**, **Fingers: FINGER_LIFTOFF_DEBOUNCE**, and **Device Setup: ACT_INTRVL0**.

Ideally, set both the liftoff debounce and the active refresh rate to zero (scan as fast as possible). However, the refresh rate is fixed for most projects.

The main tuning parameters are the max velocity and the max acceleration. These parameters are the thresholds to distinguish a fast single-finger movement from two finger tapping. If a touch has been on the panel for less than 2 scans, then the absolute movement speed is compared to the max velocity parameter. If a touch has been on the panel for at least 2 scans, then the speed is compared to the delta from the previously calculated speed, and then compared to the max acceleration parameter.

The max velocity is measured in pixels/ms. It should be set the maximum supported finger speed in for the project (for examples 2 m/s), see Equation 42. The maximum acceleration should be set to about half of the maximum velocity, however, the requirements of projects differ.

$$\text{Max Velocity Squared} = (\text{Finger Velocity} \cdot \text{Device Setup: ACT\_INTRVL0})^2$$

Equation 42

For example, a project has a pixel size of 0.2 mm, a refresh rate of 100 Hz, and a maximum speed requirement of 2 m/s. The maximum velocity is 10K pixels/s, which is 100 pixels/scan, so the max velocity is 10K, and the max acceleration is 5K.

A common problem of fast finger flick is that only a portion of the line is drawn when a finger runs across the panel. If the issue persists after optimizing the refresh rate, try the following recommendations one at a time until performance is acceptable.

- Reduce (or disable) lift-off suppression, see Section 8.8.7.2.
- Reduce the XY data IIR filtering:
  - Increase the XY IIR filter coefficient, or even set it to 1.
  - Larger coefficients mean the previous data has less influence on the current XY data.
- Reduce the raw-data IIR filtering:
  - Increase the raw data IIR filter coefficient.
  - Larger coefficients mean the previous data has less influence on the current raw-count.
  - Reduce the raw data IIR filter threshold to at least 50% of the finger liftoff threshold
- Disable the XY data IIR filter.

The **TSS: MTX_ORDER** can also impact fast finger tracking, see Section 3.3.5 for full details.

Although these changes enhance fast finger flick tracking, they may affect noise performance and touch position consistency.

Broken lines for fast swipes can also be caused by a side-effect of water rejection. See Section 8.11.1 for details.

8.8.2.1 Parameters

Table 39 Fast finger tracking parameters

Configurable Parameter	Description	Selection
Device Setup: ACT_INTRVL0	Interval time before scan in Active mode (in ms).	0 – 250 (default = 9)
Scan Filtering: XY_FILTER_MASK: XY IIR Filter Enable	Enables or disables the XY IIR filter.	Enabled / Disabled (default = Enabled)
Scan Filtering: XY_FILT_IIR_COEF_SLOW	Weighting of the input X and Y values in the IIR filter for displacements less than XY_FILT_SLOW_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Quarter)
Scan Filtering: XY_FILT_IIR_COEF_FAST	Weighting of the input X and Y values in the IIR filter for displacements greater than XY_FILT_FAST_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Raw Processing: MC_RAW_IIR_COEF	Determines IIR filter order for mutual-cap in base state.	1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Fingers: MAX_VELOCITY_SQR	Square of the finger displacement threshold for distinguishing fast finger movement from separate finger touches (in pixel^2).	0 – 4294836225 (default = 202500)
Fingers: FINGER_ID_MAX_FINGER_ACCELERATION2	"Speed" (maximum distance change in X or Y direction per refresh interval in sensors) threshold distinguishing fast object movement from separate object	0 – 4294836225 (default = 160000)

Configurable Parameter	Description	Selection
	touches. Speeds exceeding this value are subject to the ON threshold. Touches below this speed are subject to the OFF threshold. Applicable to Applicable to ALL object types (Finger/Glove/Stylus). Set this parameter to the maximum finger velocity in sensors + 1.	
Fingers: FAST_SWIPE_LINE_PROTECT_DISTANCE CYAT8168X (TSG6XL) only	Note, CYAT8168X (TSG6XL) only. Feature to maintain robust line drawing in the presence of an inconsistent touch. If a lift-off is followed by a touchdown in the next scan that is within the distance specified by this parameter, then the lift-off/touchdown data is discarded, and the touch is consistently reported. Set to zero to disable this feature.	0 – 65535 (default = 0)
Fingers: FINGER_LIFTOFF_DEBOUNCE	Number of consecutive refresh cycles for which a touch must not be detected before the lack of touch is identified as a liftoff.	0 – 63 (default = 3)

8.8.3 Small finger tracking

Small fingers have a smaller diff-count, which means the system may have difficulties in reliably recognizing them. Without proper tuning, small fingers can result in broken tracks, as shown in Figure 74.

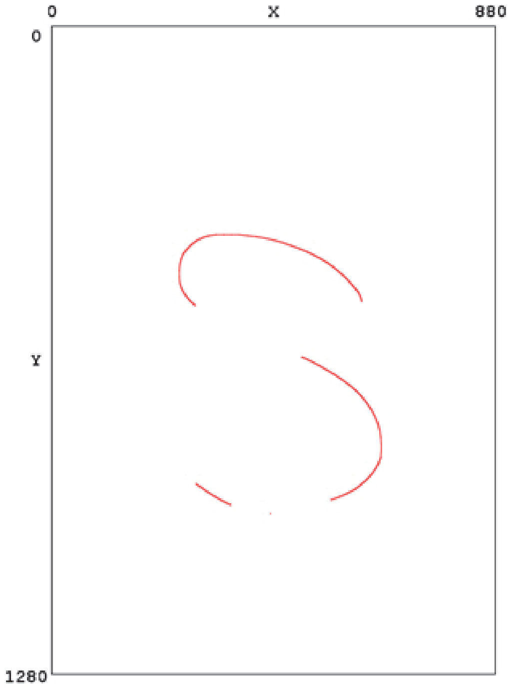


Figure 74 Small finger broken track example

When this is observed, follow these steps one at a time until performance is acceptable:

- Reduce mutual-cap finger threshold:
  - This allows a valid finger to be recognized with a lower diff-count.
  - If the touch is not detected, reduce the touchdown threshold.
  - If the touch is detected, but is then dropped, reduce the lift-off threshold.
- Reduce raw-data IIR filtering:
  - Increase the raw data IIR filter coefficient. A larger coefficient reduces the influence of the previous data.
  - Reduce the raw data IIR filter threshold. Filtering is only applied to changes below the threshold.
- Reduce XY IIR filter filtering:
  - Increase the XY data IIR coefficient. A larger coefficient reduces the influence of the previous data.

8.8.3.1 Parameters

Table 40 Small finger tracking parameters

Configurable Parameter	Description	Selection
Scan Filtering: XY_FILT_IIR_COEF_SLOW	Weighting of the input X and Y values in the IIR filter for displacements less than XY_FILT_SLOW_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Quarter)
Scan Filtering: XY_FILT_IIR_COEF_FAST	Weighting of the input X and Y values in the IIR filter for displacements greater than XY_FILT_FAST_THR.	1, 1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Raw Processing: MC_RAW_IIR_COEF	Determines IIR filter order for mutual-cap in base state.	1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Raw Processing: MC_RAW_IIR_THRESH	IIR threshold for Mutual-cap scan, above which the IIR is bypassed.	0 – 32767 (default = 100)
Fingers: FINGER_THRESH_MUT_HI	Mutual-cap finger touchdown threshold.	0 – 32767 (default = 490)
Fingers: FINGER_THRESH_MUT_LO	Mutual-cap finger threshold for liftoff.	0 – 32767 (default = 400)

8.8.4 Centroid fine-tuning

8.8.4.1 HTI and CTI measurements

Two types of signal measurements are required for the edge finger correction algorithm:

- HTI: Highest Touch Intensity
- CTI: Centered Touch Intensity

HTI measures the peak diff-count when a finger touch is centered on a core sensor, as shown in Figure 75. The HTI signal count measurement is referred to as HTI_{Xmm} in the following section, where X is the finger size in mm.

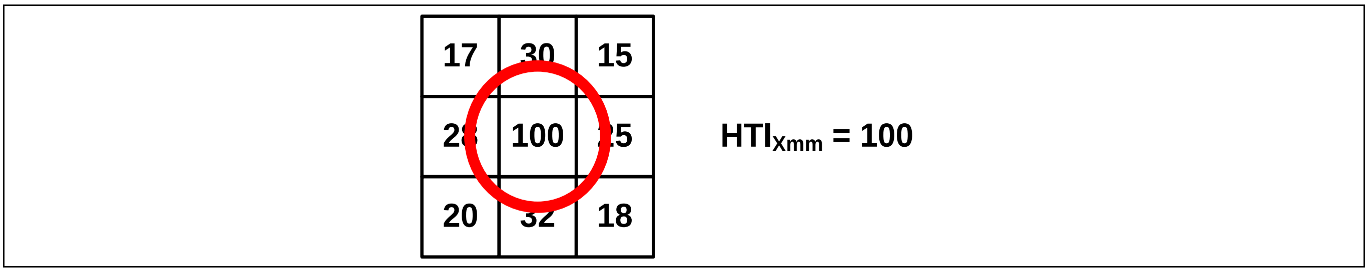


Figure 75 Highest touch intensity (HTI) signal measurement

CTI measures the peak diff-count when a finger touch is centered between the two sensors, as shown in Figure 76. The CTI signal count is the average of the two peak sensors. The CTI signal count measurement is referred to as CTI_{Xmm} in the following section, where X is the finger size in mm.

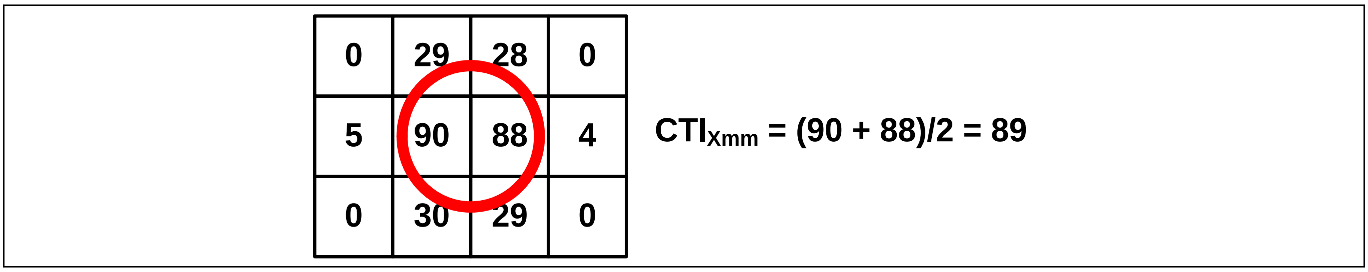


Figure 76 Centered touch intensity (CTI) signal measurement

8.8.4.2 Unit cell linearity

A unit cell, is a single XY intersection, alternatively known as a single mutual-cap sensor, as shown in the heat-map. There is an inherent accuracy error in a unit cell. This error is most easily seen in the accuracy report (1 mm step size). The following 2 graphs show the X-error using the same test module:

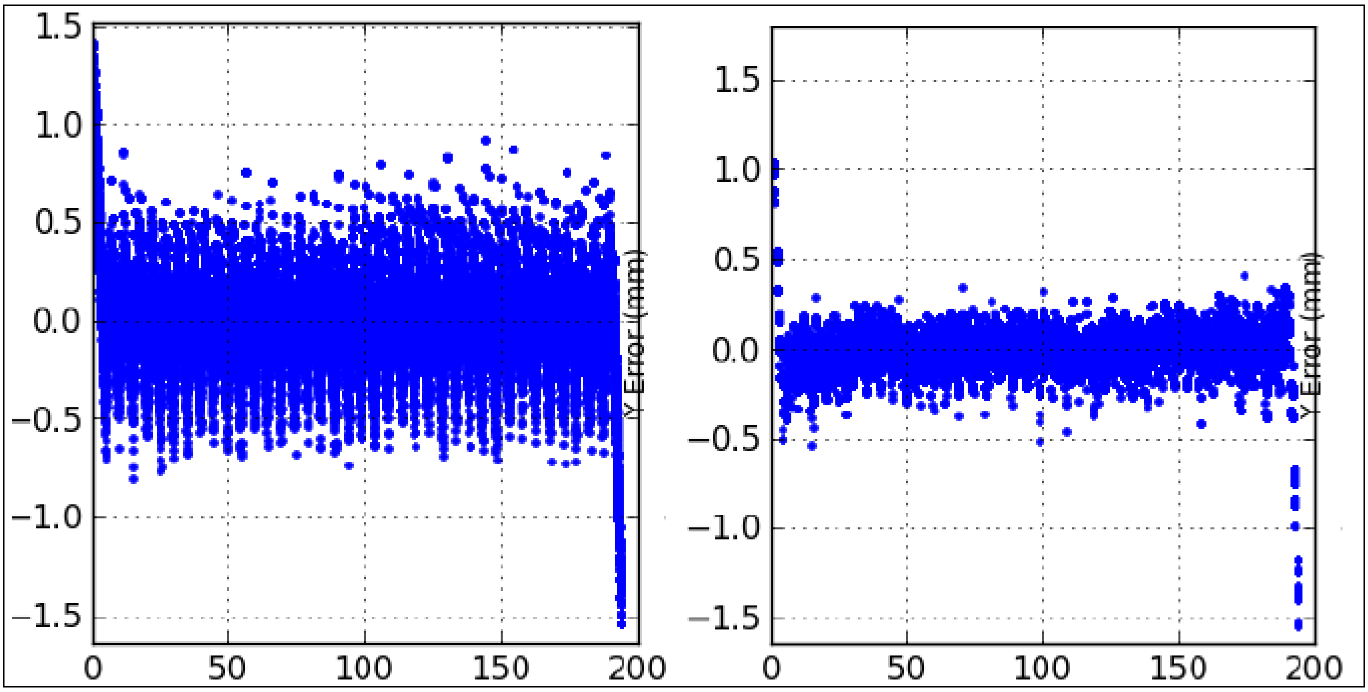


Figure 77 Good and bad unit cell linearity

The unit-cell error can be reduced by tuning the **Device Setup: FINGER_CALC_THRESH** and **Device Setup: GLOVE_CALC_THRESH** parameters. These parameters remove a constant value from all elements in the centroid calculation for finger and glove respectively.

The parameter should be set initially by the following equation. If necessary, the parameter can then be fine-tuned using the test results as a guide.

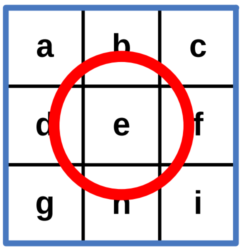
$$\text{Device Setup: xxx\_CALC\_THRESH} = \text{Round\_down}\left(\frac{\text{HTI}_{4\text{mm}} + \text{CTI}_{4\text{mm}}}{2} * 5\%\right)$$

Equation 43

8.8.4.3 Centroid finger size

The centroid algorithm uses the estimated finger size for several calculations. The typical 3x3 sum of different finger sizes must therefore be calculated and stored. The parameters **Device Setup: Z_SUM_8MM** and **Device Setup: Z_SUM_4MM** are used for the upper and lower limits for finger size detection when the touch is detected in the panel core.

Set the "Z_SUM" parameters to the 3x3 Z-sum of the respective finger sizes at the HTI location. The 3x3 sum, is the sum of the peak sensor diff-count and the eight surrounding sensors' diff-counts. This is illustrated in the following figure:



Z_SUM = Σ (a .. i)

Figure 78 Z-sum signal measurement

8.8.4.4 Parameters

Table 41 Centroid fine tuning parameters

Configurable Parameter	Description	Selection
Device Setup: FINGER_CALC_THRESH	Value is subtracted from every element before the centroid calculation. Used for finger objects.	0 – 32767 (default = 35)
Device Setup: GLOVE_CALC_THRESH	Value is subtracted from every element before the centroid calculation. Used for glove objects.	0 – 32767 (default = 35)
Device Setup: Z_SUM_8MM	Sum of the 3x3 sensor diff-count of an 8 mm finger.	0 – 65535 (default = 5200)
Device Setup: Z_SUM_4MM	Sum of the 3x3 sensor diff-count of a 4 mm finger.	0 – 65535 (default = 1270)

8.8.5 Touch threshold hysteresis and touch debounce

8.8.5.1 Touch threshold hysteresis overview

The finger threshold hysteresis is designed to help maintain consistent finger touch recognition. If the touch magnitude is close to the finger threshold, then a small amount of jitter or noise can cause multiple touchdown and liftoff events on the same touch, which is not desirable. The touch hysteresis addresses this issue by using separate mutual-cap thresholds for touchdown and liftoff. For example, the finger touchdown threshold is specified by the parameter **Fingers: FINGER_THRESH_MUT_HI**, and the liftoff threshold is **Fingers: FINGER_THRESH_MUT_LO**. Always set the liftoff threshold lower than the touchdown threshold.

There are separate thresholds for both mutual-cap and self-cap scanning. There are also separate thresholds for buttons (where "N" is the button number from 0-9), gloves, and when charger armor is active:

- Button-n fingers mutual: **Buttons: BTN_LS_ON_THRSH_MUT_n** and **Buttons: BTN_LS_OFF_THRSH_MUT_n**.
- Button-n fingers self: **Buttons: BTN_LS_ON_THRSH_SELF_n** and **Buttons: BTN_LS_OFF_THRSH_SELF_n**.
- Btn-n gloves mut: **Glove Buttons: BTN_HS_ON_THRSH_MUT_n** and **Glove Buttons: BTN_HS_OFF_THRSH_MUT_n**.
- Btn-n gloves self: **Glove Buttons: BTN_HS_ON_THRSH_SELF_n** and **Glove Buttons: BTN_HS_OFF_THRSH_SELF_n**.
- Panel fingers: **Fingers: FINGER_THRESH_MUT_HI** and **Fingers: FINGER_THRESH_MUT_LO**.
- Panel gloves: **Gloves: GLOVES_THRESH_MUT_HI** and **Gloves: GLOVES_THRESH_MUT_LO**.
- Charger armor: **Charger Armor: CA_FINGER_THRESH_MUT_HI** and **Charger Armor: CA_FINGER_THRESH_MUT_LO**.

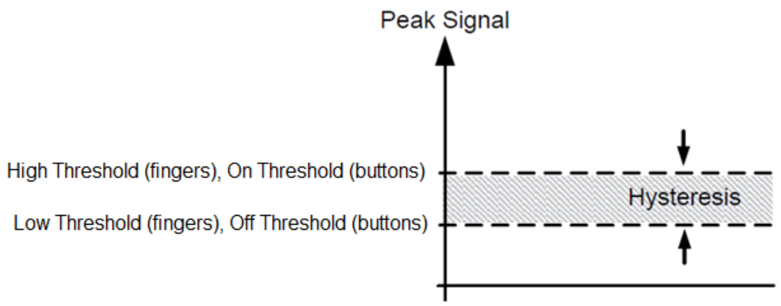


Figure 79 Mutual-cap finger and mutual/self-cap button threshold hysteresis

8.8.5.2 Touch threshold hysteresis tuning

Finger threshold hysteresis is meant to improve the finger touch recognition stability. It is achieved by using a different finger threshold for first touch detection and continuing touch detection.

1. In the Heat Map display mode, set DataType to "DiffCounts".
2. Select a 2x2 area in the center of the panel.
3. Place the smallest supported grounded metal finger over the least touch intensity (LTI) location of the selected sensors. Firmly hold the metal finger on the panel.
4. Set SensorValueType to "Max - Min". Record the maximum count value (Max_Count) of the selected sensors.
5. Set the hysteresis by setting the low threshold using Equation 44.

$$\text{THRESH\_LO} = \text{THRESH\_HI} - \text{Roundup}\left(\frac{\text{Max\_Count}}{2} + 1\right)$$

Equation 44

Note that this should be revisited after tuning the noise mitigation techniques. Such tuning can lower noise allowing for more refined hysteresis tuning.

8.8.5.3 Touch debounce methods

Debounce is a technique requiring that a signal must be in a new state for a given number of samples (or time) before the new state is recognized. Like hysteresis, debounce is designed to enhance the touch detection consistency.

The debouncing of the first touch report is touch object dependent. The touchscreen uses the parameter **Touch Mode: TOUCHMODE_FINGER_SWITCH_DEBOUNCE** and specifies the debounce in milliseconds. Buttons use **Buttons: BTN_LS_TD_DEBOUNCE** (finger) and **Glove Buttons: BTN_HS_TD_DEBOUNCE** (glove) and specifies the debounce in refresh cycles. In all cases, the first finger touch must be detected for this debounce period before it is reported to the host. If charger armor is enabled and noise is detected, then an additional first touch debounce (**Charger Armor: CA_FINGER_FT_DEBOUNCE**) is also applied.

If there are one or more fingers on the panel, the **Fingers: FINGER_MT_DEBOUNCE** parameter specifies the number of consecutive refresh cycles for which the new touch is detected before it is reported as a valid touch.

The parameter **Fingers: FINGER_LIFTOFF_DEBOUNCE** specifies the number of consecutive refresh cycles that the finger touch is not detected before it is identified as a liftoff.

After a finger liftoff, the touch controller remains in finger-mode for fast finger detection until the timeout parameter **Touch Mode: TOUCHMODE_FINGER_EXIT_DELAY** (specified in milliseconds or buttons use **Glove Buttons: GLOVE_BTN_FORBID_DEBOUNCE** and **Glove Buttons: GLOVE_BTN_MODE_SWITCH_DEBOUNCE** which use refresh cycles) is expired. After that, the device enters Active Look-for-Touch state to reduce power consumption. Notice that the timer for both **Fingers: FINGER_LIFTOFF_DEBOUNCE** and **Touch Mode: TOUCHMODE_FINGER_EXIT_DELAY** starts as soon as all fingers have lifted off from the panel.

After the touch controller has entered the LFT state, the **Touch Mode: TOUCHMODE_FINGER_SWITCH_DEBOUNCE** will be applied to the next finger touch (see Figure 80). However, if a valid finger is detected before the Finger mode exit timeout (**Touch Mode: TOUCHMODE_FINGER_EXIT_DELAY**) is expired, then **Touch Mode: TOUCHMODE_FINGER_SWITCH_DEBOUNCE** will not be applied (see Figure 81).

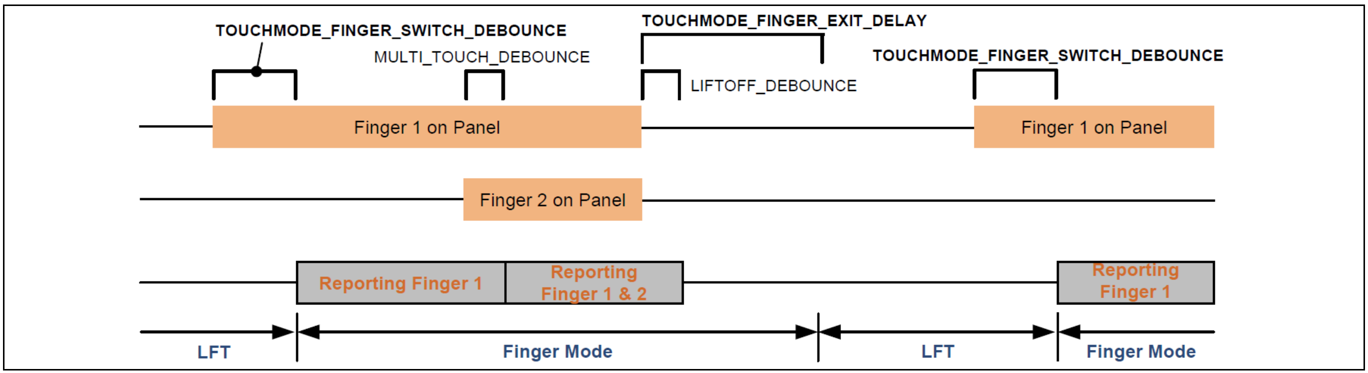


Figure 80 Finger touch debounce illustration 1

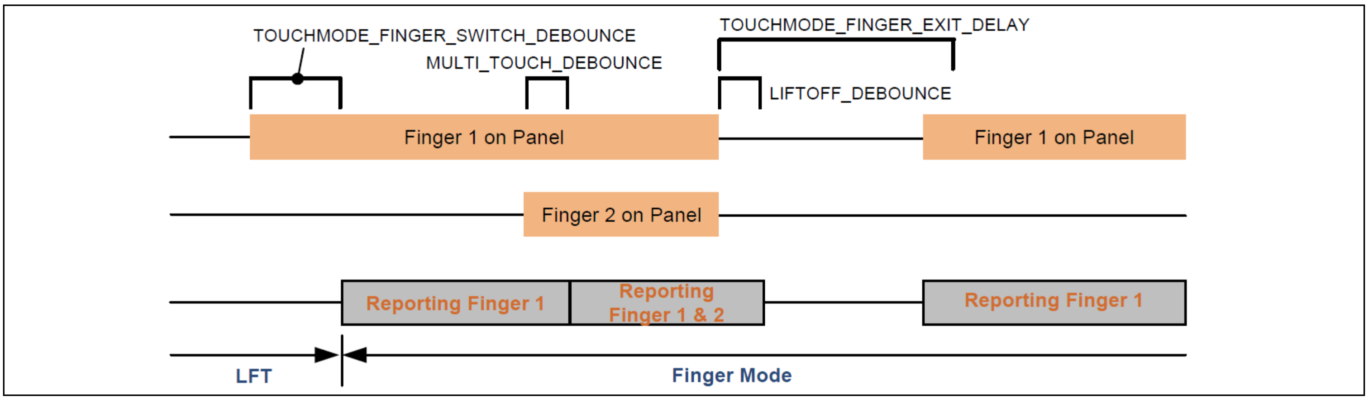


Figure 81 Finger touch debounce illustration 2

8.8.5.4 Parameters

Table 42 Touch threshold hysteresis and touch debounce parameters

Configurable Parameter	Description	Selection
Fingers: FINGER_THRESH_MUT_HI	Mutual-cap finger touchdown threshold.	0 – 32767 (default = 490)
Fingers: FINGER_THRESH_MUT_LO	Mutual-cap finger threshold for liftoff.	0 – 32767 (default = 400)
Fingers: FINGER_LIFTOFF_DEBOUNCE	Number of consecutive refresh cycles for which a touch must not be detected before the lack of touch is identified as a liftoff.	0 – 63 (default = 3)
Touch Mode: TOUCHMODE_FINGER_SWITCH_DEBOUNCE	Debounce in ms for switching from LFT to Finger mode. It specifies the time for which a finger touch must be detected before it is reported to the host.	0 – 65535 (default = 0)
Touch Mode: TOUCHMODE_FINGER_EXIT_DELAY	Delay in ms for switching from Finger mode to LFT. It specifies the time for which the device continues the full active scan for fast finger detection.	0 – 65535 (default = 300)
Gloves: GLOVES_THRESH_MUT_HI	Mutual-cap Glove Threshold ON. Used when glove was not detected yet.	0 - 32767 (default = 180)
Gloves: GLOVES_THRESH_MUT_LO	Mutual-cap Glove Threshold OFF. Used when glove was previously detected.	0 - 32767 (default = 150)
Buttons: BTN_LS_ON_THRSH_MUT_n	Button mutual-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 80)
Buttons: BTN_LS_OFF_THRSH_MUT_n	Button mutual-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 70)
Buttons: BTN_LS_ON_THRSH_SELF_n	Button self-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 65)

Configurable Parameter	Description	Selection
Buttons: BTN_LS_OFF_THRSH_SELF_n	Button self-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 40)
Buttons: BTN_LS_TD_DEBOUNCE	Number of consecutive refresh cycles for which button event must be detected prior to being reported.	0 – 255 (default = 0)
Glove Buttons: BTN_HS_ON_THRSH_MUT_n	Button mutual-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 20)
Glove Buttons: BTN_HS_OFF_THRSH_MUT_n	Button mutual-cap scan liftoff threshold for High Sensitivity Mode (Glove)for button N.	5 – 32767 (default = 10)
Glove Buttons: BTN_HS_ON_THRSH_SELF_n	Button self-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 30)
Glove Buttons: BTN_HS_OFF_THRSH_SELF_n	Button self-cap scan liftoff threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 10)
Glove Buttons: BTN_HS_TOUCHDOWN_DEBOUNCE	Number of consecutive refresh cycles for which button event must be detected prior to being reported.	0 – 255 (default = 1)
Glove Buttons: GLOVE_BTN_FORBID_DEBOUNCE	Number of consecutive refresh cycles after all regular touch liftoff for which a glove touch is not detectable.	0 – 255 (default = 10)
Glove Buttons: GLOVE_BTN_MODE_SWITCH_DEBOUNCE	Number of consecutive refresh cycles for mode switch from finger mode to glove mode.	0 – 255 (default = 1)

8.8.6 Finger tracking on panel edge

8.8.6.1 Overview

When a finger is near or at the panel edge, the part of the finger that is outside of the sensors is not detected. The centroid algorithm will incorrectly calculate the center of the finger due to the missing information. The virtual sensing method is employed by the touch controller to minimize this issue. It uses the detected finger signal and finger size information (if available) to create virtual sensors that mimic the missing finger information.

When the finger peak signal is detected on the panel edge, as shown in Figure 82, two columns of virtual sensors (V1 and V0) are created using the existing detected finger signal and finger size information to project the finger's touch zone.

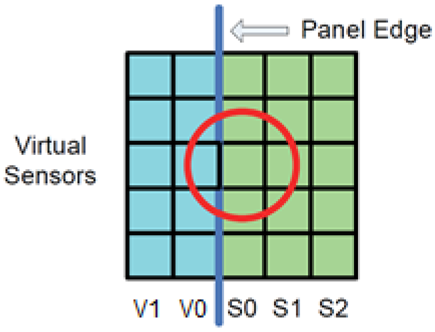


Figure 82 Virtual sensors: panel edge finger

Note that the XY filter settings also have an effect if the touch is moving. It is therefore recommended to perform edge tuning with the XY filters disabled.

8.8.6.2 Edge gain parameters

This section does not describe the tuning of the edge gain parameters. It is recommended to leave these values at their default. However, if satisfactory tuning cannot be achieved with the recommended tuning method, these parameters provide a different influence over the centroid calculation.

There are two edge gain parameters (inner and outer edge gain) for normal fingers and two additional parameters for gloves.

The outer edge gain is a multiplier for the second virtual sensor (sensor furthest from the panel edge, or column-V1). This parameter is the simplest conceptually. Increasing this parameter will attract the reported touch location to the edge of the panel. So, increase this parameter when the reported touch location does not reach the edge of the panel as fast as the real touch position. Decreasing this parameter will resist the reported touch location from reaching the edge of the panel. So, decrease this parameter when the reported touch location reaches the edge of the panel before the real touch position.

The inner edge gain is a multiplier for the first virtual sensor (sensor next to panel edge, or column-V0). This parameter has two effects.

Firstly, when the inner virtual sensor (column-V0) becomes non-zero, and the outer virtual sensor (column-V1) remains at zero, then the effect of the inner edge gain parameter is the same as the effect of the outer edge gain parameter described previously.

Secondly, when both virtual sensors are non-zero, the purpose of the inner virtual sensor (column-V0) is to smooth the transition of the touch to the panel edge. This tuning value is panel dependent, and should be tuned using robot accuracy testing at multiple positions on all 4 panel edges.

As mentioned at the start of this section, for most panels, these parameters should be left at their default values.

8.8.6.3 Edge parameter tuning: offset S1 and S2

When a finger moves off the panel edge, a small diff-count is still observed in the adjacent columns (S2 and S1) due to the fringe effect. The process of a finger moving off the panel is shown in Figure 83:

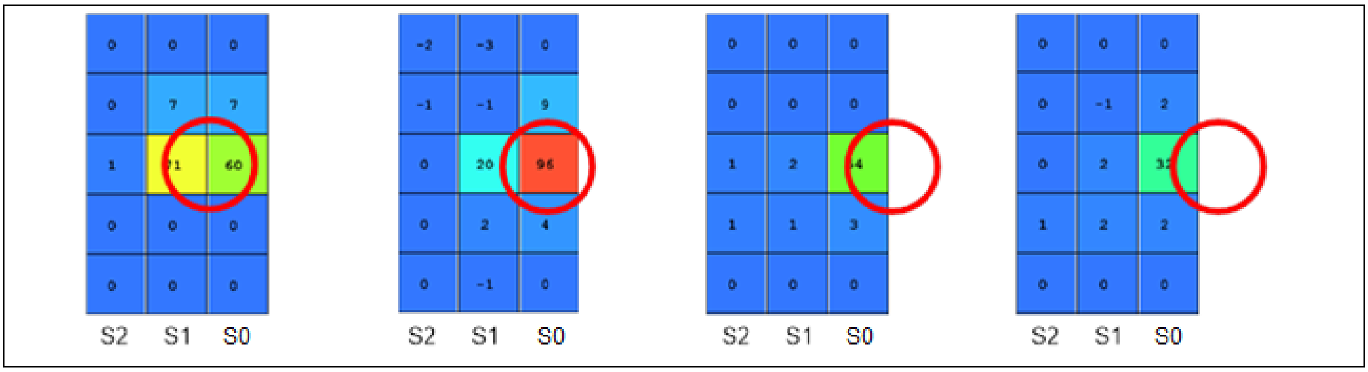


Figure 83 Diff-counts of a finger leaving the panel

For the two cases on the right of Figure 83, the diff-count on the S1 column remains about 2, even though the finger touch has already moved away from the sensor. This error often affects the edge finger centroid calculation and prevents the finger touch from reaching the panel edge.

The parameters **Device Setup: OFFSET_S1** and **Device Setup: OFFSET_S2** are used to remove this residual signal. Note that the equation below uses the HTI and CTI values defined in Section 8.8.4.1 (entitled HTI and CTI measurements). The tuning procedure is as follows:

1. Initialize the parameters using the following equation:

$$\text{Device Setup: OFFSET\_Sx} = \text{round\_down}\left(\frac{\text{HTI}_{4\text{mm}} + \text{CTI}_{4\text{mm}}}{2} * 5\%\right)$$

Equation 45

2. Place the TTHE into line drawing mode.
3. Affix a ruler or a straight edge at about 45 degrees on the panel.
4. Using both 5 mm and 8 mm metal fingers, run the test fingers from the center of the panel towards the edge along the ruler at medium speed.
5. If the line drawn bounces back from the edge as the test finger departs from the panel, or if the line drawn does not reach the panel edge (shown in Figure 84), increment both parameters.

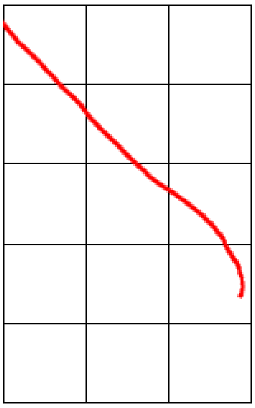


Figure 84 Offset S1/S2 too low

8.8.6.4 Edge parameter tuning: high and low pivot

The high and low pivot parameters are used to define signal adjustment when a finger is on the edge of the panel. The goal of tuning the high and low pivot parameters is to enhance the finger tracking linearity at the panel edge. This is best represented by a finger approaching the panel edge at a 45-degree angle.

Non-linearity needs tuning as either of the two virtual sensors become significant in the centroid calculation. The two locations where the virtual sensors become significant can be seen in the examples (in Figure 85). The first location can be most easily seen in the right-most two examples where the finger track is attracted to the edge too quickly (first virtual sensor is too strong). The second location can be most easily seen in the third and fourth examples where the finger track remains away from the edge too long before quickly snapping to the edge (second virtual sensor is too weak).

Examples of different settings for the high and low pivot parameters are shown in Figure 85. Note that **Device Setup: HIGH_PIVOT** must always be set higher than **Device Setup: LOW_PIVOT**. Generally:

- If the low pivot setting is too low, the finger track will have difficulty reaching the panel edge.
- If the low pivot setting is too high, the finger track will reach the edge too soon.
- If the high pivot setting is too low or too high, the finger track will appear non-linear.

The tuning procedure is as follows:

1. Initialize the parameters using the following equations:

$$\text{Device Setup: HIGH\_PIVOT} = \text{Device Setup: Z\_SUM\_4MM} * 90\%$$

Equation 46

$$\text{Device Setup: LOW\_PIVOT} = \text{Device Setup: HIGH\_PIVOT} * 60\%$$

Equation 47

2. Place the TTHE into line drawing mode.
3. Affix a ruler or a straight edge at about 45 degrees on the panel.
4. Using both 5 mm and 8 mm metal fingers, run the test fingers from the center of the panel towards the edge along the ruler at medium speed.
5. Adjust the high and low pivot values according to the type of error seen and Figure 85:

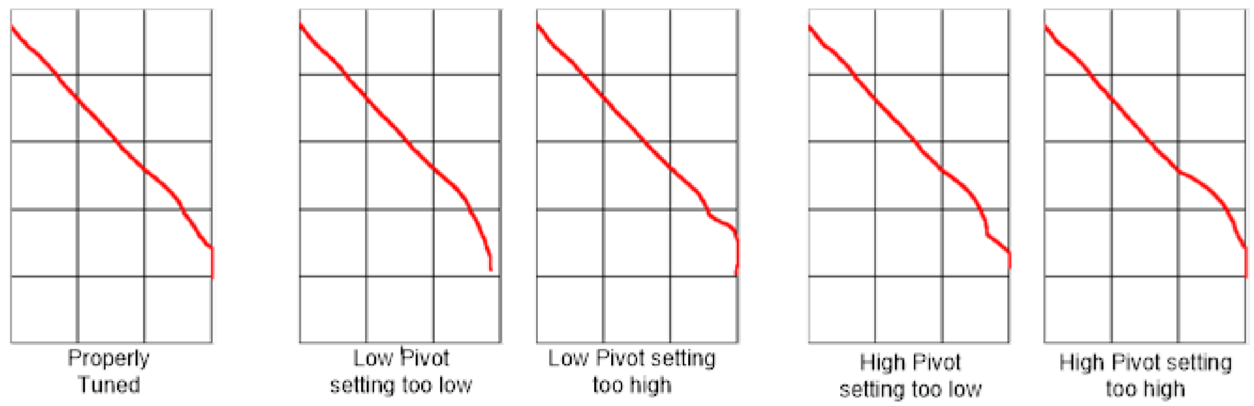


Figure 85 High and low pivot tuning

8.8.6.5 Center magnitude scale (CYAT6165/8165 (TSG6L) only)

This parameter sets the scaling factor on edge finger size detection. Calculate **Device Setup: CENTER_MAGNITUDE_SCALE** using the following equation:

$$\text{Device Setup: CENTER\_MAGNITUDE\_SCALE} = \left(\frac{HTI_{8mm} + CTI_{8mm}}{2*90}\right) * 110$$

Equation 48

8.8.6.6 Corner gain

The **Device Setup: CENTROID_CORNER_GAIN** parameter defines the gain factor which pulls a touch into the corner. This gain factor helps the situation where a touch is unable to reach the corner. To pull a touch into a corner, increase the gain factor. Decrease the gain factor if the touch is being pulled to the corner too quickly.

8.8.6.7 Parameters

Table 43 Edge tuning parameters

Configurable Parameter	Description	Selection
Device Setup: OFFSET_S1	Offset 1 value is subtracted from the second sensor from the edge.	0 – 32767 (default = 50)
Device Setup: OFFSET_S2	Offset 2 value is subtracted from the third sensor from the edge.	0 – 32767 (default = 25)
Device Setup: LOW_PIVOT	When zMagnitude = LowPivot then Scalar = 1x.	0 – 65535 (default = 685)
Device Setup: HIGH_PIVOT	When zMagnitude = HighPivot then Scalar = MinGain (based on finger size).	0 – 65535 (default = 1143)
Device Setup: CENTER_MAGNITUDE_SCALE CYAT6165/8165 (TSG6L) only	CYAT6165/8165 (TSG6L) devices only. Scale for center Magnitude (CENTER_MAGNITUDE_SCALE) used in EstimateCurve API. Rarely changed from default value.	0 – 32767 (default = 700)
Device Setup: CENTROID_CORNER_GAIN	Centroid corner gain multiplied by 256. Can be in range from 0 to 16383. The higher the gain the more position will be pulled into the corner. Rarely changed from default value.	0 – 16383 (default = 512)
Fingers: FINGER_INNER_EDGE_GAIN	Finger Inner Virtual Sensor Gain (in 1/128 increments) for panel edge finger touch correction. Rarely changed from default value.	0 – 255 (default = 8)
Fingers: FINGER_OUTER_EDGE_GAIN	Finger Outer Virtual Sensor Gain (in 1/128 increments) for panel edge finger touch correction. Rarely changed from default value.	0 – 255 (default = 120)
Gloves: GLOVES_INNER_EDGE_GAIN	Glove Gain for Edge Correction (Inner Virtual Sensor) in 1/128 increments for panel edge glove touch correction.	0 - 255 (default = 8)

Configurable Parameter	Description	Selection
	Rarely changed from default value.	
Gloves: GLOVES_OUTER_EDGE_GAIN	Glove Gain for Edge Correction (Outer Virtual Sensor) in 1/128 increments for panel edge glove touch correction. Rarely changed from default value.	0 - 255 (default = 120)

8.8.7 Touch reporting control

Not all touch reports are wanted. Extra algorithms are provided to customize the reports required for a particular project. These algorithms are:

- Basic Active Distance
  - Filter touch reports, so only significant movement is reported.
- Adaptive Liftoff Active Distance
  - This is a legacy feature, use last touch suppression below.
- First and Last Touch Suppression
  - Ignore touches while approaching or leaving the panel.

8.8.7.1 Basic active distance algorithm

The basic active distance algorithm prevents a stationary finger touch from being continuously reported to the host. The touch controller will only generate a new touch report when significant movement is detected.

There are two parts to the active distance algorithm (see Figure 86):

- First report after the initial touchdown.
- All subsequent touches while the touch is on the panel.

The active distance is tracked separately for each finger. A touch report will contain the information for each touch, but only the touches that have triggered the new report will contain updated information. For example, two new touchdowns occur, and one touch moves over the active distance, while the other touch only moves half. A new touch report will be triggered and the 1st touch will be reported with an event type of “Further the Active Distance” and the new touch location, while the 2nd touch will be reported with an event type of “No Event” and the previous touchdown touch location (despite the finger having moved).

After the initial touchdown is reported, the active distance threshold **Fingers: ACT_DIST0_SQR** applies. The touch movement (squared) must be greater or equal to this threshold for the first movement report to be generated for a finger. Any movement less than this threshold will not trigger a touch report.

After the first movement report, subsequent reports are generated when the displacement (squared, in pixels) is greater than or equal to **Fingers: ACT_DIST2_SQR**. Note that the active distance is tracked independently for each touch, and despite all touches being included in each touch report, the touch location is only updated when the active distance requirement is met.

Note that the parameters use squared distances to reduce the algorithm complexity and therefore increase performance.

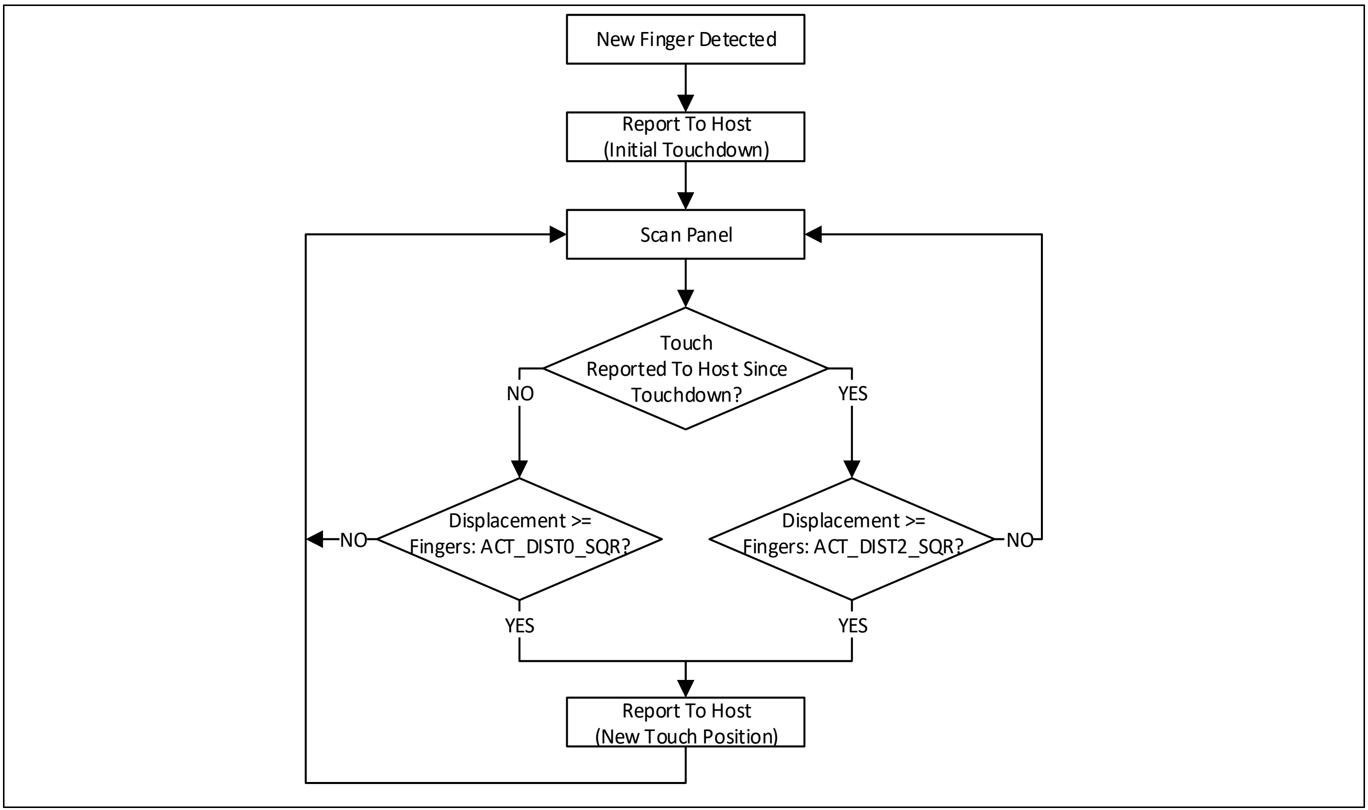


Figure 86 Active distance flow chart

8.8.7.2 First and last touch suppression

It is important to only report touches that are in contact with the module for two main reasons. Firstly, it is annoying for the user when a touch is detected too early. Secondly, when a touch is further from the panel surface, the signal is weaker, and therefore the accuracy is worse. This is most significant for a liftoff, where the liftoff location is used by the UI for many applications.

The first touch suppression algorithm reuses the multi-touch debounce (**Fingers: FINGER_MT_DEBOUNCE**) parameter. It is recommended to leave this parameter at the default. The multitouch debounce also applies to the first touch while the touch's Z-value is less than **Fingers: FINGER_FT_Z_THRESH**. It is recommended to set this value to 90% of the Z-value of the smallest finger supported (usually a 5 mm finger). To disable the first touch suppression filter, set **Fingers: FINGER_FT_Z_THRESH** to 0.

The last touch suppression algorithm will suppress a touch report if the following conditions are all true:

- Z-value drops by more than **Fingers: FINGER_LT_Z_THRESH**
- Previous movement < **Fingers: FINGER_LT_ACT_DIST_SQR**
- Current movement < **Fingers: ACT_DIST_LIFTOFF_SQR**

This algorithm can be set as aggressively as required for the particular project. To disable the last touch suppression feature, set **Fingers: FINGER_LT_Z_THRESH** to 255.

8.8.7.3 Parameters

Table 44 Touch reporting control parameters

Configurable Parameter	Description	Selection
Fingers: FINGER_MT_DEBOUNCE	Number of consecutive refresh cycles for which a touch must be detected before being reported. Always applies to the second and successive touches. Can apply to the first touch depending on the first touch suppression feature.	0 – 63 (default = 2)
Fingers: FINGER_FT_Z_THRESH	Used in the first touch suppression algorithm. If a new unreported touch has a Z-value less then this value, the touch is debounced by **Fingers: FINGER_MT_DEBOUNCE**. Set to 0 to disable first touch suppression.	0 – 255 (default = 80)
Fingers: ACT_DIST0_SQR	Squared active distance threshold (in pixels) applied only to movement from touchdown to mask unwanted jitter. Movement (squared) greater than this threshold will trigger a new touch report. After the first touch report after touchdown, the displacement threshold **Fingers: ACT_DIST2_SQR** is applied.	0 – 65535 (default = 0)
Fingers: ACT_DIST2_SQR	Squared active distance threshold (in pixels) applied to movement to mask unwanted jitter. Note the initial movement after touchdown uses the **Fingers: ACT_DIST0_SQR** threshold. Movement (squared) greater than this threshold will trigger a new touch report.	0 – 65535 (default = 0)
Fingers: ACT_DIST_LIFTOFF_SQR	Used in the last touch suppression algorithm. If an existing touch's current movement (between the last report and current scan) squared is less than this threshold, amongst other requirements (see last touch suppression section), then the touch is identified as leaving the panel and the touch report is suppressed. Set to 0 to disable last touch suppression.	0 – 65535 (default = 0)
Fingers: FINGER_LT_ACT_DIST_SQR	Used in the last touch suppression algorithm. If an existing touch's previous movement (in last touch report) squared is less than this threshold, amongst other requirements (see last touch suppression section), then the touch is identified as leaving the panel and the touch report is suppressed. Set to 0 to disable last touch suppression.	0 – 65535 (default = 0)
Fingers: FINGER_LT_Z_THRESH	Used in the last touch suppression algorithm. If an existing touch's Z-value decreases greater than this threshold less then this value, amongst other requirements (see last touch suppression section), then the touch is identified as leaving the panel and the touch report is suppressed. Set to 255 to disable last touch suppression.	0 – 255 (default = 255)

8.8.8 Hard press finger

A hard press on the panel can cause the panel to deform, which can be an issue for panels that are made of flexible material, or for panels with an air gap between the touch sensor and the display. A hard press can bend the touch sensor, reducing the distance to the display or other substrate material underneath. This can affect the mutual-cap and self-cap on the sensors that are not being touched. The change of capacitance resembles the touch signature, introducing false touches in the untouched area during the hard press event.

If the false touch's mutual-cap raw-count is significantly smaller than the hard press finger signal, then enabling the dynamic threshold touch filter (**Fingers: PEAK_IGNORE_COEF**) to remove the unintended false touches. This dynamic threshold is based on the current maximum signal peak, as shown in the following equation. Touches with mutual-cap signal peak value below this dynamic threshold are ignored and not processed.

$$\text{Dynamic Threshold} = \text{Max}_{\text{Peak}} \cdot \frac{\text{Fingers: PEAK\_IGNORE\_COEF}}{32}$$

Equation 49

Fingers: PEAK_IGNORE_COEF can be set between 0 and 32. Setting it to 0 (default) disables this feature. When multiple touches are on the panel, a large coefficient can remove smaller touches. In general, coefficients larger than 16 are not recommended. When tuning this value, make sure to test for small finger tracking while another typical-size grounded touch is on the panel.

8.8.8.1 Parameters

Table 45 Finger tracking parameters

Configurable Parameter	Description	Selection
Fingers: PEAK_IGNORE_COEF	Dynamic threshold coefficient for hard press finger. Local mutual-cap peak larger than the threshold will be ignored and not processed. Setting to 0 disables this feature.	0 – 32 (default = 0)

8.8.9 Metal finger swipe unexpected behavior

On rare occasions it has been reported that a fast-moving metal finger can generate electrostatic charge. This charge will couple to the TX signals and can produce false signal detection, which in turn can result in a false touch report.

To determine if this issue is a problem, put TTHE in Heatmap mode and set the reporting to "Max Hold". Move the metal finger along the RX lines.

If significant false signal is detected along the untouched parts of the TX lines, then use the following methods to reduce the false signal (note that these settings decrease the sensitivity to additional touches):

- Set **Scan Filtering: RXLINE_FILT_ENABLE** to "Always Enable".
- Increase **Scan Filtering: RXLINE_FILT_THRESH**.
- Increase **Fingers: FINGER_MT_DEBOUNCE** (and/or **Gloves: GLOVES_MT_DEBOUNCE**).

8.9 EMI/EMC immunity tuning

A system approach should be applied for improving EMI/EMC emissions performance. For example, a proper PCB/FPC routing, grounding, shielding and using power supply filters. The Module Design Best Practices document 001-05467 gives recommendations for hardware design.

The very robust solution for improving system immunity to external noise is adding external resistors in all sensing lines (TXs and RXs). The typical value of the resistors is 2 kΩ - 2.2 kΩ, but it is hardware dependent.

FW includes a Noise Armor feature that is a powerful tool to deal with external noise. In addition, a Touch Pin is implemented into FW for possible detection of false touches during EMI/EMC emissions tests without external communication of CY816X device with host or TTHE.

Taking into account frequency response characteristics, the Noise Armor is effective only for noise up to 20 MHz. Low frequency noise interferes with TX harmonics. Therefore, it is possible to use noise listener (a part of Noise Armor) in this frequency range to measure the noise level and to trigger appropriate noise abatement states.

The system can be sensitive to the high frequency noise (above 20 MHz), especially if the hardware part of system is designed with high frequency EMI/EMC emissions requirements in mind. Parasitic impedances in ground paths, ineffective bypass capacitors or power design issues could cause additional huge frequency responses on some high frequencies not correlated with TX harmonics. Special very high frequency components and design solution are required to avoid external noise interference above 20 MHz.

8.9.1 EMI test mode

Sometimes, during EMI/EMC tests, it is not easy to connect the touchscreen IC and host (TTHE). Ideally, an optical USB link should be used for this purpose. As an alternative, the device can be put into an EMC test mode by setting the parameter **Device Setup: EMI_TEST_MODE_DELAY_TIME** to a non-zero value. In EMC Test Mode, the controller operation is changed in the following ways:

It goes directly into operating mode from startup. There is no need to exit Bootloader mode.

One of the device's GPIO's is set up as a Touch Pin. For CYAT8168 (TSG6XL) devices, the Touch Pin is P1[2]. For CYAT8165 (TSG6L) devices, it's P1[1].

The Touch Pin is intended to drive the cathode of an external LED, set up to be visible from outside the ESD chamber.

At startup, the Touch Pin outputs three square-wave pulses with a pulse width of about 300 ms. The three-pulse event can be used to detect a reset of the system.

During operation, the Touch Pin is driven high whenever a touch event is detected on the panel. The LED can therefore be used to check for false touches. The Touch Pin is driven low after no touches are seen on the panel, after a delay controlled by **Device Setup: EMI_TEST_MODE_DELAY_TIME**. The parameter specifies the delay in increments of 100 ms.

8.9.2 Tuning of noise immunity methods

The device has a collection of noise abatement features. One part of them is passive and can be enabled/disabled only by configuration parameters, like Raw-Count filters. Another part is active and will be activated only if external noise is detected by the Noise Armor algorithm. Two consecutive actions are possible after the Noise Detection: frequency hopping and - if no quiet frequency is found - touch cancelation. Touch cancelation suppresses all touches if high noise is detected. Frequency hopping changes scanning TX frequency according to noise conditions.

8.9.3 Noise metrics report mode

In normal operation, the touchscreen system only generates a report to host if a touch event is detected. But if the Listener is enabled in TTHE by clicking View, Tool Windows, Listener, then the system will report noise metrics every 200 ms. For more frequent reports, set the parameter **Device Setup: REPORT_CFG: Noise Water metrics output enable** to “Enabled”, and **Device Setup: REPORT_CFG: Metrics output mode** to “Noise”. Doing this will cause the system to refresh the noise data after each scan.

8.9.4 Passive noise immunity tuning

Passive noise immunity features:

- Common Mode Filter.
- IIR filter.

The most important passive tuning method for EMI/EMC immunity is the Common Mode filter. The filter is very helpful to improve false touches immunity to noise radiated by an external antenna.

To be effective **Raw Processing: MC_RAW_CMF_THRESH** and **Raw Processing: SC_RAW_CMF_THRESH** should be set higher than the noise seen in the immunity test. The response from antenna can be bigger than response on human touch. In that case, the CMF filter threshold also should be bigger. In the hardware CMF filter, this can cause degradation of touch response due to parasitic filtering (software CMF filter available in CYAT8168X (TSG6XL) only).

Table 46 Parameters for 8.9.EMI/EMC immunity tuning

Configurable Parameter	Description	Selection
Device Setup: EMI_TEST_MODE_DELAY_TIME	The time in units of 100 ms to keep the EMI_Test_Mode TouchPin in the high state after touch detection. Set to 0 t disable EMI_Test_Mode.	0 – 255 (default = 0)
Device Setup: REPORT_CFG: Noise Water metrics output enable	If enabled, touch reports are sent every scan irrespective of touch events. The touch reports contain the noise metric data. If this parameter is disabled, the noise metric data is still available, but must be requested by the host using the “Get Noise Data” command.	Enabled / Disabled (default = Disabled)
Device Setup: REPORT_CFG: Metrics output mode	Type of noise metric to output (if enabled).	Noise / Water (default = Noise)

8.10 Noise Armor

Noise Armor is a set of features to mitigate the effects of low-frequency (typically less than 20 MHz) noise injected into the panel from the user’s finger. In TTHE, Noise Armor is called “Charger Armor” as it was originally developed to overcome charger noise in consumer and commercial devices. In automotive applications, this is rarely an issue because the user and touchscreen share the same ground – the vehicle chassis. Therefore, under normal circumstances, Noise Armor does not need to be enabled or tuned.

There are some cases, however, where Noise Armor can be helpful. For example, if a consumer device such as a cell phone or laptop is charged from a power port in the vehicle, noise from the charger can modulate the vehicle's ground, and touches to the panel will appear to have charger noise. Noise Armor can be used to mitigate the effects of this noise.

Noise Armor can also be used to mitigate noise effects that come from certain display images. Consider a case of a noisy display that only causes problems with some image content. Using Noise Armor, additional filtering and algorithms can be activated for the noisy content only, without affecting performance for normal low-noise images.

Before enabling the Noise Armor, the system should first be tuned for display noise (Section 8.2). Noise Armor is a collection of noise-abatement features targeted at external noise. It contains a listening algorithm that listens specifically for external noise to choose which features to enable. The goal of Noise Armor tuning is to eliminate jitter and false touches in the presence of external noise.

When Noise Armor is activated (noise level above "undetectable"), advanced features (for example water rejection, smart-scan, hover, and glove detection) are disabled. However, CAPSENSE™ buttons will still be processed.

Noise Armor is enabled by setting the parameter **Charger Armor: CHARGER_ARMOR_ENABLE** to "Enabled". When enabled, Noise Armor adds special panel scans between normal mutual-cap scans. It then calculates several noise metrics (described below) based on the results of the noise scans. Therefore, it should be noted that enabling Noise Armor reduces panel refresh rate.

8.10.1 Noise Armor states

There are four Charger Noise Effect Levels. They are shown in the Data Monitor window in the TTHE and are accessed by navigating to **View** > **Tool Windows** > **Data Monitor**. Selecting the Start button will display the Noise Effect Level and other touch information, as shown in Figure 87.

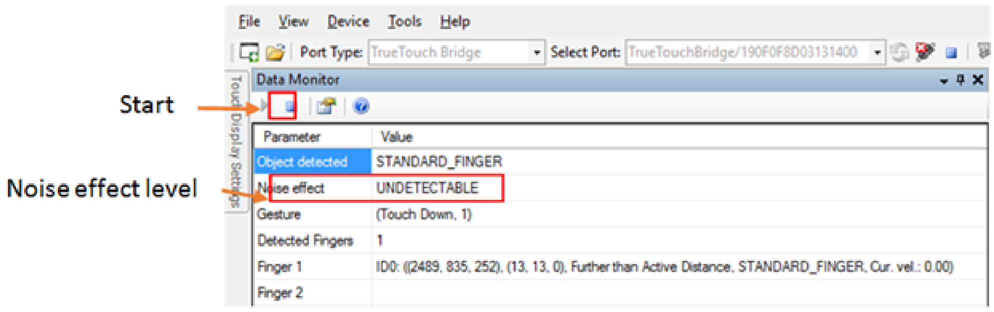


Figure 87 TTHE data monitor

The Noise Effect Levels are summarized in Table 47. It starts from the "UNDETECTABLE" level and sequentially escalates to the next level (up to the "EXCESSIVE NOISE" level) based on the measured noise. Additional tools are enabled every time Noise Armor escalates to the next level. If Noise Armor is at the EXCESSIVE NOISE level, it indicates that Noise Armor has exhausted all the available tools and is still unable to mitigate the injected noise. The Adaptive Frequency Hop (AFH) has cycled through all the alternative TX hop frequencies and is unable to find one that meets the noise impact target.

Noise Armor will escalate to the Excessive Noise level if no quiet frequencies are available in the AFH set. In this case, all touch data is blocked from being sent to the host. This ensures that no false touch data is sent when the device is being affected by an excessive amount of external noise. **Charger Armor: CA_EXCESSIVE_NOISE_DEBOUNCE** determines number of consecutive refresh cycles for which touch report is blocked after detecting Excessive Noise level.

Table 47 Charger noise levels

Noise Effect Level	Description	Behavior
UNDETECTABLE	Noise Armor is turned off (**Charger Armor: CHARGER_ARMOR_ENABLE** = Disabled). OR Noise Armor is enabled, but noise mitigation features are not activated as the detected external noise is not high enough.	Base configuration is used.
LEVEL_2	Noise Armor is enabled. Wideband Noise Metric > **Charger Armor: WB_THRESH**, or Frequency Noise Metric > **Charger Armor: NMF_DETECT_THRESH**, or Injected Touch Noise Metric > **Charger Armor: NMI_THRESH** The conditions above depend on **Charger Armor: CA_TRIG_SRC**	Alternative (high noise mitigating) settings for: Number of TX pulses RX line filter can be enabled Raw-data filters Touch thresholds Base TX frequency is still employed.
LEVEL_3	Level 2 tuning configuration is unable to mitigate the injected external noise. The Injected Touch Noise Metric continues to be above the threshold (**Charger Armor: NMI_THRESH**).	Activate AFH. The device dynamically switches between the 3 alternative TX hop frequencies along with the custom configurations to stay away from the noise impact.
EXCESSIVE NOISE	Noise Armor is unable to mitigate the noise. The Injected Touch Noise Metric continues to be above the threshold after the AFH logic has looped through the alternative TX frequencies the number of times specified in parameter **Charger Armor: AFH_HOP_CYCLES_COUNT**. No available quiet frequencies in the AFH set. Block all touch data to the host during specified parameter **Charger Armor: CA_EXCESSIVE_NOISE_DEBOUNCE**.	No touches are reported.

The following outlines the Noise Armor tuning steps.

- Set up Noise Armor noise metrics.
- Level 2 tuning:
  - Set-up the Raw-count and Touch filters in Noise Armor.
  - Extend the integration time and configure the RX Line Filter.
- Level 3 tuning:
  - Set-up the Alternative Frequency Hop (AFH).
  - Fine tune the AFH configuration options.

Detail of each step is discussed in the following sections.

8.10.2 Noise metrics

After the noise scan, Noise Armor calculates three different noise metrics to measure the external noise. The metrics are summarized in Table 48. The three types of noise metric are: Wideband Noise Metric, Noise Frequency Metric, and Injected Touch Noise Metric. Listener window is shown in Figure 88.

Table 48 Noise metrics

Noise Metric	Description
Wideband Noise Metric	Max of the all noise measured and integrated over a wide frequency band. This metric is enabled if **Charger Armor: CA_TRIG_SRC** is set to "WB" or "WB+NMI". Noise Armor is activated if the metric is enabled and above **Charger Armor: WB_THRESH**. Metric is displayed as "Wideband Noise" in Listener.
Noise Frequency Metric	Percentage ratio between the measured noise period and the current TX period in use. This metric is always enabled. Noise Armor is activated if this metric is above **Charger Armor: NMF_DETECT_THRESH**. Metric is displayed as "NMF" in Listener.
Injected Touch Noise Metric	Jitter-based noise measurement monitors touch jitter, false touch, and dropped touch. It processes the mutual-cap listen scan data and evaluates the signal peak movement. It escalates Noise Armor to the next Noise Effect Level or next hop frequency if metric is above threshold **Charger Armor: NMI_THRESH**. Metric is displayed as "Injected Touch" in Listener. The metric is evaluated at each scan if **Charger Armor: CA_TRIG_SRC** is set to "NMI" or "WB+NMI", or if triggered by the Wideband Noise Metric or Noise Frequency Metric.

Each of the metrics can be displayed in TTHE's Listener Window, shown in Figure 88. Use the following steps to set up the Listener Window:

1. In TTHE, click View, Tool Windows, Listener. This displays the window.
2. Set the listener to display noise data by setting the parameter **Device Setup: REPORT_CFG: Metrics output mode** to "Noise".
3. The system will report noise metrics every 200 ms. For more frequent reports, set the parameter **Device Setup: REPORT_CFG: Noise Water metrics output enable** to "Enabled". Doing this will cause the system to refresh the noise data after each scan. When tuning is complete, it's recommended to reset **Device Setup: REPORT_CFG: Noise Water metrics output enable** back to "Disabled" to prevent excessive noise reports to the host in normal operation.

4. **IMPORTANT**: the listener window only reports the noise metrics if Noise Armor is enabled. If any of the noise metrics exceeds its threshold, then Noise Armor will be activated and the metrics will be affected by the activated Noise Armor settings. Therefore, to get an accurate picture of the noise values, the thresholds must be set such that Noise Armor is not activated, for example:

- **Charger Armor: WB_THRESH** should be set to 32767.
- **Charger Armor: NMF DETECT THRESH** should be set to 255.
- **Charger Armor: NMI THRESH** should be set to 32767.

With these settings, the actual metric values can be seen, and the results can be used to set the thresholds properly in the tuning steps to follow.

5. To start the display in the Listener Window, click the blue start arrow as shown in Figure 88.

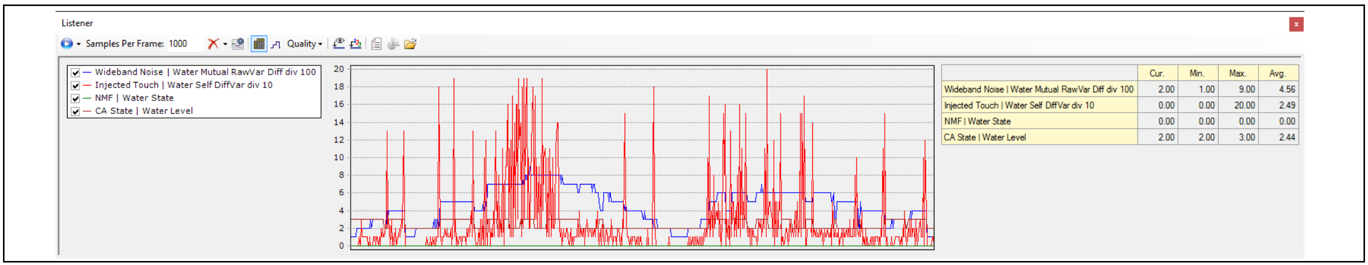


Figure 88 TTHE listener window

Wideband Noise Metric:

The Wideband Noise Metric is used to detect the presence of external noise. It essentially integrates all the noise measured over a wide frequency band and takes the maximum value out of the measured noise count (peak noise detection). This metric has flat response and gain across all frequencies with maximum absolute value of 127. This noise metric is displayed as "Wideband Noise" in the TTHE Listener.

Noise Armor is activated (noise is present) when the measured Wideband Noise Metric exceeds the threshold **Charger Armor: WB_THRESH**. This also enables the Injected Touch Noise Metric to evaluate the noise.

Parameter **Charger Armor: WB_CMF_ENABLE** allows enabling the common-mode filter for Wideband Noise Metric measurement. It enhances the noise metric measurement consistency and prevents false triggering Noise Armor due to display noise injection. Filter is enabled by default.

One potential side effect of this Wideband Noise Metric CMF filter is that it may decrease the noise listening sensitivity for external noise impact that is common across all the RX in the same RX scan slot. One example is a very noisy finger hovering several centimeters above the touch panel. A large panel area is affected, and the injected noise is common to a wide range of sensors. The external noise impact could be filtered out by this CMF filter at the listener. Noise Armor is not activated to mitigate the noise impact and false touches may appear.

Noise Frequency Metric:

The Noise Frequency Metric is the percentage ratio between the measured noise period and the current TX period in use. This noise metric is displayed as "NMF" in the TTHE Listener. The NFM can activate Noise Armor if above threshold **Charger Armor: NMF_DETECT_THRESH**.

Injected Touch Noise Metric:

The Injected Touch Noise Metric is a jitter-based noise measurement method which monitors the touch jitter, false touch, and dropped touch. A fake touch is added in the firmware to monitor the noise impact. It processes the mutual-cap listen scan data and evaluates the signal peak movement. This noise metric is used to determine if Noise Armor should escalate to the next Noise Effect Level and to enable additional features mitigating the external noise. The threshold **Charger Armor: NMI_THRESH** specifies the jitter in distance in units of the percentage of the sensor pitch. This noise metric is displayed as "Injected Touch" in the TTHE Listener.

The noise metric thresholds are summarized in Table 49.

Table 49 Noise metric parameters

Noise Metric	Parameter	Range	Description
-	**Charger Armor: CA_TRIG_SRC**	WB, NMI, or WB+NMI	Defines which noise metric can trigger Noise Armor. The Noise Frequency Metric can always trigger it.
Wideband Noise Metric	**Charger Armor: WB_THRESH**	0 – 32767 (default = 25)	Detect noise presence and start Injected Touch NM.
	Charger Armor: WB_CMF_ENABLE	Disabled / Enabled (default = Enabled)	Common Mode Filter (CMF) enable for Wideband Noise Metric scan. CMF improves Wideband Noise Metric measurement consistency and prevents false triggering Noise Armor due to display noise injection.
Noise Frequency Metric	**Charger Armor: NMF_DETECT_THRESH**	0 – 255 (default = 35)	Start Injected Touch NM.
	Charger Armor: CA_NMF_LIMIT	0 – 127 (default = 10)	Specifies the HI_LIMIT/LO_LIMIT for sample comparison in Listener scan for NMF. Default values HI_LIMIT=10, LO_LIMIT= -10
Injected Touch Noise Metric	**Charger Armor: NMI_THRESH**	0 – 32767 (default 10)	The threshold for Injected Touch NM specifies the jitter in distance in units of the percentage of the sensor pitch (100 equals to one sensor pitch) Noise Armor escalates to the next Noise Effect Level or the next alternative hop frequency if the metric is above this threshold. The metric maximum value is 102. Setting the threshold above 102 disables noise level escalation from the injected touch metric, including AFH. Note that there is a RAM copy of the parameter: CA_NMI_THRESH
	Charger Armor: NMI_TCH_MAGNITUDE	0 – 32767 (default 1000)	Magnitude of the injected touch during the listener scan. This value should be set close to the touch response on the typical finger.
	Charger Armor: NMI_TOUCH_THRESH	0 – 32767 (default = 400)	This parameter is used to detect a false touch error condition resulting from the fake injected touch. If the condition is detected, then the NMI noise calculation is skipped.

Noise Metric	Parameter	Range	Description
			Normally this parameter can be left at it default value. If false touches are reported only when charger armor is enabled, try increasing this parameter. If too high, the NMI noise metric is disabled because the calculation is skipped.
-	**Charger Armor: SC_TRIG_THRESH**	-32768 – 0 (default -32768)	Noise Armor is activated if the minimum Self-cap diff-count signal is below this threshold. Setting threshold to -32768 disables this feature.

To collect better noise statistics (more samples) several listener scans are done. Number of scans is configured by the parameter **Charger Armor: NMI_SCAN_CNT** and **Charger Armor: NM_WB_SCAN_COUNT**.

8.10.3 Noise Armor state transitions

Figure 89 shows the Noise Armor state diagram. Noise Armor can be activated by either of the following two conditions:

- Noise metrics determined by **Charger Armor: CA_TRIG_SRC**.
- The minimum self-cap diff-count signal is below threshold **Charger Armor: SC_TRIG_THRESH**.

The self-cap condition is designed for the situation where the noise is detected with self-cap scan but not with mutual-cap scan. Situation could occur if the self-cap and mutual-cap scanning frequencies are different. And the external noise frequency content overlaps only with the self-cap frequency spectrum but not with the mutual-cap frequency spectrum. If Water Rejection enabled, false liftoff observed in self-cap scanning due to external noise effect will cause the device misinterpreting it as water effect. The corresponding touch will not be reported even if the mutual-cap touch detection is valid.

This feature is disabled if the threshold is set to -32768 and it is disabled by default. Enabling this feature is recommend only if the situation described above is observed. One side effect of this feature is that stuck touch may occur if the device powers up with a palm on the panel.

If the Injected Touch Noise Metric is above **Charger Armor: NMI_THRESH**, it sequentially escalates the Noise Effect Level from the Undetectable state to Level 3. When Noise Armor is at Level 3, it cycles through all alternative TX hop frequencies if the Injected Touch Noise Metric is still above the threshold. Parameter **Charger Armor: AFH_HOP_CYCLES_COUNT** specifies the number of times the AFH logic looping through the alternative TX frequencies before entering Excessive Noise level.

When Noise Effect Level is at Level 2 or below, the mutual-cap TX frequency and signal scaling are the same as the base configuration. When Noise Effect Level is at Level 3, the alternative configurations are used and they can be customized for each TX hop frequency selection. If no external noise and no finger touch is detected for a duration longer than **Charger Armor: CA_REVERT_TIME_MS**, the device exits Noise Armor in descending order of the Noise Effect Levels.

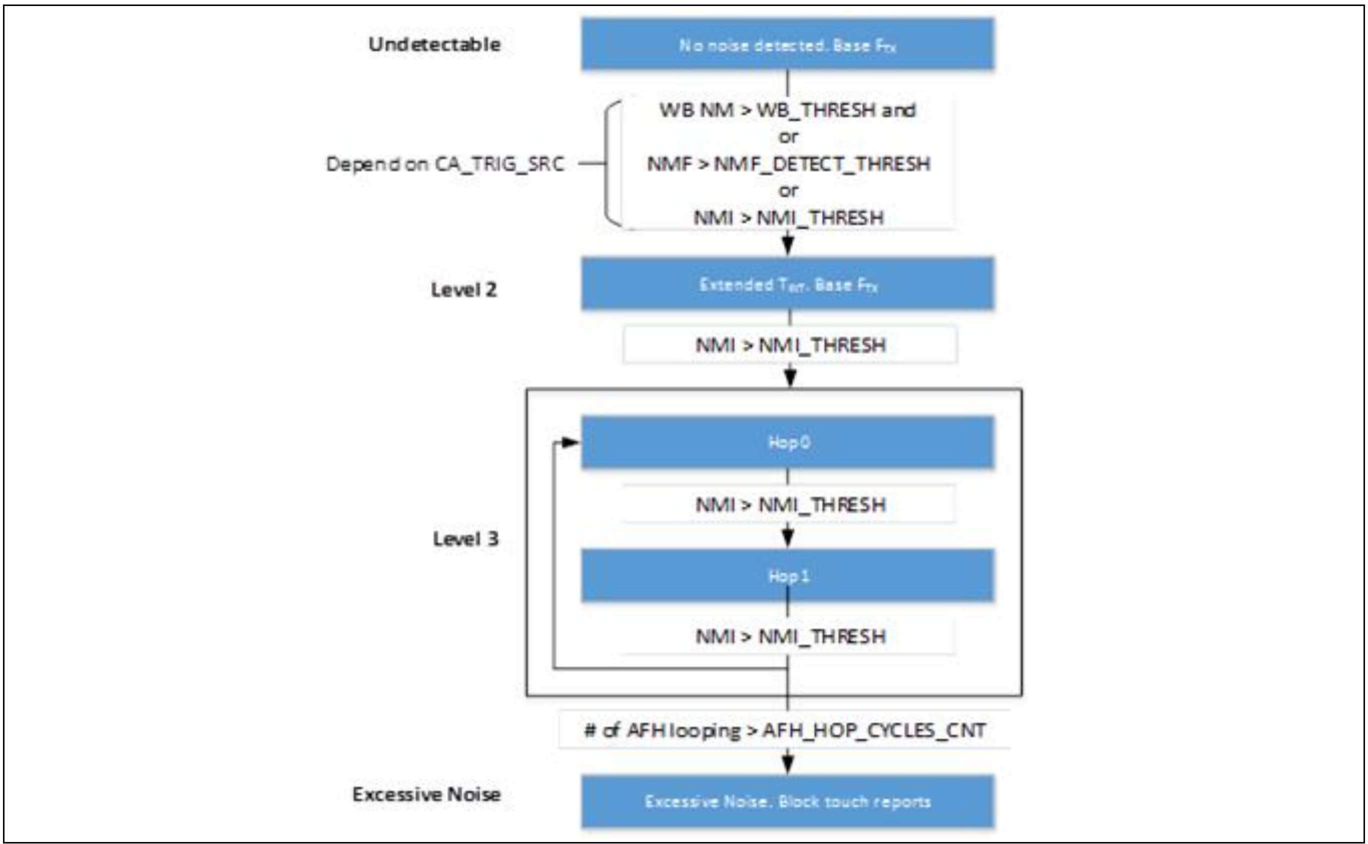


Figure 89 Charger Armor state diagram

8.10.4 Level 2 - 3

Level 2 uses two main groups of parameters to improve noise mitigation. The first group affects the digital data. The second group affects the analog setup.

The digital parameters affect the Raw-count filters and finger thresholds. A full set of Noise Armor-specific raw-count digital filters is available for mutual-cap and CAPSENSE™ buttons. A full set of Noise Armor-specific fat finger size, mutual-cap finger threshold and multi-touch debounce parameters are also available.

The analog parameters allow an extended integration time and the RX line filter to be used (Table 50).

Table 50 Noise Armor level 2 configurable parameters

Item	Noise Armor Parameter	Description	Selection
RX Line Filter	**Scan Filtering: RXLINE_FILT_ENABLE**	RX Line Filter enable.	Disabled / Controlled by CA / Always Enabled (default = Disabled)
	Scan Filtering: RXLINE_FILT_DEBOUNCE	The debounce applied by the RX line filter to any local maxima that passes the threshold requirement.	0 - 255 (default = 2)
	Scan Filtering: RXLINE_FILT_THRESH	Threshold (percentage of maximum proximal 3x3 surrounding signal) at which the Noise Armor RX line filter rejects a local maximum.	0 - 100 (default = 88)

Item	Noise Armor Parameter	Description	Selection
TX Pulses	**Charger Armor: CA_MC_BASE_TX_PULSES_NUM**	Number of TX pulses per conversion for extended Mutual-cap integration time used in Noise Effect Level 2 and higher.	1 - 255 (default = 64)

When the Noise Effect Level is "Undetectable", the number of mutual-cap TX Pulses is specified by **TSS: TX_PULSES_MC**. When in Level 2 or higher, it is specified by parameter **Charger Armor: CA_MC_BASE_TX_PULSES_NUM** and the parameter **TSS: TX_PULSES_MC** is ignored.

The RX Line Filter is a feature that can improve false touch rejection. The parameter **Scan Filtering: RXLINE_FILT_ENABLE** controls whether it can be always enabled, or always disabled, or controlled by CA. In the last case, the RX Line Filter is disabled when the Noise Armor noise level is below 2, and it is enabled when the Noise Armor noise level is at 2 or above.

The RX Line Filter is used to reject false signal manifesting along the same RX lines as a real touch. In normal operation, noise is detected on all RX electrodes. In extreme cases, the noise can cause random false touches along the affected RX electrodes. The RX Line Filter is designed to detect false signal due to noise couple by the touch.

The RX Line Filter works by processing each local maxima (potential touch). The filter calculates the Z8-sum (see 7.1 for definition) for all local maxima within +/-3 RX lines. The current maxima is removed if it is less than a percentage of the maximum Z8-sum. The percentage is defined in **Scan Filtering: RXLINE_FILT_THRESH**. In the example in Figure 90, the finger touch on the right is rejected.

The RX line filter further debounces any local maxima that passes the threshold requirement. The number of scan frames to debounce is defined by **Scan Filtering: RXLINE_FILT_DEBOUNCE**.

If a wide range of finger sizes is expected under the Level 2 noise, reduce **Scan Filtering: RXLINE_FILT_THRESH** until both the minimum-sized finger and maximum-sized finger touching the same RX electrodes are recognized.

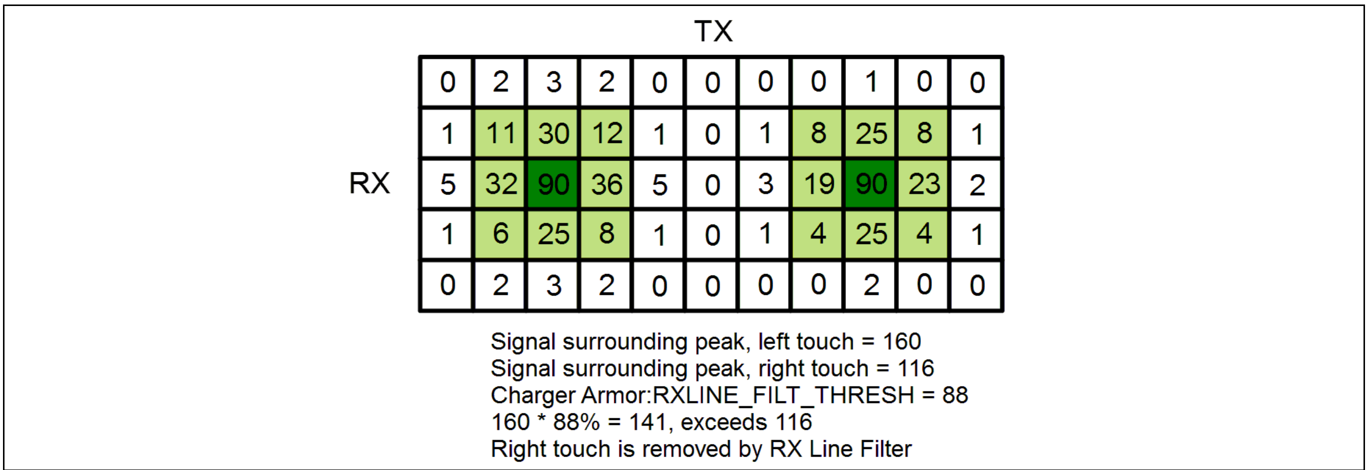


Figure 90 RX line filter example

Proceed to Level 3 tuning if Level 2 tuning cannot effectively mitigate the external noise.

If Noise Armor is unable to solve the noise by increased integration time and additional filtering, Adaptive Frequency Hopping may be an efficient way to solve the problem by applying different scanning configurations to avoid the noise. The objective of this tuning is to find a set of the TX frequencies that have non-overlapping harmonics and adjust the options to ensure proper switching between these configurations. The hopping frequencies should not overlap with the display noise frequencies or their harmonics. The TX frequency depends on the HOPx period parameter according to Equation 50, where HOPx can be HOP0 or HOP1

$$\text{TX Frequency}_{\text{HOPx}} = \frac{48\text{ MHz}}{2 * \text{Charger Armor: CA\_HOPx\_TX\_PERIOD\_MC}}$$

Equation 50

Each hop frequency has the individual number of TX pulses per conversion.

Table 51 Charger armor parameters

Configurable Parameter	Description	Selection
Charger Armor: CA_TOUCH_REVERT_TIME_MS	Delay in ms after charger noise is removed with stable signal on the panel, before going back to normal scanning (no charger noise detected). Rarely changed from default value.	50 - 4294967295 (default = 1000)
Charger Armor: CA_MAX_XY_MOVEMENT_REVERT	Maximum movement allowed for a touch that is on the panel to force revert when noise is not detected in CA. Rarely changed from default value.	0 - 32767 (default = 0)
Charger Armor: CA_HOST_CTRL: Enable	Enable/Disable HOST Controlled Charger Armor. Initializes the CHARGER_STATUS RAM value at device power-up. Rarely changed from default value.	Enabled/Disable (default = Disable)
Charger Armor: CA_HOST_CTRL: Default Value	Initializes the CHARGER_STATUS RAM value at device power-up. If charger noise is expected to be present after device power-up, set this bit to Enable. Otherwise, leave it set to Disable. Rarely changed from default value.	Enabled/Disable (default = Disable)
Charger Armor: WB_REVERT_THRESH	When WB noise is larger than this threshold, it will not revert from charger mode. Rarely changed from default value.	0 – 255 (default = 15)
Charger Armor: CA_DYN_CAL_SAFE_RAW_RANGE	Safe range of Baseline HOP Compensation Data (in absolute Raw-counts). Rarely changed from default value.	0 – 65535 (default = 50)
Charger Armor: CA_DYN_CAL_NUM_SENSOR_THLD_PERCENT	Dynamic Recalibration will be triggered when percentage (scaled to [0-127]) of properly compensated crossings is lower than this value. Rarely changed from default value.	0 – 255 (default 96)
Charger Armor: CA_MC_RAW_FILTER_MASK: CMF Filter	Enable/Disable CMF Filter. Rarely changed from default value.	Disabled/Enabled

Configurable Parameter	Description	Selection
		(default = Enabled)
Charger Armor: CA_MC_RAW_FILTER_MASK: IIR Filter	Enable/Disable IIR Filter. Rarely changed from default value.	Disabled/Enabled (default = Enabled)
Charger Armor: CA_MC_RAW_IIR_COEF	IIR Coefficient for Mutual-cap scan when Charger Armor is active. Rarely changed from default value.	1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Charger Armor: CA_MC_RAW_IIR_THRESH	IIR Threshold Base for Mutual-cap scan when Charger Armor is active. Rarely changed from default value.	0 – 32767 (default = 100)
Charger Armor: CA_MC_RAW_CMF_THRESH	CMF Threshold Base for Mutual-cap scan when Charger Armor is active. Rarely changed from default value.	0 – 32767 (default = 200)
Charger Armor: CA_BTN_MC_RAW_FILTER_MASK: CMF Filter	Enable/Disable CMF Filter. Rarely changed from default value.	Disabled/Enabled (default = Enabled)
Charger Armor: CA_BTN_MC_RAW_FILTER_MASK: IIR Filter	Enable/Disable IIR Filter. Rarely changed from default value.	Disabled/Enabled (default = Enabled)
Charger Armor: CA_BTN_MC_RAW_IIR_COEFF_BUT CYAT6165X/8165X (TSG6L) **Charger Armor: CA_BTN_MC_RAW_IIR_COEFF_BUT** 8168X (TSG6XL)	IIR Coefficient for Mutual-cap Button scan when Charger Armor is active. Rarely changed from default value.	1/2, 1/4, 1/8, 1/16, 1/32 (default = One Half)
Charger Armor: CA_BTN_MC_RAW_IIR_THRESH	IIR Threshold Base for Mutual-cap Button scan when Charger Armor is active. Rarely changed from default value.	0 – 32767 (default = 100)
Charger Armor: CA_BTN_MC_RAW_CMF_THRESH	CMF Threshold Base for Mutual-cap Button scan when Charger Armor is active. Rarely changed from default value.	0 – 32767 (default = 200)
Charger Armor: NM_BURST_OVFW_THRESH	The number of consecutive mutual-cap samples allowed to overflow before raw data processing is blocked. Prevents bad data from burst noise. Rarely changed from default value.	0 – 65535 (default = 65535)

Configurable Parameter	Description	Selection
Charger Armor: OVERFLOW_CNT_BUTTON_THRESHOLD	The number of consecutive button samples allowed to overflow before raw data processing is blocked. Prevents bad data from burst noise. Rarely changed from default value.	0 – 65535 (default = 65535)
Charger Armor: CA_HOP0_TX_PERIOD_MC	First AFH alternative frequency Mutual-cap scan TX period configuration. Rarely changed from default value.	6165X/8165X (TSG6L) 48 – 1000 8168X (TSG6XL) 68 – 1000 (default = 246)
Charger Armor: CA_HOP1_TX_PERIOD_MC	Second AFH alternative frequency Mutual-cap scan TX period configuration. Rarely changed from default value.	6165X/8165X (TSG6L) 48 – 1000 8168X (TSG6XL) 68 – 1000 (default = 277)
Charger Armor: CA_HOP0_TX_PULSES_MC	First AFH alternative frequency Mutual-cap scan number of TX Pulses per conversion. Rarely changed from default value.	0 – 255 (default = 64)
Charger Armor: CA_HOP1_TX_PULSES_MC	Second AFH alternative frequency Mutual-cap scan number of TX Pulses per conversion. Rarely changed from default value.	0 – 255 (default = 64)
Charger Armor: CA_FINGER_THRESH_MUT_HI	Mutual-cap finger touchdown threshold applied in the presence of charger noise (level 1 or higher). Rarely changed from default value.	0 – 32767 (default = 300)
Charger Armor: CA_FINGER_THRESH_MUT_LO	Mutual-cap finger liftoff threshold applied in the presence of charger noise (level 1 or higher). Rarely changed from default value.	0 – 32767 (default = 300)
Charger Armor: CA_FINGER_MT_DEBOUNCE	Number of consecutive refresh cycles for which a touch must be detected prior to being reported when Charger Armor is not active. Applies to 2nd and successive touches but not to 1st touch. Rarely changed from default value.	0 – 255 (default = 2)

Configurable Parameter	Description	Selection
Charger Armor: CA_FINGER_FT_DEBOUNCE	Number of consecutive refresh cycles for which a touch must be detected prior to being reported when Charger Armor is active. Applies to the first touch. Rarely changed from default value.	0 – 255 (default = 0)
Charger Armor: CA_FINGER_Z9_FILT_SCALE	Z9 filter scale for normal finger when Charger Armor is active. Peak with Z9 < (scale*CA_FINGER_THRESHOLD_MUTUAL) will be removed. Rarely changed from default value.	0 – 4 (default = 2)
Charger Armor: CA_MIN_FAT_FINGER_SIZE	Minimum number of contiguous activated panel intersections that define a fat finger when Charger Armor is active. It is used when a fat finger was not detected yet. Rarely changed from default value.	0 – 255 (default = 15)
Charger Armor: CA_MAX_FAT_FINGER_SIZE	Maximum number of contiguous activated panel intersections that define a fat finger when Charger Armor is active. It is used when a fat finger was not detected yet. Rarely changed from default value.	0 – 255 (default = 40)
Scan Filtering: RXLINE_FILT_ENABLE	Enables the RX Line Filter.	Disabled / Controlled by CA / Enabled (default = Disabled)
Scan Filtering: RXLINE_FILT_DEBOUNCE	The debounce the RX line filter applies when a local maxima passes the RX line filter threshold.	0 – 255 (default = 2)
Scan Filtering: RXLINE_FILT_THRESH	Percentage of the maximum Z8-sum threshold (from all maxima within +/-3 RX lines) that the RX line filter rejects a local maximum (touch).	0 – 100 (default = 88)

8.11 Water rejection

Water Rejection is controlled by the **Scan Filtering: WATER_REJ_ENABLE** parameter. The algorithm based on analyzing the data from both mutual-cap and self-cap scanning. This process is mostly automatic, the only parameter that needs tuning is **Scan Filtering: WATER_REJ_SNS_WIDTH**.

Note that enabling **TSS: SHIELD_EN_SC** significantly improves water rejection performance, but it increases electromagnetic emission.

8.11.1 Locked sensor width

The issue is that water rejection algorithm rejects all mutual-cap local maximums disregarding the signal level if there is no self-cap signal on the exact same row/column. Because the self-cap and mutual-cap scans are not done at the same time, it is possible that the local maximum of a fast-moving finger will be detected on different sensors for the mutual-cap and self-cap scans. This scenario causes the water rejection algorithm to reject the local maxima, and therefore discard the finger as a real touch, which results in a liftoff report. This is illustrated in the following figures:

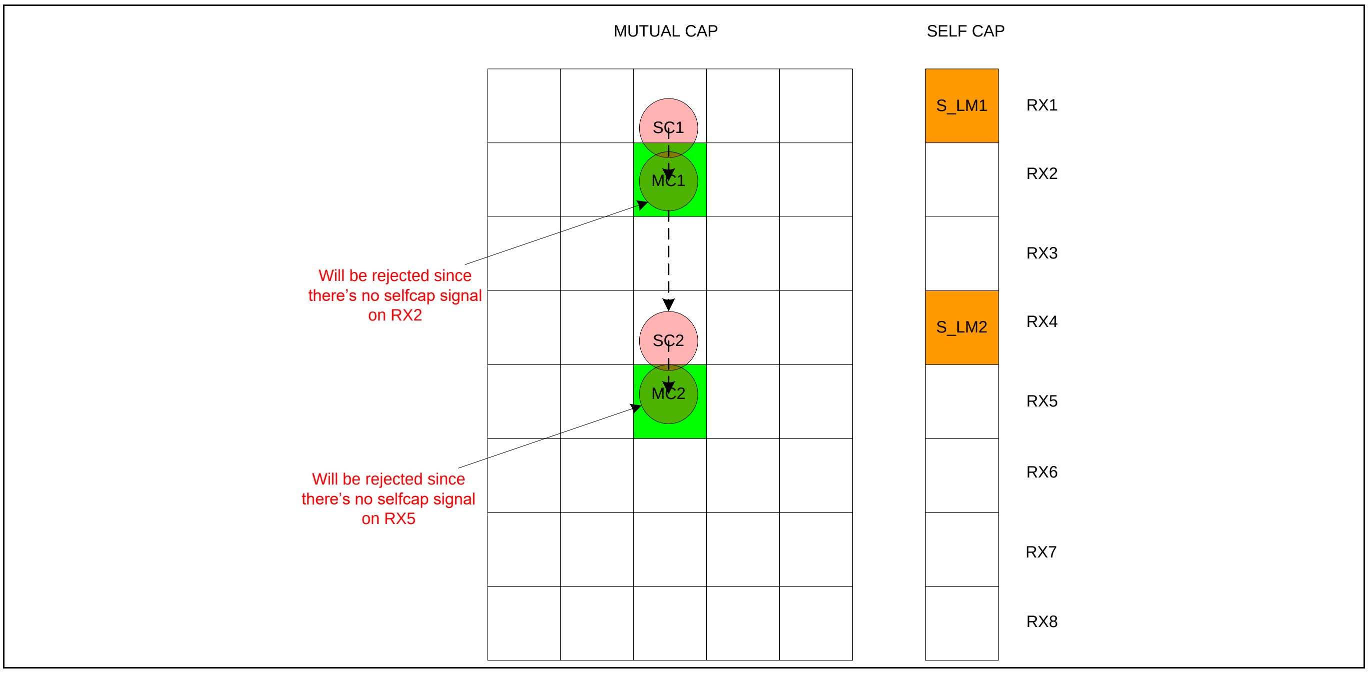


Figure 91 Fast moving finger needs to lock adjacent sensors

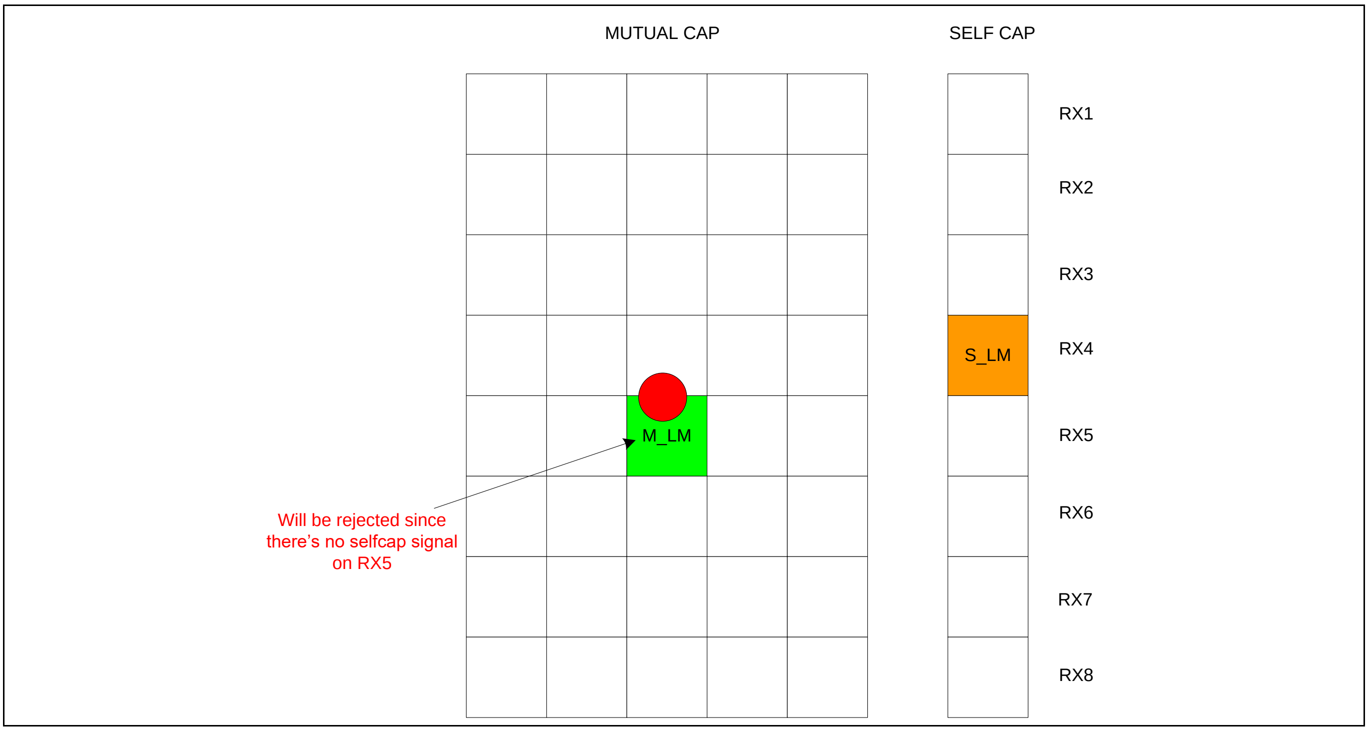


Figure 92 Small finger needs to lock adjacent sensors

The water rejection sensor width feature will extend the area for self-cap signal look-up for the configured width. If the width is set to 0, this feature will be disabled and the finger detection algorithm only look at the same self-cap row/column as the mutual-cap local maxima. If the width is set to non-zero, then the finger detection algorithm will check not only row X where mutual-cap local maximum was detected but rows (X-SNS_WIDTH) to (X+SNS_WIDTH). This is shown in the following figure:

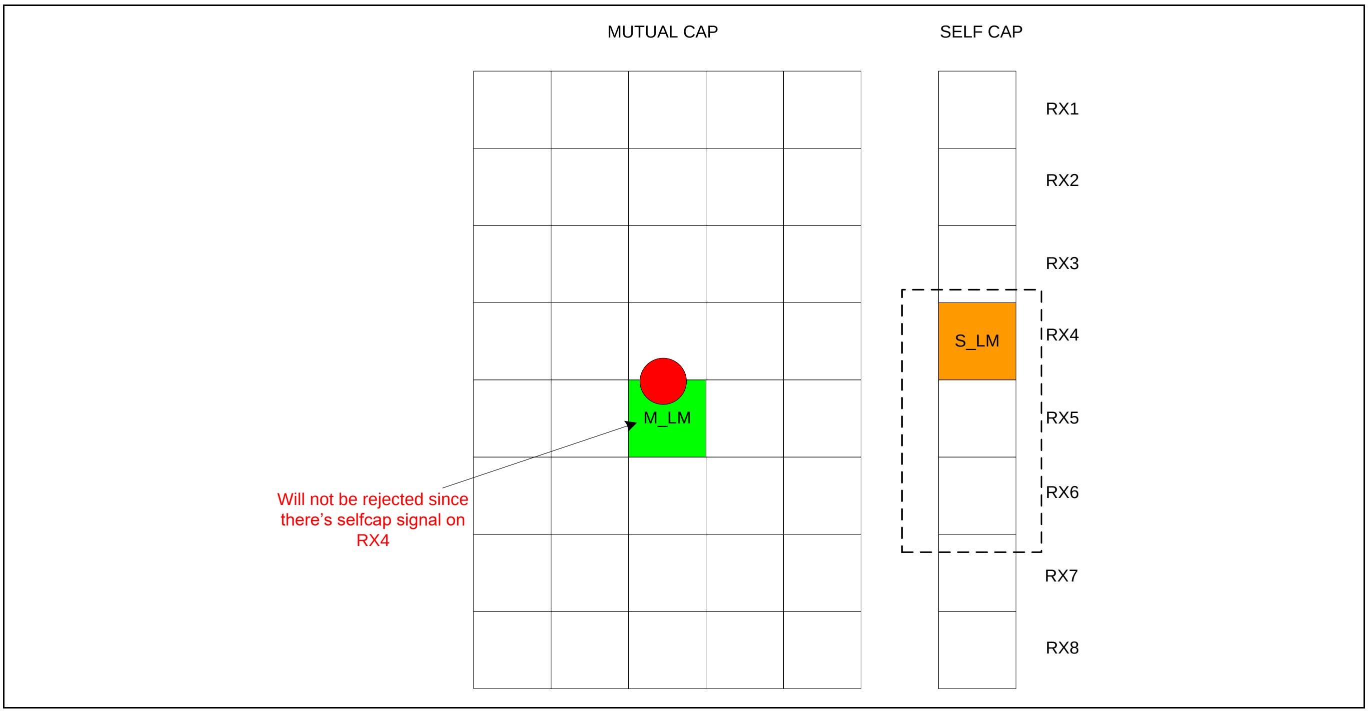


Figure 93 Water rejection sensor width 1

It is recommended to leave the **Scan Filtering: WATER_REJ_SNS_WIDTH** to its default of 2.

8.11.2 Parameters

Table 52 Wet finger detection parameters

Configurable Parameter	Description	Selection
Scan Filtering: WATER_REJ_ENABLE	Enables or disables the advanced water rejection feature; when enabled, self-cap scanning is performed on every panel scan.	Enabled / Disabled 6165X/8165X (TSG6L) (default = Disabled) 8168X (TSG6XL) (default = Enabled)
Scan Filtering: WATER_REJ_SNS_WIDTH	Sensor width for Water Rejection (Maximum value is 7). Rarely changed from default value.	0 – 7 6165X/8165X (TSG6L) (default = 0) 8168X (TSG6XL) (default = 2)

8.12 Wet finger tracking

8.12.1 Background

The wet finger algorithm continuously evaluates and compares the raw-count and diff-count data to determine if additional measure is required to filter out the water effects. These measures include dynamic finger threshold, disabling large object and multi-finger detection, signal peak filtering, alternative multi-touch debounce, and limiting to single-touch reporting. There are 3 levels of water effects (level 1, 2 and 3) and different measures are applied at different water effect levels. The goal of wet finger tracking is to identify the presence of water on panel and to determine what level of water effect mitigation measures to be applied to enhance tracking any finger detected thereafter.

8.12.1.1 Mutual-cap raw-count and self-cap diff-count signal for water and finger touches

Water rejection and look-for-touch features should be enabled for wet finger tracking algorithm, because both mutual-cap and self-cap data are used to evaluate for touch signal. When self-cap shield is enabled (**TSS: SHIELD_EN_SC** = Enabled) water has a minimal impact on self-cap counts, but a significant impact on mutual-cap counts. This difference in reaction to water is the basis for water rejection. An example is shown in Figure 94.

When self-cap shield is disabled (**TSS: SHIELD_EN_SC = Disabled**), water affects the counts and it is very difficult to find proper tuning for wet finger algorithm.

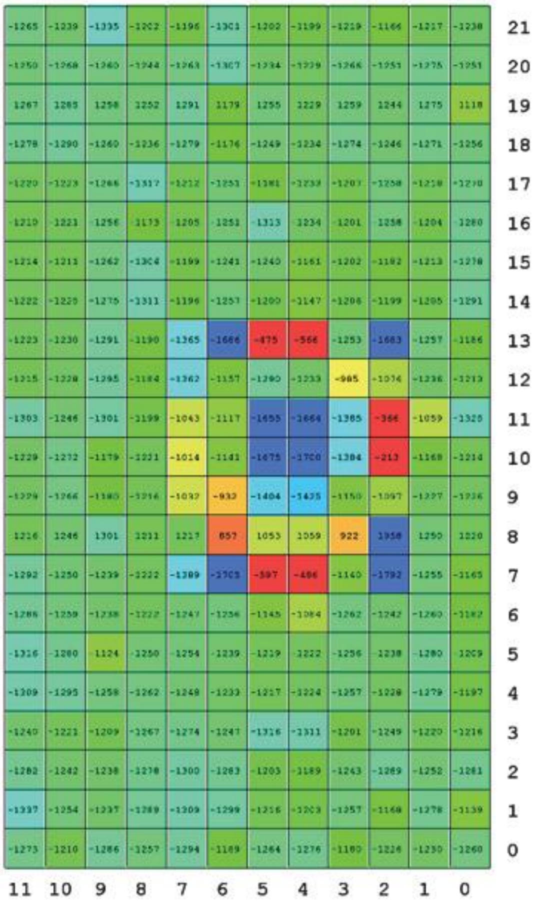


Figure 94 Mutual-cap raw-count heat map with water on panel

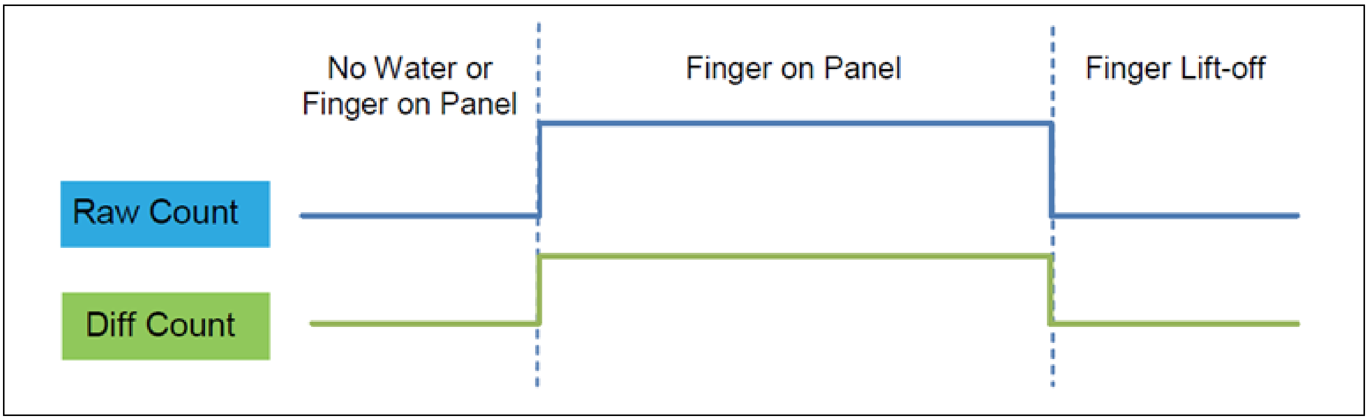


Figure 95 Self-cap diff-count and mutual-cap raw-count changes without water on panel

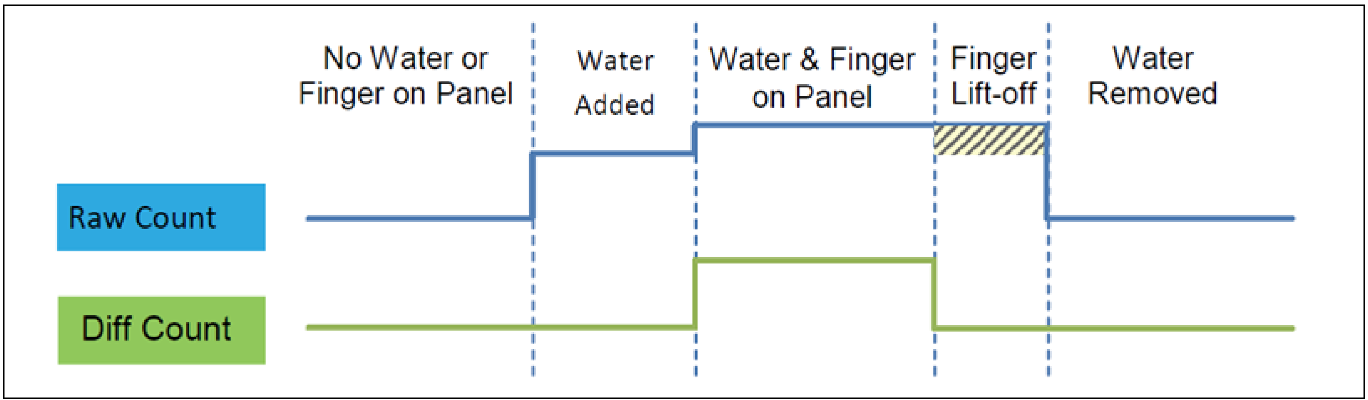


Figure 96 Self-cap diff-count and mutual-cap raw-count changes with water on panel

Figure 95 and Figure 96 illustrate the mutual-cap raw-count and self-cap diff-count signal count in the cases of dry finger tracking, water on panel, and wet finger tracking. With water on panel, mutual-cap raw-count is expected to increase while the change in self-cap diff-count is minimal. Both mutual-cap raw-count and self-cap diff-count are expected to increase in the case of dry finger touch and wet finger touch. Therefore, one can identify the presence of water by comparing the relative signal count between self-cap diff-count and mutual-cap raw-count.

8.12.1.2 Raw-count and diff-count signal variation for water and finger touches

Detecting changes in the raw-count and diff signal count before and after finger touch or the presence of water require storing multiple copies of the panel data which uses a lot of RAM memory. An alternative method is to evaluate the total signal count variation across the panel as illustrated in Figure 97. The total signal count variation is the sum of the absolute difference between the neighboring mutual-cap sensors across the RX.

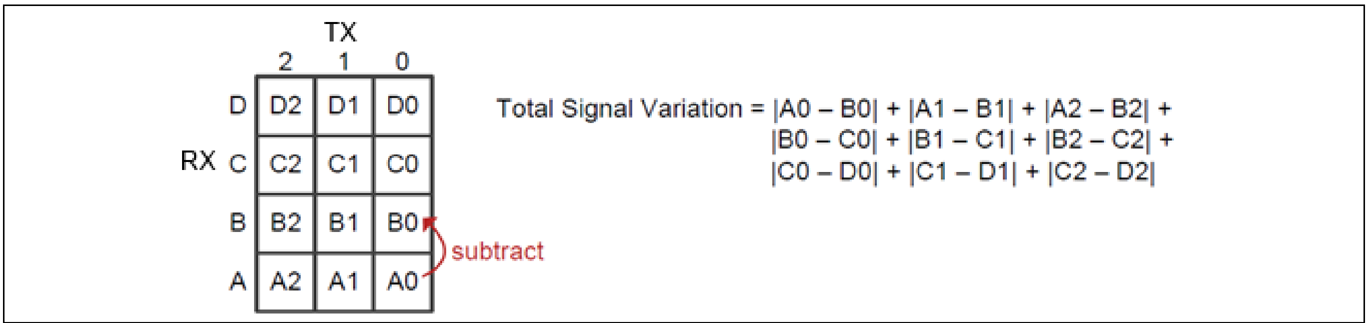


Figure 97 Raw-count and diff-count signal variation illustration

To minimize the variation effect due to noise, the absolute difference between the neighboring sensors must be higher than a threshold value to be added to the total signal count variation. If the difference is below this threshold, the value is ignored and is not added to the total signal count variation.

Test results in Figure 98 and Figure 99 show that tracking the total signal count variation allows the device to identify the water on panel and a finger touch on the panel. Both mutual-cap raw-count and self-cap diff-count total signal variation increase with a touch on the panel. Mutual-cap raw-count total signal variation significantly increases if only water is on the panel.

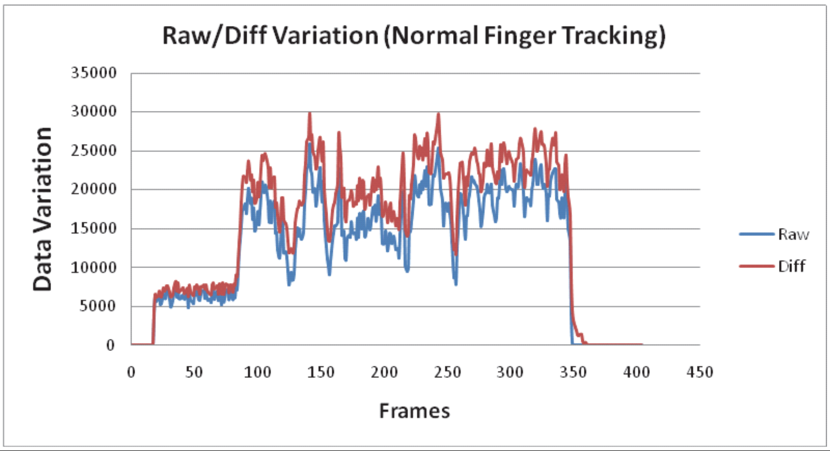


Figure 98 Mutual-cap raw-count and self-cap diff-count signal variation normal finger tracking test

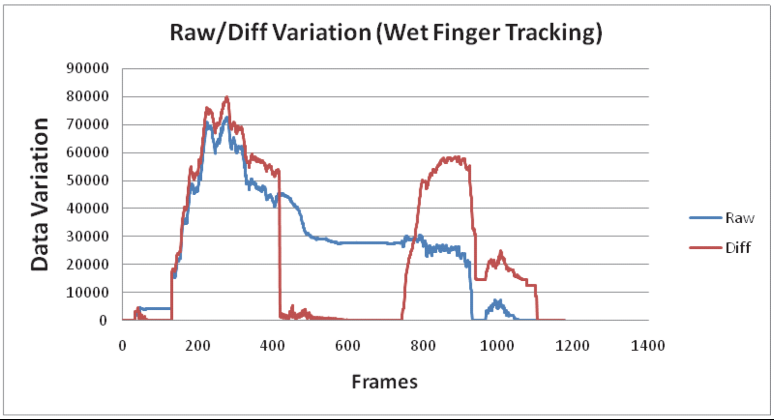


Figure 99 Mutual-cap raw-count and self-cap diff-count signal variation wet finger tracking test

8.12.2 Wet finger detection

To trigger the wet finger tracking, one must enter the right state in the state machine and the detected total signal variation must exceed the specified threshold to enter the proper water effect level.

8.12.2.1 State machine

The state machine is designed to differentiate between the following 4 states:

- No water or finger on panel.
- Normal Finger on Panel (no water).
- Water on panel (no finger).
- Both water and finger on panel.

To ensure the change in state is correct, entering and exiting wet finger tracking is debounced by the parameters **Wet Finger: WF_ENTER_DEBOUNCE** and **Wet Finger: WF_EXIT_DEBOUNCE**. If no activity is detected for longer than define by **Wet Finger: WF_WATER_STATE_TIMEOUT**, the wet finger state is exited.

Raw-count and diff-count total signal variations are used to determine the state transitions. The state machine consists of 4 states, one state for each of the cases above. It is designed to ensure that the wet finger tracking enhancement measures are only applied when water is detected on the panel. The state transitions are illustrated in Figure 100. “Raw = 1” means that the raw-count total signal variation exceeds the threshold **Wet Finger: WF_RAW_VAR_THRESH**. “Diff = 1” means that the diff-count total signal variation exceeds the threshold **Wet Finger: WF_DIFF_VAR_THRESH**.

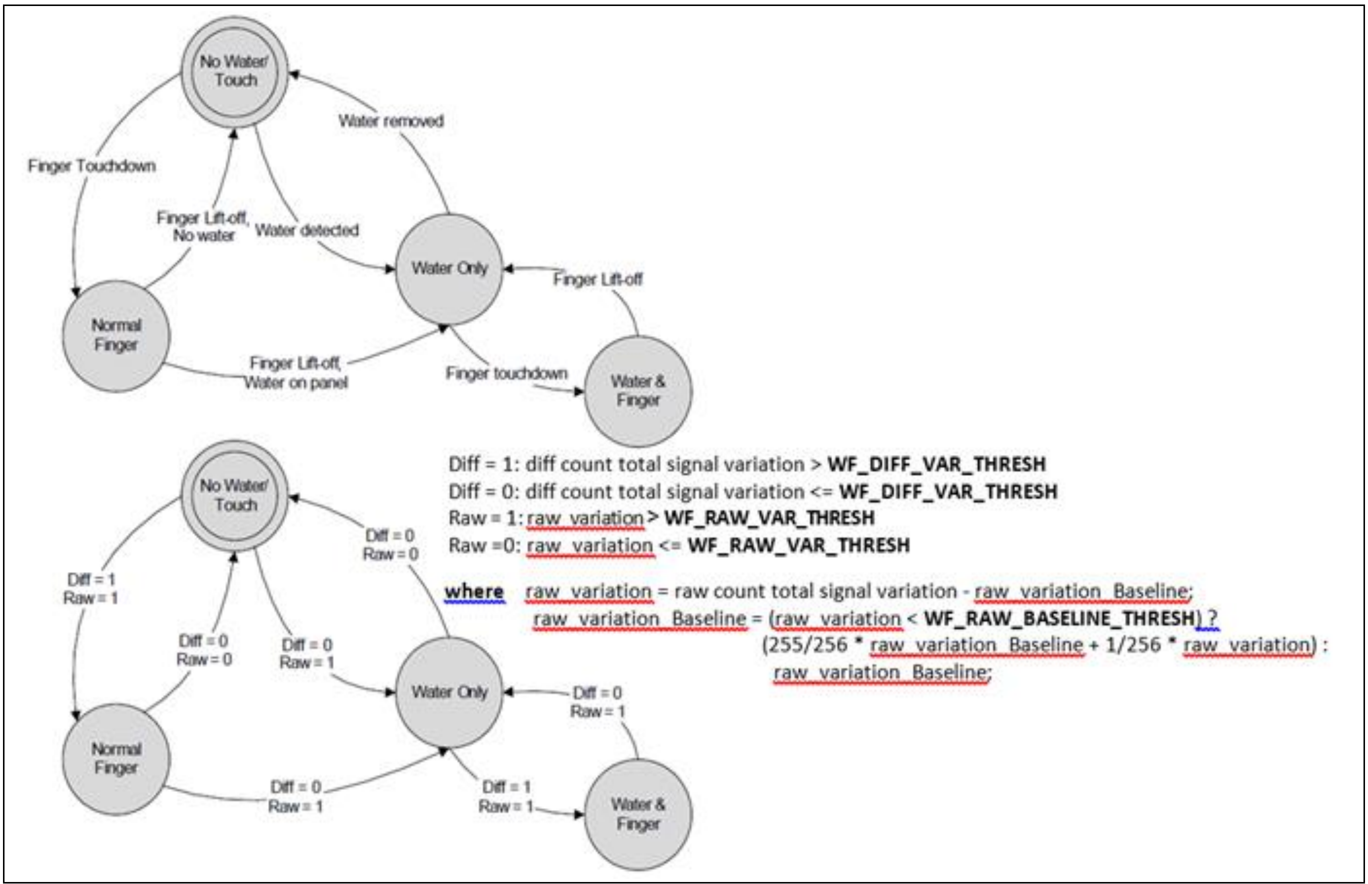


Figure 100 Wet finger tracking state machine

Debouncing is applied for entering the “Water Only” state. The condition must be detected in 10 consecutive refresh cycles before the state machine enters the “Water Only” state. This is done to prevent false triggering due to noise or signal residue. Debouncing is not applied for exiting the “Water Only” state. Different wet finger enhancement measures are applied when the state machine is in either “Water Only” or “Water & Finger” state.

Wet Finger Tracking State Machine effects also water state variable:

- Water state is 1 (no water detected) in “No Water/Touch” or “Normal Finger” state.
- Water state is 2 (water detected) in “Water Only” or “Water & Finger” state.

8.12.2.2 Water effect levels and wet finger tracking enhancement measures

There are four water effect levels. Device will evaluate the water effect level only when the state machine is in either “Water Only” or “Water & Finger” state. Different wet finger tracking enhancement measures are applied at each level (see Table 53).

If the device is in “No Water/Touch” or “Normal Finger” state, the water effect level is 0 (no water level) and no wet finger tracking enhancement measure will be applied.

The device is in water effect level 1 (low level of water) if the detected raw-count total signal variation is below the threshold **Wet Finger: WF_LEVEL_THRESH**. It enters level 2 (medium level of water) if it is above **Wet Finger: WF_LEVEL_THRESH** and below four times of **Wet Finger: WF_LEVEL_THRESH**. It enters level 3 (high level of water) if it is equal to or above 4 times of **Wet Finger: WF_LEVEL_THRESH**.

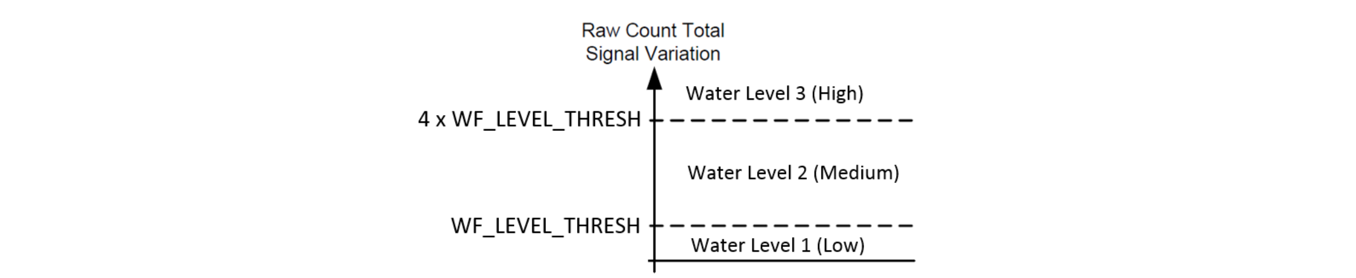


Figure 101 Water effect level threshold illustration

So, device can be in following water effect levels:

- Level 1 (water state is 2: water detected): Very little water exists on the panel. The water is usually very thin film such as condensation or very small drops of water sprayed to the panel. Multi-finger tracking may work at this time.
- Level 2 (water state is 2: water detected): Small Droplets (0.1 mm to 3 mm) exists on panel. Priority is reliable tracking of single finger.
- Level 3 (water state is 2: water detected): Large Droplets (3 mm to 18 mm) or thick water film exists on panel. No false touch is the highest priority.

Table 53 Water effect levels and wet finger tracking enhancement measures

	Wet Finger Tracking Enhancement Measures	Summary Description (detailed description below)	Water Level 1	Water Level 2	Water Level 3
1	Dynamic Finger Threshold	The mutual-cap finger threshold is dynamically modulated based on the peak signal value.	Yes	Yes	Yes
2	Alternative Multi-Touch Debounce	The alternative multi-touch debounce value (**Wet Finger: WF_MT_DEBOUNCE**) is used.	Yes	Yes	Yes
3	Signal Peak Filtering Algorithm 1 (negative signal filtering)	Filter evaluates surrounding negative signal count to remove the artificial signal peak due to water coupling effect.	Yes	Yes	Yes
4	Signal Peak Filtering Algorithm 2 (Z8 filtering)	Filter compares the peak signal and the surrounding signal sum to remove artificial signal peak due to water coupling effect.	-	Yes	Yes

	Wet Finger Tracking Enhancement Measures	Summary Description (detailed description below)	Water Level 1	Water Level 2	Water Level 3
5	Signal Peak Filtering Algorithm 3 (proximity filtering)	Filter removes the false signal peaks surrounding a finger touch which has the highest peak signal count.	-	Yes	Yes
6	Disabling Large Object and Multi-Finger Detection	Large object detection is disabled and the touch zone multi-touch evaluation is also disabled to minimize false touches and touch reporting consistency.	-	Yes	Yes

1. Dynamic finger threshold:

The mutual-cap finger threshold is dynamically modulated based on the peak signal value. The mutual-cap finger threshold is specified by the following equation. It is the maximum of the mutual-cap finger threshold for normal touch detection and the max peak signal diff-count scaled by the parameter **Wet Finger: WF_THRESH_MUT_COEF**. The higher finger threshold will minimize false touches due to water effects.

$$\text{Threshold} = \text{Max}(\text{Normal Finger Threshold}, \text{Peak Threshold})$$

Equation 51

where

$$\text{Normal Finger Threshold} = \text{Fingers: FINGER\_THRESH\_MUT\_HI (or LO)}$$

Equation 52

$$\text{Peak Threshold} = \frac{\text{Max Peak Signal} * \text{Wet Finger: WF\_THRESH\_MUT\_COEF}}{128}$$

Equation 53

2. Alternative multi-touch debounce:

The normal finger touch multi-touch debouncing (**Fingers: FINGER_MT_DEBOUNCE** or **Gloves: GLOVES_MT_DEBOUNCE**) is ignored and **Wet Finger: WF_MT_DEBOUNCE** is used instead.

3. Signal peak filtering algorithm 1 (negative signal filtering):

Water coupling tends to randomize the diff-count between the adjacent sensors. One possible scenario is that it sends some of the sensors to negative diff-count while others to a higher positive diff-count as shown in Figure 102. This peak filter is designed to remove signal peaks that are artificially created due to this water coupling effect. The absolute sum of the negative diff-counts of the eight sensors surrounding the peak (Z8_Neg_Abs_Sum) is calculated (positive diff-counts are excluded). The signal peak is removed and will not be reported as finger touch if Z8_Neg_Abs_Sum is greater than half of the total diff-count sum around the peak (Z8_Sum). This peak filter is applied for all water effect levels.

	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
10	272	-75	-41	159	297	12	144	146	10	-2232	-4106	4155	4792	2909	-6475	-1	1233	285	-1	5	-230
9	228	13	9	172	376	23	-1	-911	-1507	1675	4251	-363	7	2019	4200	-6109	235	1081	0	0	23
8	84	24	-1	39	-1	-668	-2063	-286	5019	3651	5019	-2882	-3534	-853	3297	3276	-3826	-8	329	30	9
7	65	22	27	-59	-2091	-1141	2428	4585	1426	466	3334	-3114	-2648	-942	1581	3249	1825	-2910	343	158	81
6	22	28	0	-431	178	2646	1323	-9	-550	1727	-1538	-1535	-66	0	1121	1836	889	347	25	637	55
5	11	50	118	1830	2385	-412	-777	-867	-391	794	-3171	1364	1480	701	790	979	-1	-34	1020	1257	90
4	46	68	176	620	2963	17	-901	-1127	-541	-5162	86	3405	2439	916	748	1195	107	392	743	-15	84
3	21	68	113	-431	-1655	2782	1332	-863	-984	0	-1270	1701	1336	63	504	1602	911	1822	186	-2730	79
2	13	30	9	57	-1175	-2231	614	2193	1888	-1030	-58	778	921	1922	3262	-1897	-1507	1733	-2411	1155	113
1	-138	29	1	96	21	0	60	8	-125	596	2292	-1	-93	-1237	-3710	1254	2763	39	590	652	64
0	41	-35	9	2	39	11	69	199	220	18	695	552	69	-969	424	804	381	257	136	206	8

Figure 102 Diff-count example due to water coupling randomization

Signal Peak is removed if:

$$\text{Z8\_Neg\_Abs\_Sum} > \frac{\text{Z8}_{\text{Sum}}}{2}$$

Equation 54

<u>4. Signal peak filtering algorithm 2 (Z8 filtering):</u>

This filter is applied when water effect level is at 1 or above. Another possible scenario caused by the water coupling randomization is that it could artificially increase the diff-count on a small number of sensors along the water traces while leaving the surrounding sensors at relatively low signal count. This could potentially cause false touch. This filter is designed to remove the artificial peaks created by this effect. It compares the peak diff-count against the diff-count sum of the 8 sensors surrounding the peak (Z8_Sum). The signal peak is removed and is not reported as finger touch if Z8_Sum is less than 1/16 of peak diff-count multiplied by **Wet Finger: WET_FINGER_Z8_MULT** (by default twice the peak diff-count because default value of **Wet Finger: WET_FINGER_Z8_MULT** is 32).

Signal Peak is removed if:

$$\text{Z8}_{\text{Sum}} < \frac{\text{Wet Finger: WET\_FINGER\_Z8\_MULT} \cdot \text{Local Peak Signal}}{16}$$

Equation 55

The drawback of this filter is that it could potentially affect the touch reporting for finger size smaller than a sensor pitch in the dry area of the panel. It could create a broken finger track as the small finger is travelling across the dry area of the panel.

5. Signal peak filtering algorithm 3 (proximity filtering):

This filter is applied when the water effect level is at 2 (medium level of water) or above. A third possible scenario caused by the water coupling randomization is that it could create multiple signal peaks around a real finger touch. This filter is specifically designed to remove these artificial peaks around the real finger. The filter searches for the peak with the maximum signal count (Max_Peak_Signal) and creates a rejection boundary box 3 sensors away from this peak. Any other signal peak detected within this 7x7 boundary box is removed and is not reported.

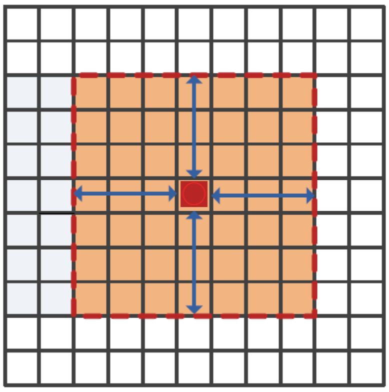


Figure 103 Peak filtering rejection boundary box

The drawback of this filter is that it could potentially remove valid touches when multiple touches are located nearby in the dry area of the panel. Because this filter only applies to the finger touch with the maximum signal count, this filter will not apply to all touches if multiple fingers are on the panel.

6. Signal peak filtering algorithm 4 (large object filtering):

A large water puddle on the panel could affect the diff-count over many sensors. The touch controller could misinterpret the affected area as a large object, falsely removing the finger touchdown in the area. When the device is in water level 1 (low level of water), 2 (medium level of water) or 3 (high level of water) the normal max fat finger size specified in **Fingers: MAX_FAT_FINGER_SIZE** is ignored and it is set to 255. This reduces the chance of false trigger of large object detection.

The centroid algorithm searches through each touch zone to determine if one or multiple touches should be reported for the detected touch object if two or more signal peaks are detected. But with the constant changing of the shape of the water trace and the signal count randomization due to water coupling, the algorithm could misinterpret the results and report the wet finger and the water trace as multiple finger touches. Therefore, when the device is in water level 1, 2 or 3 (water detected), the multi-finger feature specified in **Fingers: FINGER_OBJECT_FEATURES: multi finger object** is ignored and is internally disabled.

8.12.3 Tuning procedure

The following are general guidelines for parameter configuration.

To simplify wet finger parameter tuning, the standard TTHE Listener (View->Tool Windows->Listener) can be used. It can display water metrics data of water detection algorithms (mutual-cap raw-count and self-cap diff-count variation signals, water state and level) in real time including them to the TTHE Report instead of the Noise information. To enable this, set **Device Setup: REPORT_CONFIG: Metrics output mode** to “Water”, and **Device Setup: REPORT_CONFIG: Noise Water metrics output enable** to “Enabled”. This enables metrics data monitoring in the Listener.

Note: Setting **Device Setup: REPORT_CONFIG: Noise Water metrics output enable** to "Enabled" will cause Touch Report generation during each scan even no finger on the panel. It is recommended to disable this parameter after wet finger parameter tuning is complete, to avoid touch report generation without touch.

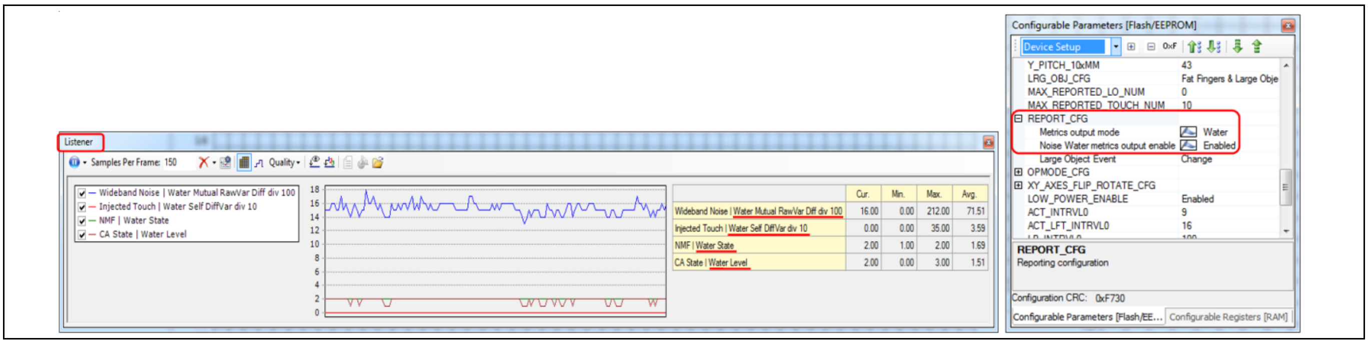


Figure 104 Example water metrics data from Listener

Note: water metrics data (Mutual-cap Raw and Self-cap Diff Variation signals, water state and level) is described in Section 8.12.2.1 and Section 8.12.2.1.

1. Scale the wet finger threshold parameters:

- Set **Wet Finger: WF_RAW_BASELINE_THRESH** to be about 2.7 times the **TSS: SCALE_FACT_MC** value or keep it default.
- Set **Wet Finger: WF_RAW_VAR_THRESH** to be about 3.3 times the **TSS: SCALE_FACT_MC** value or keep it default.
- Set **Wet Finger: WF_DIFF_VAR_THRESH** to be about 1.7 times the **TSS: SCALE_FACT_SC** value or keep it default.
- Set **Wet Finger: WF_LEVEL_THRESH** to be about 5 times the **TSS: SCALE_FACT_MC** value or keep it default.

   Note: To find optimal settings of these parameters it is recommended to monitor the water metrics data in the Listener window (see Figure 104). These parameters usage is explained in Figure 100 and Figure 101.

2. Configure **Wet Finger: WF_RAW_CALC_THRESH** and **Wet Finger: WF_DIFF_CALC_THRESH**. The signal count difference between adjacent mutual-cap sensors must be larger than this threshold to be included in calculating the total diff-count signal variation. Difference below the threshold is excluded from the calculation. Evaluate the signal count differences for both mutual-cap raw-count and self-cap diff-count on a dry panel without finger touch, and set the thresholds to the average value observed.

   Note: For the panel without good raw-count signal uniformity, it is hard to find an optimal **Wet Finger: WF_RAW_CALC_THRESH**, so it is recommended to set **Wet Finger: WF_RAW_CALC_THRESH** about 0.

3. Increase **Wet Finger: WF_MT_DEBOUNCE** if false touch is observed. Decrease parameter if it drops out small finger touch.
4. The algorithm could be falsely triggered by small fingers on a dry panel. It is because the small finger has similar touch signature as water droplets. Increase **Wet Finger: WF_RAW_VAR_THRESH** if observing small finger tracking issue on dry panel. Increase parameter reduces the change of false triggering.
5. There is an additional algorithm that can be helpful for filtering of false peaks with small signals during wet finger tracking. To enable it, **Fingers: PEAK_IGNORE_COEF** should be set to non-zero value. It is recommended to keep it to 10 or less, because a higher value can cause rejection of a small real finger when a large finger is on the panel. It will reject all peaks < ((**Fingers: PEAK_IGNORE_COEF** * Max_Peak_Signal) / 32), where Max_Peak_Signal is the peak with the maximum signal count. It works for water levels 0-3 (with and without water detected).

8.12.4 Parameters

Table 54 Wet finger detection parameters

Configurable Parameter	Description	Selection
Wet Finger: WF_ENABLE	Wet finger tracking feature enable.	Enabled / Disabled (default = Enabled)
Wet Finger: WF_RAW_CALC_THRESH	The threshold for calculating the variation of raw-counts when water drop on panel. Raw-count variation that above this threshold will be included.	0 – 65535 (default = 150)
Wet Finger: WF_RAW_BASELINE_THRESH	The threshold for updating the baseline of raw-count variation. Raw-count variation that under this threshold will be updated.	0 – 65535 (default = 800)
Wet Finger: WF_DIFF_CALC_THRESH	The threshold for calculating the variation of diff-counts when water drop on panel. Diff-count variation that above this threshold will be included.	0 – 65535 (default = 100)
Wet Finger: WF_RAW_VAR_THRESH	The threshold for detection water by calculating the variation of raw-counts when water drop on panel.	0 – 65535 (default = 1000)
Wet Finger: WF_DIFF_VAR_THRESH	The threshold for detection water by calculating the variation of diff-counts when water drop on panel.	0 – 65535 (default = 500)
Wet Finger: WF_LEVEL_THRESH	The threshold for detection high water by calculating the variation of raw-counts when water drop on panel.	0 – 65535 (default = 1500)
Wet Finger: WF_ENTER_DEBOUNCE	This is the debounce to enter water mode from non-water mode. Water mode enters after water is detected for several frames (specified by this value).	0 – 255 (default = 5)
Wet Finger: WF_EXIT_DEBOUNCE	This is the debounce to exit from water mode to non-water mode. Water mode exits after water is detected for several frames (specified by this value).	0 – 255 (default = 10)
Wet Finger: WF_WATER_STATE_TIMEOUT	This is the timeout to exit from water state to the "no water/touch" state. Units are in minutes. Set to 0 to disable this feature. Rarely changed from default value.	0 – 255 (default = 0)
Wet Finger: WF_THRESH_MUT_COEF	Dynamic mutual-cap finger threshold coefficient for wet finger tracking. Mutual-cap finger threshold is specified by the following equation. Recommend leaving parameter at default value. Threshold = Max (**Fingers: FINGER_THRESH_MUTUAL_HI**, Max_Pk_Signal * **Wet Finger: WF_THRESH_MUT_COEF** / 128).	1 – 128 (default = 48)
Wet Finger: WF_MT_DEBOUNCE	Alternative multi-touch debounce when wet finger is detected. Debounce is applied to the second and the consecutive touches but not to the first touch.	0 – 15 (default = 5)
Wet Finger: WET_FINGER_Z8_MULT	Multiplier for the Z8 peak filter. Local peak is removed if the Z8_Sum is less than 1/16 of the local	1 – 100 (default = 32)

Configurable Parameter	Description	Selection
	peak diff-count multiplied by **Wet Finger: WET_FINGER_Z8_MULT**.	
Scan Filtering: WATER_REJ_SNS_WIDTH	Sensor width for Water Rejection (Maximum value is 7). Rarely changed from default value.	0 – 7 6165X/8165X (TSG6L) (default = 0) 8168X (TSG6XL) (default = 2)
Fingers: PEAK_IGNORE_COEF	Dynamic threshold coefficient for hard press finger. Local mutual-cap peak larger than the threshold will be ignored and not processed. Setting to 0 disables this feature.	0 – 32 (default = 0)

8.13 Enhanced wet finger detection

This section guides you in tuning a system for:

- Water on a panel before a system is powered up or before wakeup.
- Water on a finger that touches a panel.
- Small water droplets.

The tuning in this section is quite time consuming, but can be used to enhance tuning done in the preceding section for wet-finger tracking. It should only be done if it is necessary to meet the system requirements.

8.13.1 Data collection

The steps in this section use data captured during a wet finger touchdown and tracking sequence. The data can be collected in TTHE. Switch to the Heat Map display mode, and configure it to display Diff Counts. Prepare a wet finger, and click the blue arrow to start collecting data in TTHE: Collect 50 packets. Fifty packets should be sufficient. Immediately after clicking the arrow, move the wet finger in a trajectory like in Figure 105.

The collected data is saved in the project's folder in the log.csv file. The file will be arranged by rows, with a timestamp for the frame collection, and a column for each column:row intersection.

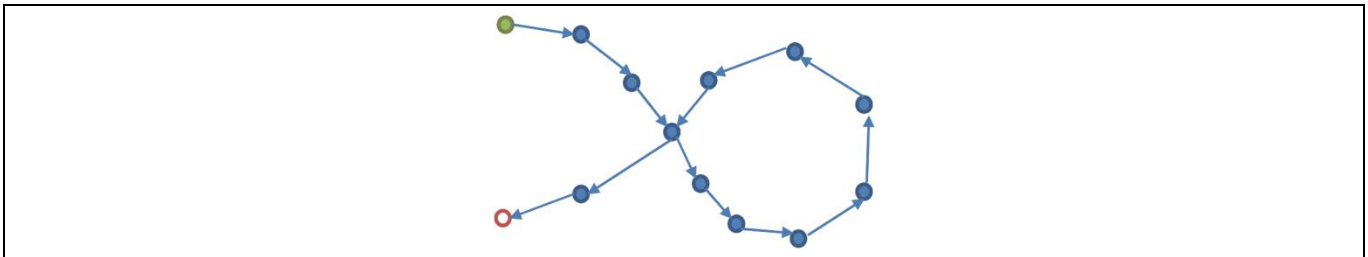
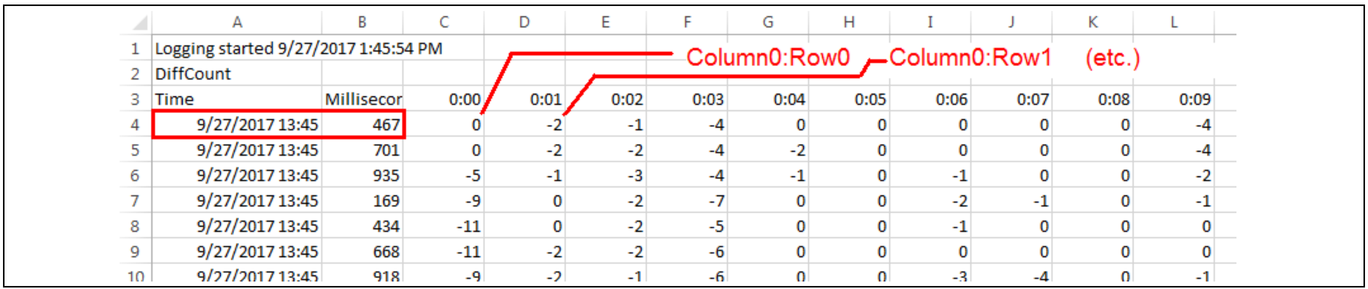


Figure 105 Track wet finger trajectory



	A	B	C	D	E	F	G	H	I	J	K	L
1	Logging started 9/27/2017 1:45:54 PM											
2	DiffCount											
3	Time	Millisecor	0:00	0:01	0:02	0:03	0:04	0:05	0:06	0:07	0:08	0:09
4	9/27/2017 13:45	467	0	-2	-1	-4	0	0	0	0	0	-4
5	9/27/2017 13:45	701	0	-2	-2	-4	-2	0	0	0	0	-4
6	9/27/2017 13:45	935	-5	-1	-3	-4	-1	0	-1	0	0	-2
7	9/27/2017 13:45	169	-9	0	-2	-7	0	0	-2	-1	0	-1
8	9/27/2017 13:45	434	-11	0	-2	-5	0	0	-1	0	0	0
9	9/27/2017 13:45	668	-11	-2	-2	-6	0	0	0	0	0	0
10	9/27/2017 13:45	918	-9	-2	-1	-6	0	0	-3	-4	0	-1

Figure 106 Portion of log.csv after capture

For each frame in the capture, rearrange the data in the csv file to create a heat map. Note that this takes significant manipulation of the spreadsheet. The collection of heat maps is the data that will be used for the steps in this section.

A quicker method is to select a single heat map in the sequence, and take a screen capture of the heat map in TTHE. If this method gives acceptable end results, then it is OK to use.

8.13.2 Water detection

The first step is to detect water on the panel.

Note the major touch peak in the heat map under analysis. It represents the location of the actual finger. Aside from this, there will be other smaller peaks that are the result of water on the panel. See Figure 107 for an example.

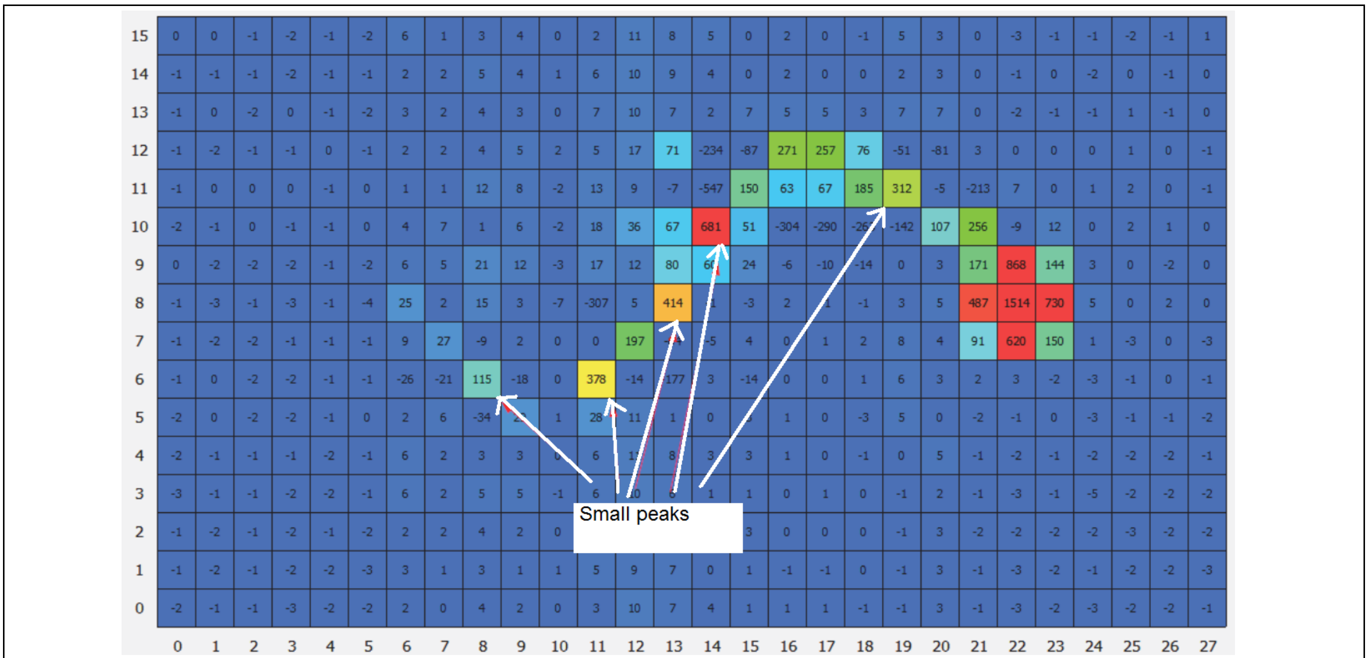


Figure 107 Small peaks in wet finger trajectory

A peak will be considered water if

- Its peak value exceeds **Wet Finger: WF_DETECT_PEAK_THR**, and
- Its Z9 sum is less than **Wet Finger: WF_DETECT_PEAK_Z9_SCALE** * **Wet Finger: WF_DETECT_PEAK_THR**, and
- Its Z8 sum is less than (1/8)***Wet Finger: WF_DETECT_PEAK_Z8_SCALE** * peak value

Look at the captured data, and adjust **Wet Finger: WF_DETECT_PEAK_THR** to a value below the small peaks, so they can be considered water. Also adjust **Wet Finger: WF_DETECT_PEAK_Z9_SCALE** such that the first Z9 condition above is met. Finally, adjust **Wet Finger: WF_DETECT_PEAK_Z8_SCALE** so that the Z8 condition is met.

Water will be detected if the number of water peaks equals or exceeds the count and debounce requirements.

Wet Finger: WF_DETECT_PEAK_COUNT sets how many water peaks must be detected in the frame before water mode is entered. The higher this value, the harder to enter water mode but the more accurate detection is. The recommended value is 1-3.

Wet Finger: WF_DETECT_PEAK_DEBOUNCE is how many frames should meet the condition of count requirement before water mode is entered. The higher this value, the harder it is to enter water mode but the more accurate detection is. The recommended value is 0-3.

The parameter **Wet Finger: WF_DETECT_PEAK_Z9_THR** is used to detect whether the peaks are caused by a real touch. Usually the touch has a higher Z9 sum than peaks caused by water. To set this value, touch the smallest required finger to a dry panel, and calculate the Z9 sum. Set the parameter to 80% of this value.

Note: There is an additional algorithm that can be helpful for filtering of false peaks with small signals during wet finger tracking. To enable it **Fingers: PEAK_IGNORE_COEF** should be set to no-zero value (up to 10. Higher value usage can cause of small real finger rejection when large finger on the panel). It will reject all peaks < ((PEAK_IGNORE_COEF * Max_Peak_Signal) / 32), where Max_Peak_Signal is peak with the maximum signal count. It works in all water level 0-3 (with and without water detected).

8.13.3 Parameters

Table 55 Enhanced wet finger detection parameters

Configurable Parameter	Description	Selection
Wet Finger: WF_DETECT_PEAK_THR	Peak threshold for detecting weak peaks. If this is zero then the feature is disabled.	0 – 32768 (default = 100)
Wet Finger: WF_DETECT_PEAK_Z9_THR	Z9 sum threshold for detecting major peaks by finger.	0 – 32768 (default = 1000)
Wet Finger: WF_DETECT_PEAK_Z9_SCALE	Z9 scale for detection weak peaks.	0 – 4 (default = 2)
Wet Finger: WF_DETECT_PEAK_Z8_SCALE	Z8 scale in 1/8 for detecting weak peaks made by water.	0 – 255 (default = 12)
Wet Finger: WF_DETECT_PEAK_DEBOUNCE	Debounce before entering wet mode.	0 – 255 (default = 3)
Wet Finger: WF_DETECT_PEAK_COUNT	Peak count on the touch trajectory.	1 – 255 (default = 2)
Fingers: PEAK_IGNORE_COEF	Dynamic threshold coefficient for hard press finger. Local mutual-cap peak larger than the threshold will be ignored and not processed. Setting to 0 disables this feature.	0 – 32 (default = 0)

8.14 Palm on startup

The palm on startup feature is only evaluated if water rejection (**Scan Filtering: WATER_REJ_ENABLE**) is not enabled. The water rejection algorithm will perform the palm on startup functionally more robustly.

The palm on startup feature, if enabled, monitors the diff-count data. If a configurable number of sensors have diff-counts less than a configurable threshold, for a configurable period, then all baselines are reset.

The requirements for this feature are highly project specific. If the module will be mounted close to horizontally, then there is a higher possibility that a conductive object may be on the panel at startup, and that it will remain there for a longer period. However, the more common case, is for a more vertical orientation, where a conductive object is unlikely to be on the panel at startup, and if it is, that it will remain for any significant period.

The recommendations for the palm on startup parameters (assuming a vertically mounted module) is as follows:

1. Set the number of sensors to the large object size (**Fingers: MAX_FAT_FINGER_SIZE**).
2. Set the startup interval to 30 seconds (30000).
3. Set the negative diff-count threshold to about 1/3 of the mutual-cap finger liftoff threshold (**Fingers: FINGER_THRESH_MUT_LO**).

8.14.1 Parameters

Table 56 Palm on startup parameters

Configurable Parameter	Description	Selection
Raw Processing: PALM_STARTUP_INTERVAL	Time interval (in ms) used by the palm on startup algorithm to trigger a baseline reset.	0 – 4294967295 (default = 0)
Raw Processing: PALM_STARTUP_NUM_OF_SENSORS	Number of sensors used by the palm on startup algorithm to trigger a baseline reset.	0 – 32767 (default = 32767)
Raw Processing: PALM_STARTUP_THRESHOLD	Diff-count negative threshold used by the palm on startup algorithm to trigger a baseline reset.	0 – 32767 (default = 100)

8.15 Stuck touch timeout

Different market segments have differing requirements. For example, consumer goods require fast response, but can easily be reset by turning off and on again; in contrast, automotive requires that the system must recover from all scenarios possible. The “stuck touch timeout” feature falls in the latter category.

Stuck touches are highly unlikely in well-tuned systems. However, some critical systems may still require a back-up if stuck touches do occur.

To tune the stuck touch algorithm, simply set the parameter **Device Setup: STUCK_TOUCH_TIMEOUT** to the required duration (units are minutes). If a touch is detected for longer than the defined duration, then the touch will be regarded as a “stuck touch” and the baselines are reset. To disable this feature, set the parameter to 0.

8.15.1 Parameters

Table 57 Stuck touch timeout parameters

Configurable Parameter	Description	Selection
Device Setup: STUCK_TOUCH_TIMEOUT	Number of minutes a touch will be detected before it is determined to be a "stuck touch" and the baselines are reset. Set to 0 to disable.	0 – 60 (default = 0)

8.16 Dynamic calibration

Dynamic recalibration ensures that the device is properly calibrated with the connected touch panel. It evaluates the calibration data of each scanning mode and compares against a target threshold at each device power-up, reset or wake up after Deep Sleep. It is not evaluated when the device is in normal operation. If the calibration data fails to meet the specified target, it will automatically execute a device calibration, and the existing calibration data in the flash memory will be replaced.

A device calibration is triggered by any of these events:

- First entry into the touch application after device programming.
- A configurable parameter that requires calibration is updated using TTHE.
- Power up or Wake up after Deep Sleep, if **Calibration: DYNAMIC_CALIBRATION_ENABLED** is enabled and the raw-count variation exceeds any of the parameters **Calibration: SAFE_RAW_RANGE_PERCENT_MC**, **Calibration: SAFE_RAW_RANGE_PERCENT_SC**, **Calibration: SAFE_RAW_RANGE_PERCENT_MC_BTN**, or **Calibration: SAFE_RAW_RANGE_PERCENT_SC_BTN**.
- Clicking Calibrate IDAC in TTHE.

Note: The device will NOT automatically recalibrate after a bootload.

8.16.1 Parameters

Table 58 Calibration parameters

Configurable Parameter	Description	Selection
Calibration: DYNAMIC_CALIBRATION_ENABLED	Enabling dynamic calibration.	Enabled / Disabled (default = Disabled)
Calibration: SAFE_RAW_RANGE_PERCENT_MC	This parameter sets the tolerable variation in charge for automatic recalibration. The value is specified in percentage and is scaled to the integer 128 (i.e. 50% = 64).	6 – 121 (default = 121)
Calibration: SAFE_RAW_RANGE_PERCENT_SC		
Calibration: SAFE_RAW_RANGE_PERCENT_MC_BTN		
Calibration: SAFE_RAW_RANGE_PERCENT_SC_BTN		

9 Glove tuning

9.1 Introduction

Glove support makes it convenient for users to navigate the touchscreen without having to take off their gloves or use expensive conductive gloves.

Gloves come in wide variety of materials and thicknesses. A sampling of different gloves is shown in Figure 108. For testing and characterization, it is recommended using standard-sized metal fingers with insulating overlays such as Delrin or polycarbonate plastic. Figure 109 shows an example of a test finger with a 6 mm overlay. Note that thickness of the overlay does not necessarily simulate the same thickness of glove – different glove materials will have different equivalent metal finger overlay thicknesses. The conversion can be determined by comparing the diff-counts of the metal finger to those of the actual glove.

Figure 108 Examples of different types of glove

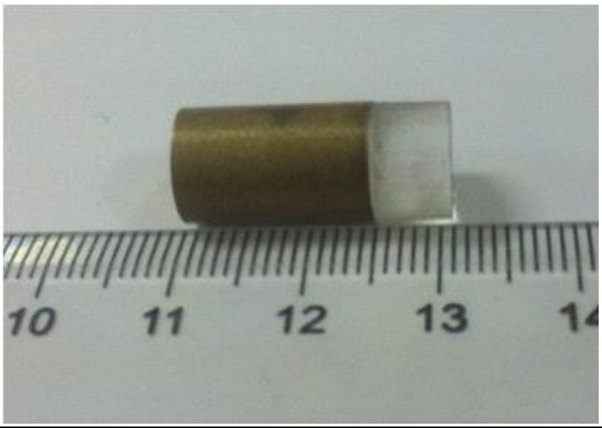


Figure 109 Metal finger with overlay

A gloved finger generates a much smaller diff-count value than a bare finger. Examples of a 14 mm metal finger touch and glove touch are shown in Figure 110. With a 2.5 mm overlay the mutual-cap peak signal and 5x5 signal sum drop by more than 80 percent. The signal count drops only modestly when further increasing the overlay thickness from 2.5 mm to 5 mm.

-3	-50	25	10	2	5	-6
5	-51	149	224	145	17	-2
67	1	914	1501	974	190	18
84	203	1590	2047	1667	340	63
62	10	1012	1653	1094	195	4
10	-53	179	299	172	31	-19
-6	-64	28	12	1	10	-4

14 mm metal finger
on panel
peak = 2047
5x5 sum = 14503

-7	-11	3	-3	-5	16	4
-14	-14	34	1	1	17	6
-2	0	138	241	174	72	26
0	1	255	395	329	123	23
0	0	168	284	210	76	23
-3	-11	40	1	15	28	11
-1	-20	3	-9	0	13	1

14 mm metal finger
with 2.5 mm overlay
peak = 395
5x5 sum = 2578

-1	-8	1	-5	-2	1	3
-8	-10	23	1	1	16	15
-1	0	104	137	91	48	21
-5	1	152	187	153	70	18
-1	1	87	122	86	46	30
-1	-4	22	12	1	6	7
-17	-12	1	-9	0	2	2

14 mm metal finger
with 5 mm overlay
peak = 187
5x5 sum = 1353

Figure 110 Examples of 14-mm finger touch signal with different overlay height

9.2 Touch modes

The touch controller supports glove detection by using three touch modes:

- Look-for-Touch (LFT) Mode – used when there is no object detected on the panel.
- Finger Mode – used when a finger object is detected on the panel.
- Glove Mode – used when a glove object is detected on the panel.

A separate set of scanning parameters, including number of TX pulses, finger size parameters, edge gain parameters, and raw-data filters are available in Glove Mode. These parameters are independent of those in the Finger Mode and they are used only when the device is in Glove or Look-for-Touch Mode. Additionally, there are separate touch debounce and threshold parameters for Glove Mode, which are only used when a glove touch is detected.

The touch controller behaves differently according to the active mode:

- Because gloves have lower signals than fingers, there are a separate set of scanning parameters to detect gloves. For example, the number of TX pulses to detect gloves is determined by **TSS: TX_PULSES_GLOVE_MC** and **TSS: TX_PULSES_GLOVE_SC** rather than **TSS: TX_PULSES_MC** and **TSS: TX_PULSES_SC**.
- With no object detected and glove support is enabled, the two touch controller families behave differently:
  - CYAT6165X/8165X (TSG6L) controllers always uses the glove-type scanning parameters.
  - CYAT8168X (TSG6XL) controllers can use either glove or finger scanning parameters, see **Touch Mode: TOUCHMODE_CONFIG_IN_LFT**.
- The finger-type scanning parameters are only activated in Finger Mode, after a finger is detected on the panel.
- If a glove touch is detected, it is reported as a “Thick Glove” touch type. See the Technical Reference Manual, document 001-99382, for details about touch reporting.
- TTHE displays a glove touch with a hatched marker rather than a solid-color marker.
- In Glove Mode, a maximum of two touches may be reported, compared to a maximum of ten (configurable in **Device Setup: MAX_REPORTED_TOUCH_NUM**) for fingers.

Table 59 summarizes the touch modes.

Table 59 Touch modes summary

Item	Look-for-Touch mode	Finger Mode	Glove Mode
Scanning and filtering	CYAT6165X/8165X (TSG6L) uses glove parameters CYAT8168X (TSG6XL) depends on **Touch Mode: TOUCHMODE_CONFIG_IN_LFT**	Finger parameters	Glove parameters
Thresholds and debounce	N/A	Finger parameters	Glove parameters
Maximum number of reported objects	N/A	Up to 10 (configurable)	Up to 2 (configurable)
Reported touch type	None	Finger	Thick Glove
TTHE indicator	None	Solid dot	Hatched dot

9.3 Transitioning between touch modes

The touch mode is determined by several factors: are gloves supported, is an object on the panel, the size and characteristics of the diff-count heat map, and timing. There are parameters for each of these factors.

9.3.1 Enabling/disabling glove detection

To enable glove tracking, set the parameter **Touch Mode: TOUCHMODE_CONFIG** to “FingerAndGlove”. To disable it, set it to “FingerOnly”. When disabled, all the glove tuning parameters are unused and may be left at their default values.

When glove tracking is enabled, and no object is on the panel, the controller scans the panel with glove-type parameters. For example, the number of TX pulses is determined by **TSS: TX_PULSES_GLOVE_SC** and **TSS: TX_PULSES_GLOVE_MC** rather than **TSS: TX_PULSES_MC** and **TSS: TX_PULSES_SC**. The same is true for all parameters that have a glove equivalent.

9.3.2 Touch mode switching

The Glove and Finger mode state diagram is shown in Figure 111. From Look-for-Touch (LFT) Mode, the controller may transition to Glove Mode or Finger Mode, depending on the characteristics of the detected object. Also, if in Glove Mode, it may change directly to Finger Mode if a strong signal is detected. There is no direct transition from Finger Mode to Glove Mode.

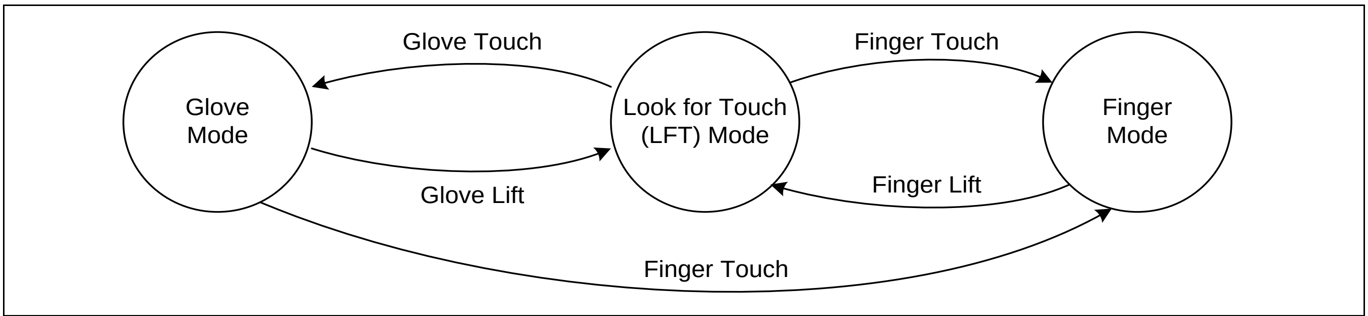


Figure 111 Touch modes switching diagram

There are two conditions for mode transitions: signal level and timing (debounce).

Once the device switches to a new touch mode, it applies the specific set of detection criteria of the new touch type to look for a valid touch object. If the device enters Finger Mode, a valid finger touch must meet the threshold requirement as described in Section 6.6 (signal scaling) and Section 7.1 (threshold). If the device enters Glove Mode, it then applies the criteria described in Section 9.3.2.2 to look for a valid glove touch.

9.3.2.1 Signal level conditions

The mode is determined by the mutual-cap peak diff-count. If it exceeds **Touch Mode: TOUCHMODE_GLOVE_HTI**, then (after debounce), the controller changes to Finger Mode. If it is below **Touch Mode: TOUCHMODE_GLOVE_HTI** but above the glove threshold **Gloves: GLOVES_THRESH_MUT_HI**, then Glove Mode is activated.

Switching from Glove Mode to Finger Mode happens when the mutual-cap peak signal exceeds **Touch Mode: TOUCHMODE_GLOVE_HTI**. Take the example of a finger touching down the panel while a glove touch is currently being reported. If the mutual-cap signal of the finger is higher than **Touch Mode: TOUCHMODE_GLOVE_HTI**, the glove touch will be rejected and not reported to the host. The finger will be reported to the host if it meets the finger detection requirements. Make sure that **Fingers: FINGER_THRESH_MUT_HI** is properly tuned for thin glove (thickness < 1 mm) for reliable thin glove tracking in Finger mode. Otherwise, a thin glove touch will cause the device switching from Glove Mode to Finger Mode, but it is then rejected for not meeting the finger threshold.

Switching from Finger Mode to Glove Mode requires first exiting Finger Mode back to LFT Mode and then entering Glove Mode. Therefore, all finger touches must be removed from the panel. The subsequent glove touch mutual-cap diff-count must be less than **Fingers: FINGER_THRESH_MUT_HI** preventing the touch object to be identified as a finger touch. It must also be lower than **Touch Mode: TOUCHMODE_GLOVE_HTI** and greater than **Gloves: GLOVES_THRESH_MUT_HI**.

Figure 112 shows how the controller uses the mutual-cap difference counts and parameters to select mode and determine which objects are reported.

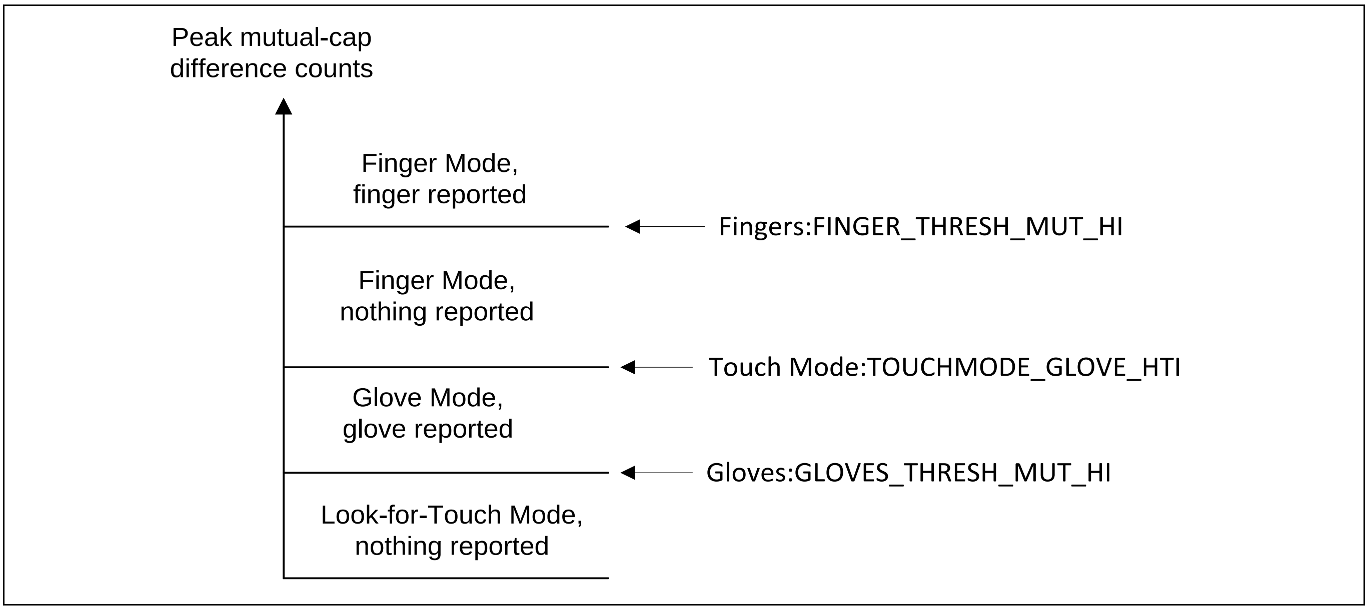


Figure 112 Mode and object reporting

9.3.2.2 Debounce conditions

This section describes the debounce for transitioning between touch modes. In most cases, the default debounce parameters will be acceptable, and tuning will not be required. This section describes the parameters in case the user wants to customize the time it takes to transition from one mode to another.

9.3.2.2.1 Look-for-touch to Glove mode and back

Figure 113 shows a simple case of a glove touch and release from a panel. After the touch, the controller waits for the larger of the two debounce periods **Gloves: GLOVES_FT_DEBOUNCE** and **Touch Mode: TOUCHMODE_GLOVE_SWITCH_DEBOUNCE**, switches to Glove Mode, and reports the glove touch. When the touch is released, the liftoff debounce **Gloves: GLOVE_LIFTOFF_DEBOUNCE** is applied, the touch reporting is halted. After **Touch Mode: TOUCHMODE_GLOVE_EXIT_DELAY**, the controller goes back to Look-for-Touch mode with glove scanning parameters applied.

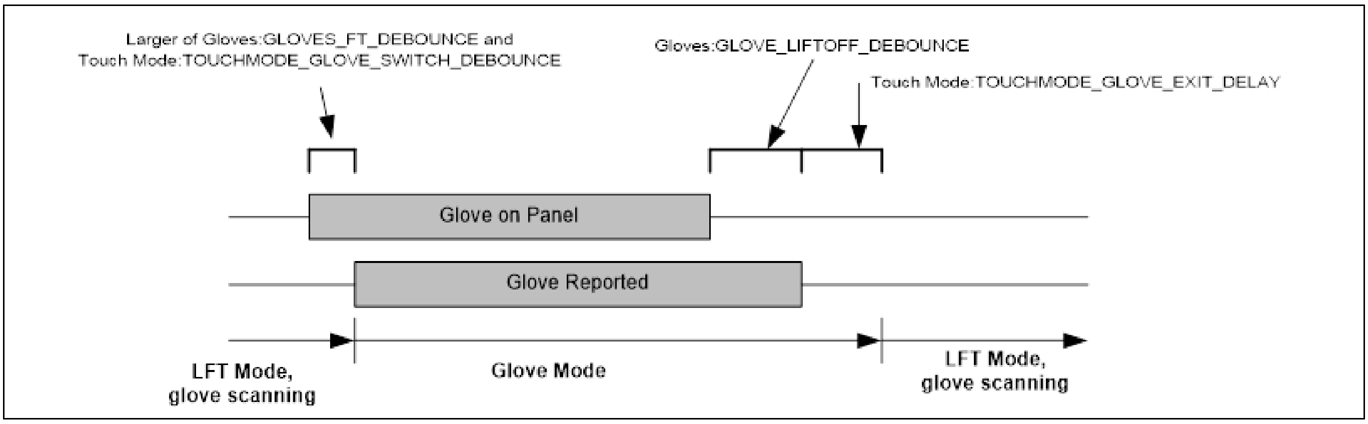


Figure 113 Glove touch and release

9.3.2.2.2 Look-for-touch to Finger Mode and back

Figure 114 shows the case of a finger touching and being removed from a panel. Initially, the controller is in Look-for-Touch mode, with the scanning parameters set to detect a possible glove touch. When a finger touches the panel, the controller applies the debounce parameter **Touch Mode: TOUCHMODE_FINGER_SWITCH_DEBOUNCE**, and then reports the finger. The scanning is still set for Glove Mode, until **Touch Mode: TOUCHMODE_FRAME_NUM_TO_CONFIRM_FINGER_MODE** completes. At this point, the controller is in finger mode, such as would be the case if glove support is disabled.

When the finger is lifted, the controller applies the normal **Fingers: FINGER_LIFTOFF_DEBOUNCE** time, then stops reporting the finger. The controller continues finger scanning until **Touch Mode: TOUCHMODE_FINGER_EXIT_DELAY** has passed, then goes back to glove scanning. The delay before going back to glove scanning gives better performance in cases where a finger is tapping quickly on the screen.

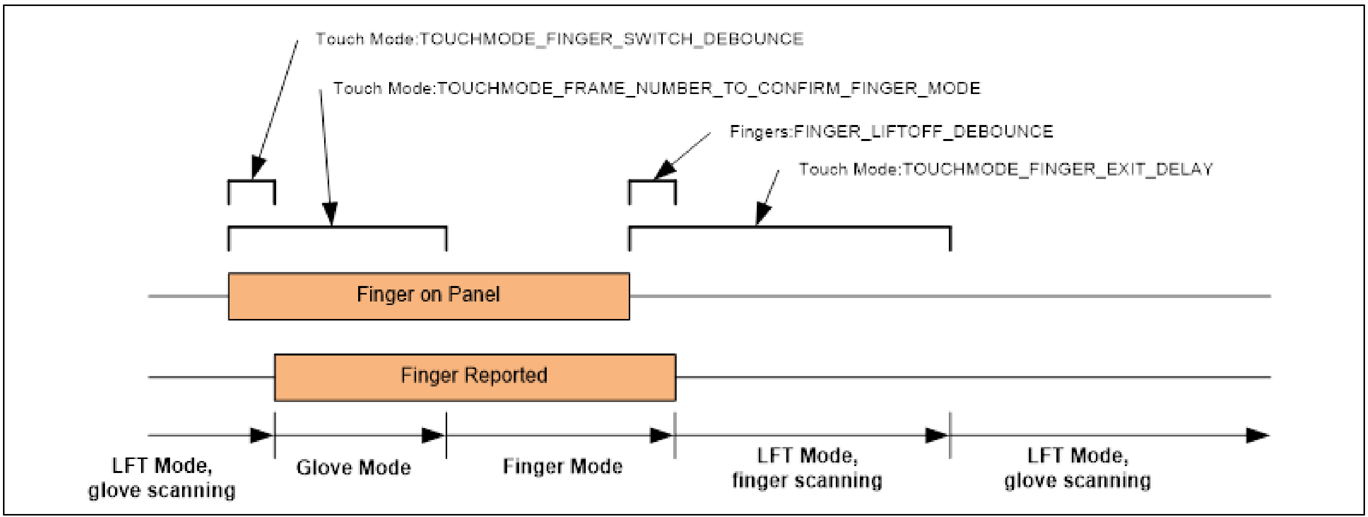


Figure 114 Finger touch and lift

9.3.2.2.3 Glove-to-Finger mode

Finger Mode has priority over Glove Mode. If a finger touches the screen while a glove is currently being reported, then the glove touch is rejected and the controller switches to finger mode.

9.3.2.2.4 Finger-to-Glove mode

There is no direct transition from Finger Mode to Glove Mode. To reach Glove Mode, the finger must be lifted long enough for the controller to switch to Look-for-Touch mode.

9.3.3 Glove tuning

This section gives step-by-step procedures for glove tuning.

9.3.3.1 Glove Tuning Setup

As discussed in Section 9.1, the glove peak signal count can be as little as one-twentieth of that of a normal finger touch depending on the glove thickness. Follow these steps to set up for the glove tuning:

1. Enable glove support by setting **Touch Mode: TOUCHMODE_CONFIG** to "FingerAndGlove".
2. Disable Water Rejection by setting **Scan Filtering: WATER_REJ_ENABLE** to Disabled. This ensures that the glove touch will not be baselined out during tuning.
3. Disable all the signal filtering in **Raw Processing: GLOVE_MC_RAW_FILTER_MASK** and **Raw Processing: GLOVE_SC_RAW_FILTER_MASK**. This ensures that the glove signal is not filtered out during tuning.
4. Make sure that tuning is performed in a quiet environment. No display noise, charger noise, or any other noise should be present. Noise mitigation should be considered only after completing the initial glove tuning.
5. Configure TTHE to display DiffCounts in Heat Map mode. Make sure self-cap counts are displayed by choosing "Self" Scan Type in the Touch Display Settings window. Touch the screen with a small finger, and note the reported self-cap counts. The counts will not update after the finger is on the panel, because water rejection was disabled earlier. However, the self-cap counts can still be seen.
6. Initialize **Gloves: GLOVES_THRESH_SELF** to less than 5% of the small finger average self-cap diff-count. This ensures that the device will enter Active mode from Active LFT or Low-Power mode upon detecting glove touch.

7. Prepare gloves with thickness between 1 mm and the maximum target thickness for tuning. Glove with thickness less than 1 mm should be tuned as normal finger touch.
8. Initialize the TX pulses control parameters to their panel (and slider and button) equivalent, for example:

$$\text{TSS: TX\_PULSES\_GLOVE\_MC} = \text{TSS: TX\_PULSES\_MC}$$

Equation 56

9.3.3.2 Glove thresholds

The primary condition for Glove Mode vs. Finger Mode is the peak diff-count. For the first touch, if the diff-count is high, above the threshold specified by **Touch Mode: TOUCHMODE_GLOVE_HTI**, then the controller stays in Finger Mode. If the diff-count is below this threshold but above **Gloves: GLOVES_THRESH_MUT_HI**, then the diff-count condition for Glove Mode is met.

$$\text{Touch Mode: TOUCHMODE\_GLOVE\_HTI} > \textit{Glove Peak Diffcount} > \textit{Gloves}: \textit{GLOVES}\text{\_THRESH\_MUT\_HI}$$

Equation 57

Once in Glove Mode, subsequent touches will still be reported as gloves if they do not exceed the finger threshold, even if they exceed **Touch Mode: TOUCHMODE_GLOVE_HTI**. The diff-count condition has hysteresis: whilst the peak diff-count remains above **Gloves: GLOVES_THRESH_MUT_LO**, the Glove Mode condition is still met. This hysteresis is for the same purpose as the Finger Mode hysteresis defined by the parameters **Fingers: FINGER_THRESH_MUT_HI** and **Fingers: FINGER_THRESH_MUT_LO**.

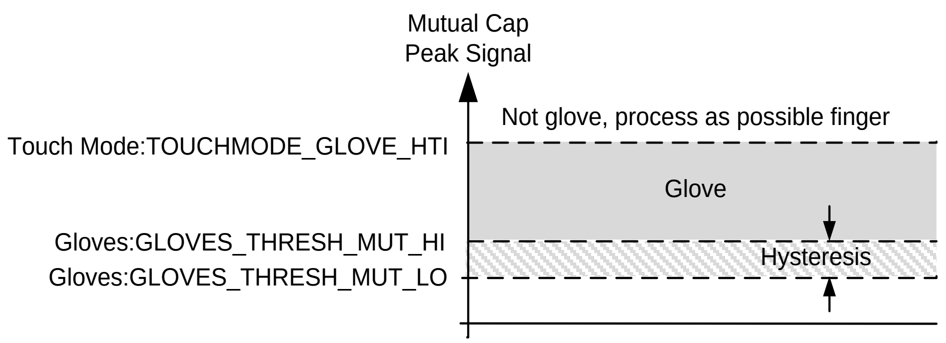


Figure 115 Threshold levels for glove detection

Tuning Steps: Use the following steps to set the threshold parameters:

1. Temporarily disable the surrounding sensor sum condition by setting parameter **Gloves: GLOVES_Z8_FILT_SCALE** to zero.
2. Temporarily disable the 3x3 sum condition by setting **Gloves: GLOVES_Z9_FILT_SCALE** to zero.
3. Use the thickest glove to be supported. Display diff-counts in TTHE, and place the thick glove at a high touch intensity (HTI) position near the center of the panel. See Figure 47 for the definition of HTI and LTI position.
4. Note the mutual-cap diff-count for the thickest glove in the HTI position. If it is well below 100 counts, adjust **TSS: SCALE_FACT_MC** such that the mutual-cap signal count of the thickest glove at the HTI position is about 100. This may affect the Finger Mode mutual-cap finger threshold, raw-data filter thresholds, and baseline threshold. Adjust these Finger Mode mutual-cap thresholds accordingly.
5. Move the gloved finger around the panel, noting the maximum and minimum peak diff-counts.

6. Set **Gloves: GLOVES_THRESH_MUT_HI** slightly lower than the minimum noted peak diff-count, and **Gloves: GLOVES_THRESH_MUT_LO** to be slightly lower still, to provide debounce. For example:

$$\text{Gloves: GLOVES\_THRESH\_MUT\_HI} = 0.8 \cdot \text{Thickest Glove LTI DiffCount}$$

Equation 58

$$\text{Gloves: GLOVES\_THRESH\_MUT\_LO} = 0.65 \cdot \text{Thickest Glove LTI DiffCount}$$

Equation 59

7. Use the thinnest glove that is to be reported as glove. Move it around the panel, watching diff-counts. Set the parameter **Touch Mode: TOUCHMODE_GLOVE_HTI** to be 10% larger than the maximum noted peak diff-count. Note that if there is a large difference between **Fingers: FINGER_THRESH_MUT_HI** and **Touch Mode: TOUCHMODE_GLOVE_HTI**, then there may be a dead zone in which thin gloves are not reported. Do not leave a large separation between these two parameters' values.

$$\text{Touch Mode: TOUCHMODE\_GLOVE\_HTI} = 1.1 \cdot \text{Thinnest Glove HTI DiffCount}$$

Equation 60

8. Follow the procedure from Section 7.4 to determine the self-cap glove thresholds using the thickest glove touch, also see Equation 61.

- If water rejection is enabled, set **Gloves: GLOVES_THRESH_SELF**.
- If self-cap LFT is enabled, proper configuration of **Touch Mode: TOUCHMODE_LFT_SELF_THRSH** ensures that the thickest glove can be detected in the self-cap LFT state and Low-Power state, and the device will promptly transition to the mutual-cap LFT state.

$$\text{Touch Mode: TOUCHMODE\_LFT\_SELF\_THRESH} < \text{Gloves: GLOVES\_THRESH\_SELF} < \text{Fingers: FINGER\_THRESH\_SELF}$$

Equation 61

9. Evaluate the glove tracking with different thick gloves in the Line Drawing display mode. If broken glove tracks or unreliable glove detection is observed, try the following changes one at a time.

- Decrease both **Gloves: GLOVES_THRESH_MUT_HI** and **Gloves: GLOVES_THRESH_MUT_LO** proportionally.
- Increase **Touch Mode: TOUCHMODE_GLOVE_HTI**.
- Decrease **Gloves: GLOVES_Z8_FILTER_SCALE**.
- Decrease **Gloves: GLOVES_Z9_FILT_SCALE**.

10. Restore the original value of **Gloves: GLOVES_Z8_FILT_SCALE**.
11. Restore the original value of **Gloves: GLOVES_Z9_FILT_SCALE**.

9.3.3.3 Glove mutual-cap raw-data IIR filter thresholds

Follow these steps to tune the mutual-cap raw-data IIR filter threshold:

1. Enable the IIR filter in **Raw Processing: GLOVE_MC_RAW_FILTER_MASK: IIR Filter**.

2. Set the raw-data IIR filter threshold **Raw Processing: GLOVE_MC_RAW_IIR_THRESH** to 75% of the **Gloves: GLOVES_THRESH_MUT_HI**.

$$\text{Raw Processing: GLOVE\_MC\_RAW\_MC\_THRESH} = 0.75 * \text{Gloves: GLOVES\_THRESH\_MUT\_HI}$$

Equation 62

3. Evaluate the glove tracking with different thick gloves. If a broken trace is observed, try the following changes one at a time:

- Decrease **Gloves: GLOVES_THRESH_MUT_HI** to just 1 count above **Raw Processing: GLOVE_MC_RAW_IIR_THRESH**.
- Increase **Raw Processing: GLOVE_MC_RAW_IIR_COEF**.
- Decrease **Raw Processing: GLOVE_MC_RAW_IIR_THRESH**.
- Reduce the parameters **Scan Filtering: XY_FILT_IIR_COEF_FAST** and **Scan Filtering: XY_FILT_IIR_COEF_SLOW** to strengthen IIR position filtering.

9.3.3.4 Self cap glove fine-tuning

Configure the self-cap tuning if Water Rejection is enabled. The device has limited water rejection capability when it is in Glove Mode. The weaker glove signal makes it difficult to distinguish between a large water droplet and a glove touchdown. However, combining with the raw-count filters and baseline filter, Water Rejection is helpful in minimizing some of the environmental noise and signal residue impact without compromising the glove touch detection. Complete the previous sections to configure the glove detection with mutual-cap scan before proceeding the self-cap tuning.

1. Set **Scan Filtering: WATER_REJ_ENABLE** to "Enabled".
2. Enable the IIR filter in **Raw Processing: GLOVE_SC_RAW_FILTER_MASK: IIR Filter**. Set the raw-count IIR filter threshold **Raw Processing: GLOVE_SC_RAW_IIR_THRESH** to 75% of the **GLOVES: GLOVES_THRSH_SELF**.

$$\text{Raw Processing: GLOVE\_SC\_RAW\_IIR\_THRESH} = 0.75 * \text{Gloves: GLOVES\_THRESH\_SELF}$$

Equation 63

3. Evaluate the glove tracking with different thick gloves. If a broken glove trace is observed in the Line Drawing display mode, try the following changes one at a time:

- Decrease **Gloves: GLOVES_THRESH_SELF** (keep **Touch Mode: TOUCHMODE_LFT_SELF_THRSH** at least 1 lower)
- Decrease **Raw Processing: GLOVE_SC_RAW_IIR_THRESH**.
- Increase **Raw Processing: GLOVE_SC_RAW_IIR_COEF**.

9.3.3.5 Surrounding sensor sum

For a valid glove signal, the peak must not be too narrow. This condition is checked by comparing the sum of the eight surrounding sensors (Z8 sum) to the peak value. To allow tuning and scaling, the peak value is multiplied by the parameter **Gloves: GLOVES_Z8_FILT_SCALE**. The Z8 sum must exceed the product.

$$Z8_{\text{Sum}} > \text{Glove Peak Diffcount} \cdot \text{Gloves: GLOVES\_Z8\_FILTER\_SCALE}$$

Equation 64

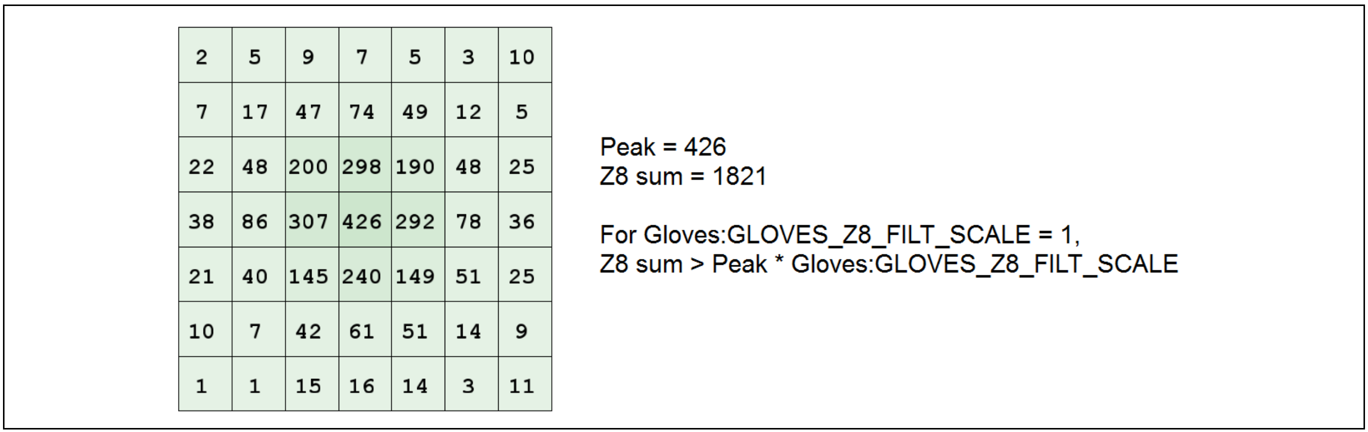


Figure 116 Example of valid Z8 sum condition

In some cases, a burst of noise may cause a false glove touch to be registered. If the noise burst affects only a single sensor, the surrounding sensor condition may be used to suppress the false touch. Increase the parameter **Gloves: GLOVES_Z8_FILT_SCALE** until a good balance between false and real glove touches is achieved. In most cases, the default value will be acceptable.

9.3.3.6 3x3 sum

Like normal finger touch detection, the calculated mass of the mutual-cap diff-count 3x3 signal sum must be greater than the product of the lift-off threshold and **Gloves: GLOVES_Z9_FILT_SCALE**.

$$\text{3x3 sum} > \text{Gloves: GLOVES\_THRESH\_MUT\_LO} \cdot \text{Gloves: GLOVES\_Z9\_FILT\_SCALE}$$

Equation 65

For glove touch at a panel edge, a 2x3 grid is used to instead of 3x3. At a corner, a 2x2 grid is used. When a glove touch is at the edge or corner of a panel, the scaling factor used for the finger threshold is modified according to Table 60.

Table 60 Glove threshold scale factor at core, edge, and corner

Gloves: GLOVES_Z9_FILT_SCALE	Scaling Factor		
	Panel Core	Panel Edge	Panel Corner
4	4x	2x	1x
3	2x	1x	Disabled
2	2x	1x	Disabled
1	1x	Disabled	Disabled
0	1x	Disabled	Disabled

Like the Z8 condition, the Z9 condition may prevent noise-induced false touches. Adjust **Gloves: GLOVES_Z9_FILT_SCALE** to achieve a good balance at the panel core, edges, and corners.

9.3.3.7 Glove touch debounce

For glove touch debounce, the first touch must be valid and stable for a period specified by **Gloves: GLOVES_FT_DEBOUNCE**. Subsequent touches must be valid and stable for a (typically shorter) period specified by **Gloves: GLOVES_MT_DEBOUNCE**. When a glove touch is removed, it must stay removed for **Gloves: GLOVE_LIFTOFF_DEBOUNCE**.

Figure 117 and Figure 118 show examples of the combined effects of mode and touch debounce. Figure 117 shows an illustration of first glove touch, second glove touch, touch removal, and then another touch. Figure 118 is similar, except the final touch happens before **Touch Mode: TOUCHMODE_GLOVE_EXIT_DELAY** expires. This gives a faster reporting of the final touch, because the mode debounce is not needed.

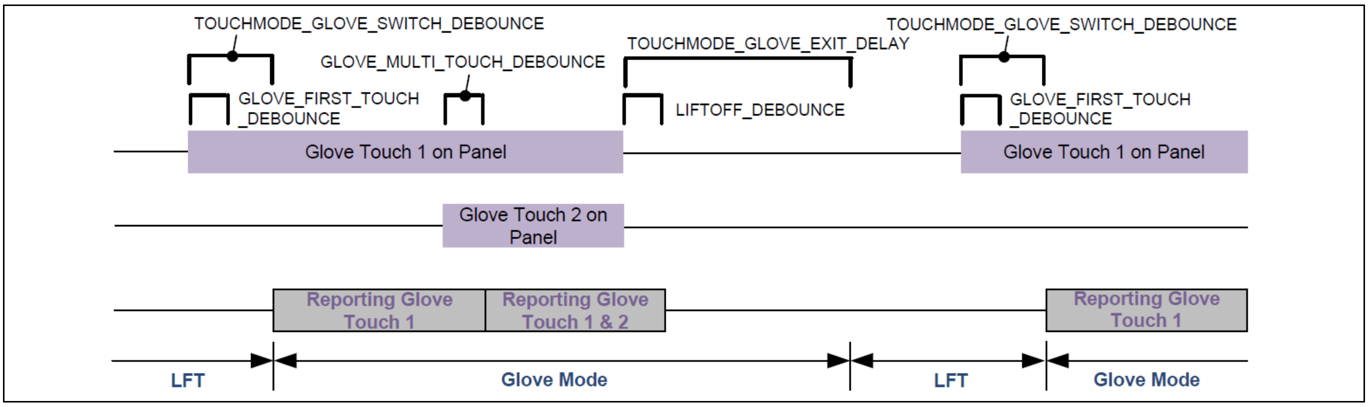


Figure 117 Glove touch debounce illustration 1

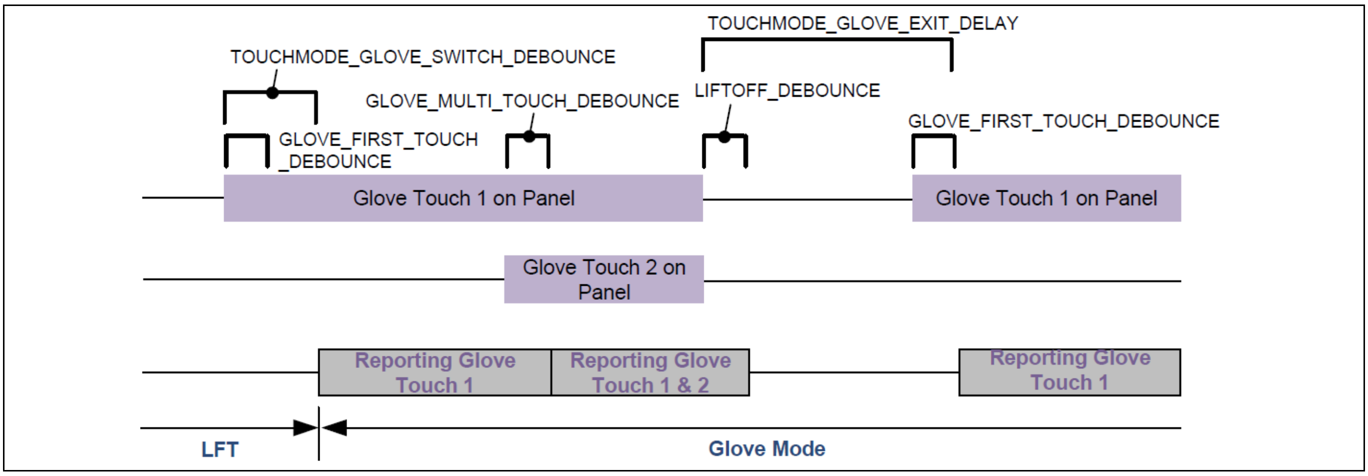


Figure 118 Glove touch debounce illustration 2

9.4 Glove buttons

Glove buttons are described as part of the buttons chapter in Section 10.1.1.

9.5 Parameters

Table 61 Touch modes summary

Configurable Parameter	Description	Selection
Gloves: GLOVES_THRESH_MUT_HI	Mutual-cap Glove Threshold ON. Used when glove was not detected yet.	0 - 32767 (default = 180)
Gloves: GLOVES_THRESH_MUT_LO	Mutual-cap Glove Threshold OFF. Used when glove was previously detected.	0 - 32767 (default = 150)
Gloves: GLOVES_THRESH_SELF	Self-cap Gloves Threshold.	0 - 32767 (default = 65)
Gloves: GLOVES_Z9_FILT_SCALE	Z9 filter scale for normal gloves when Charger Armor is not active. Peak with Z9<(scale*GLOVES_THRESH_MUT_LO) will be removed.	0 - 4 (default = 4)
Gloves: GLOVES_Z8_FILT_SCALE	Z8 filter scale for gloves detection. A valid touch must satisfy this requirement: Z8 Sum > Glove Peak Diff-count * GLOVES_Z8_FILT_SCALE.	0 - 127 (default = 2)
Gloves: GLOVES_MIN_FAT_SIZE	Minimum number of contiguous activated panel intersections that define a fat touch. It is used when a fat touch was not detected yet.	0 - 255 (default = 10)
Gloves: GLOVES_MIN_FAT_SIZE_HYST	Hysteresis applied to the minimum number of intersections defining a fat touch; on detection of a fat touch reducing the minimum fat touch threshold.	0 - 255 (default = 2)
Gloves: GLOVES_MAX_FAT_SIZE	Minimum number of contiguous activated panel intersections that define a large object. It is used when a large object was not detected yet. Fat touch detection takes priority in determining whether to increase or reduce the threshold. Make sure the condition (0 <= GLOVES_MAX_FAT_SIZE_HYST <= 1/2 of GLOVES_MAX_FAT_SIZE_ON) is true when tuning.	0 - 255 (default = 35)
Gloves: GLOVES_MAX_FAT_SIZE_HYST	Hysteresis applied to minimum number of intersections defining a large object; on detection of a large object reducing the maximum fat touch threshold. Fat touch detection takes priority in determining whether to increase or reduce the threshold. Make sure the condition (0 <= GLOVES_MAX_FAT_SIZE_HYST <= 1/2 of GLOVES_MAX_FAT_SIZE_ON) is true when tuning.	0 - 255 (default = 8)
Gloves: GLOVES_FT_DEBOUNCE	Number of consecutive refresh cycles for which a glove touch must be detected prior to being reported. Applies to the first glove touch.	0 - 63 (default = 3)
Gloves: GLOVES_FT_DEBOUNCE_EDGE _MASK	Individual first glove touch debounce enable for each of the 4 edges.	Enabled / Disabled

Configurable Parameter	Description	Selection
		(default = Enabled)
Gloves: GLOVES_MT_DEBOUNCE	Number of consecutive refresh cycles for which a glove touch must be detected prior to being reported. Applies to the second and successive glove touches.	0 - 63 (default = 3)
Gloves: GLOVE_LIFTOFF_DEBOUNCE	Number of consecutive refresh cycles for which a glove touch must not be detected before the lack of touch is identified as a liftoff.	0 – 63 (default = 5)
Touch Mode: TOUCHMODE_FRAME_NUM_TO_CONFIRM_FINGER_MODE	Number of frames which must continuously satisfy the absolute finger criteria before finger scanning mode is locked in.	1 - 255 (default = 10)
Touch Mode: TOUCHMODE_CONFIG_IN_LFT CYAT8168X (TSG6XL) only	CYAT8168X (TSG6XL) only. Controls whether to use glove or finger scanning parameters in the LFT state.	GloveConfig / FingerConfig (default = GloveConfig)

10 Button detection

10.1 Standard buttons

The touch controller can detect CAPSENSE™ buttons independently of the main touchscreen area. Up to ten buttons are supported. Buttons can be designed for self-cap, mutual-cap, or hybrid (both self and mutual) detection.

Conceptually, tuning for a button is like tuning for a finger: the scanning parameters such as TX frequency and pulse counts are set up, and thresholds are chosen based on signal response. The button scan happens independently of the main panel scan.

In this tuning guide, most button tuning parameters are listed along with their main-panel counterparts. The parameters used for buttons generally contain the letters "BTN" in the parameter name. The exceptions are the touch thresholds, and a special parameter controlling whether to process buttons when a touch is detected on the touchscreen (**Buttons: BTN_PROCESS_IF_TOUCH_DETECTED**).

There are touch detection thresholds for mutual-cap and self-cap, and thresholds for activating and releasing the button (giving hysteresis). Parameters containing the letters "LS" are low-sensitivity and work with fingers; parameters containing the letters "HS" are high-sensitivity and work with gloved fingers. The process for tuning them is the same as the mutual-cap finger threshold (see Section 7.2). Note that if hybrid scanning mode is selected (**TSS: SCANNING_MODE_BUTTON**), a touch is only reported to the host when both mutual-cap and self-cap diff-count are greater than their corresponding threshold.

10.1.1 Parameters

Table 62 Standard button parameters

Configurable Parameter	Description	Selection
Buttons: BTN_LS_ON_THRSH_MUT_n	Button mutual-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 80)
Buttons: BTN_LS_OFF_THRSH_MUT_n	Button mutual-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 70)
Buttons: BTN_LS_ON_THRSH_SELF_n	Button self-cap scan touchdown threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 65)
Buttons: BTN_LS_OFF_THRSH_SELF_n	Button self-cap scan liftoff threshold for Low Sensitivity Mode (Finger) for button N.	5 – 32767 (default = 40)
Buttons: BTN_LS_TD_DEBOUNCE	Number of consecutive refresh cycles for which button event must be detected prior to being reported. Rarely changed from default value.	0 – 255 (default = 0)
Buttons: BTN_PROCESS_IF_TOUCH_DETECTED	Allow buttons processing when touch detected on the panel. Rarely changed from default value.	Enabled / Disabled (default = Disabled)

10.2 Glove buttons

Glove processing for buttons is a simplified version of the process on the touchscreen. Glove processing is enabled using the same parameter (set **Touch Mode: TOUCHMODE_CONFIG** to "FingerAndGlove").

When detecting gloves, the button is in a high-sensitivity mode. The glove button parameters therefore contain the acronym "HS". When detecting fingers, the buttons are in low-sensitivity mode, and the parameters contain "LS".

Use the following procedure to determine the thresholds. Note, use the mutual-cap or self-cap parameters depending on the button scanning method (**TSS: SCANNING_MODE_BUTTON**).

1. In the TTHE Heat Map display mode, set DataType to "DiffCounts".
2. Using the smallest finger and thickest glove to be detected, note the average diff count.
3. Set **Glove Buttons: BTN_HS_ON_THRSH_xxx_n** and **Glove Buttons: BTN_HIGHSEN_MODE_THRSH_xxx** to 90% of the average diff count.
4. Set **Glove Buttons: BTN_HS_OFF_THRSH_xxx_n** to 70% of the average diff count.
5. Set **Glove Buttons: BTN_LOWSEN_MODE_THRSH_xxx** to **Buttons: BTN_LS_OFF_THRSH_xxx_n**.

The remaining parameters are for debounce. Their settings are project specific. The parameters are listed below along with some trade-offs:

- **Glove Buttons: BTN_HS_TOUCHDOWN_DEBOUNCE**:
  - Increase to give better noise or false-touch immunity.
  - Decrease to improve response time.
- **Glove Buttons: GLOVE_BTN_FORBID_DEBOUNCE**:
  - Increase to give better noise or false-touch immunity.
  - Increase if a finger touch lifting off sometimes reports a glove touch
  - Decrease to improve glove response time after a finger lift-off.
- **Glove Buttons: GLOVE_BTN_MODE_SWITCH_DEBOUNCE**:
  - Increase to give better noise or false-touch immunity.
  - Increase if a finger touch lifting off sometimes reports a glove touch
  - Decrease to improve glove response time after a finger lift-off.

10.2.1 Parameters

Table 63 Glove button parameters

Configurable Parameter	Description	Selection
Touch Mode: TOUCHMODE_CONFIG	Select whether gloves objects can be detected, or fingers only.	FingerOnly / FingerAndGlove (default = FingerAndGlove)
Glove Buttons: BTN_HS_ON_THRSH_MUT_n	Button mutual-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 20)
Glove Buttons: BTN_HS_OFF_THRSH_MUT_n	Button mutual-cap scan liftoff threshold for High Sensitivity Mode (Glove)for button N.	5 – 32767 (default = 10)

Configurable Parameter	Description	Selection
Glove Buttons: BTN_HS_ON_THRSH_SELF_n	Button self-cap scan touchdown threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 30)
Glove Buttons: BTN_HS_OFF_THRSH_SELF_n	Button self-cap scan liftoff threshold for High Sensitivity Mode (Glove) for button N.	5 – 32767 (default = 10)
Glove Buttons: BTN_HS_TOUCHDOWN_DEBOUNCE	Number of consecutive refresh cycles for which button event must be detected prior to being reported. Rarely changed from default value.	0 – 255 (default = 1)
Glove Buttons: BTN_HIGHSEN_MODE_THRSH_MUT	Mutual-cap threshold to enter high-sensitive mode.	0 – 32767 (default = 40)
Glove Buttons: BTN_HIGHSEN_MODE_THRSH_SELF	Self-cap threshold to enter high-sensitive mode.	0 – 32767 (default = 40)
Glove Buttons: BTN_LOWSEN_MODE_THRSH_MUT	Mutual-cap threshold to enter low-sensitive mode.	0 – 32767 (default = 200)
Glove Buttons: BTN_LOWSEN_MODE_THRSH_SELF	Self-cap threshold to enter low-sensitive mode.	0 – 32767 (default = 350)
Glove Buttons: GLOVE_BTN_FORBID_DEBOUNCE	Number of consecutive refresh cycles after all regular touch liftoff for which a glove touch is not detectable. Rarely changed from default value.	0 – 255 (default = 10)
Glove Buttons: GLOVE_BTN_MODE_SWITCH_DEBOUNCE	Number of consecutive refresh cycles for mode switch from finger mode to glove mode. Rarely changed from default value.	0 – 255 (default = 1)

10.3 Wake-up button detection (CYAT6165X/8165 (TSG6L) only)

A wake-up button allows the system to go to an extremely low-power state, but still wake up via the touchscreen. When a wake-up button is enabled, it is automatically scanned when the touch controller is placed in the deep sleep state. From the deep sleep state, the system briefly wakes up and scans the button. If no touch is detected, then the touch controller returns to the deep sleep state until the next wake-up button scan. However, if a validated touch is detected, the touch controller asserts a GPIO pin. This pin is monitored by the host, which can then switch the touch system back to the active state, if required.

There are a few important points related to the wake-up button:

- Currently only the TSG6L base FW supports a wake-up button.
- TSG6XL will add this feature in base FW version 1.5.
- The wake-up button will only function if at least one button is present in the project.
- The touch controller does not wake from deep sleep, the host must send this command as needed.
- A secondary reset signal is available if the wake-up button is held for a configurable period.
- When the touch controller is in the active state, behavior is as a standard CAPSENSE™ button, with the GPIO state tracking the button state (without debounce).
- This section will only cover the behavior when the touch controller is in the deep sleep state.

- Testing of the GPIOs must be performed manually, the TTHE does not provide this function.

CAPSENSE™ buttons are scanned independently of the main touchscreen area. Buttons can be designed for self-cap or mutual-cap; note that hybrid (both self and mutual) scanning is not available for a wake-up button.

10.3.1 Wake-up button basic setup

Enable the wake-up button with **Wake up Button: WU_BTN_ENABLE**. This button will be the same as button 0 in the project (if no buttons are defined, then the wake-up button will not function). Select the desire scan mode in **Wake up Button: WU_BTN_SCAN_MODE**. Note that the scan mode must align with the button scan mode (**TSS: SCANNING_MODE_BUTTON**); if the button scan mode is hybrid, then there is no restriction on the wake-up button scan mode. The number of TX pulses (**Wake up Button: WU_BTN_TX_PULSES**) should be selected depending on thepower and SNR required for this project.

The hardware design will dictate the wake-up pin to use (**Wake up Button: WU_BTN_HOST_WAKEUP_PIN**), the drive mode (**Wake up Button: WU_BTN_WAKEUP_PIN_DM**), and the polarity (**Wake up Button: WU_BTN_WAKEUP_PIN_ACT_LEVEL**). Note that only P1[0] is currently supported for the wake-up button. This pin must be tested manually (using an oscilloscope or DMM), the TTHE does not provide this function.

10.3.2 Deep sleep state wake-up button

The process is illustrated in the following flow chart:

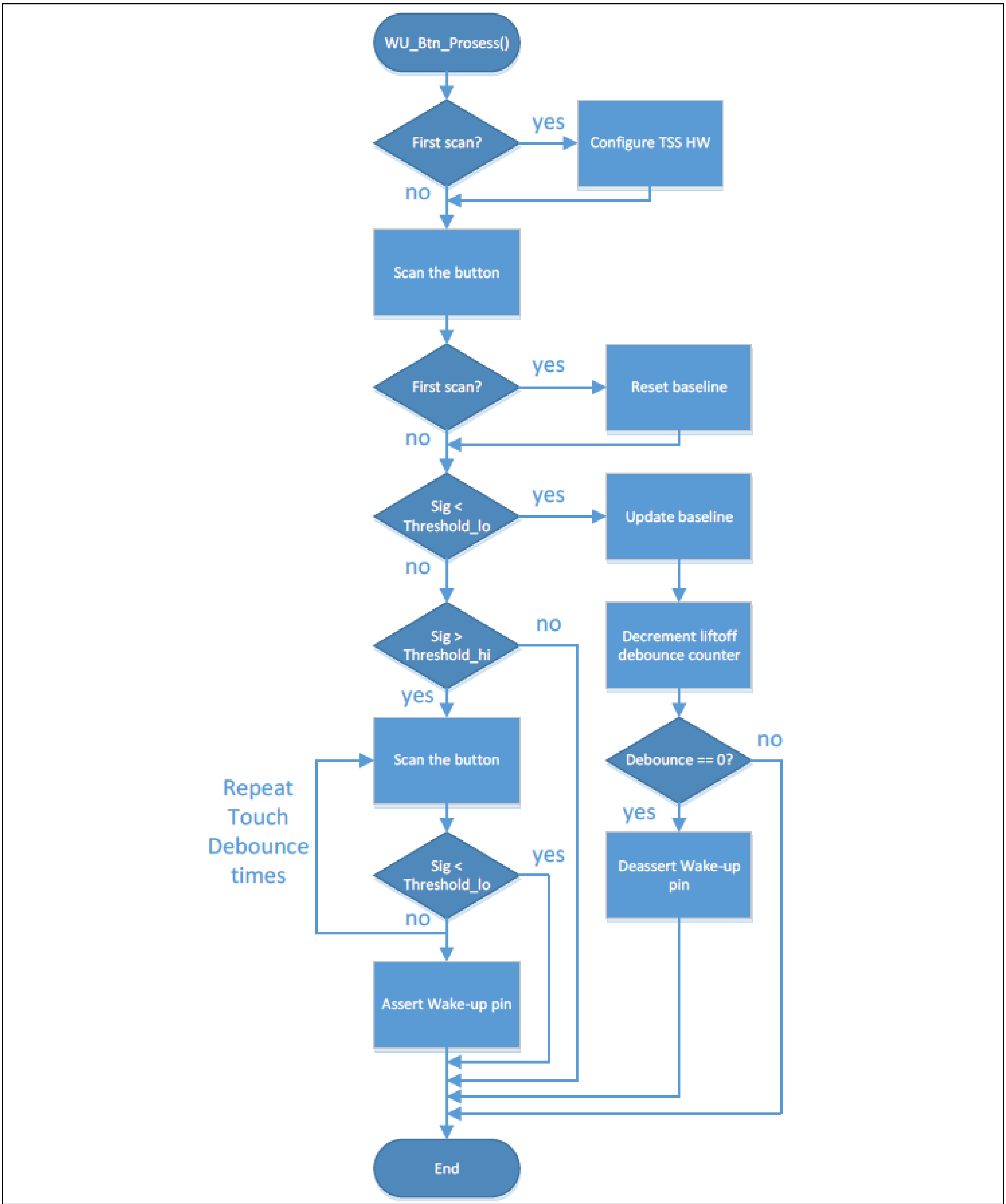


Figure 119 Wake-up button deep sleep flow chart

Specify the wake-up button scan interval (**Wake up Button: WU_BTN_SCAN_INTERVAL**) depending on the required power response time, and baseline tracking. The baselines can only be updated after a button scan, so in designs where frequent temperature changes are expected, the scan rate should be high enough for the baselines to track.

The touch thresholds are the same as for normal CAPSENSE™ buttons. The TTHE parameters used depend on the button configuration, as shown in Figure 120.

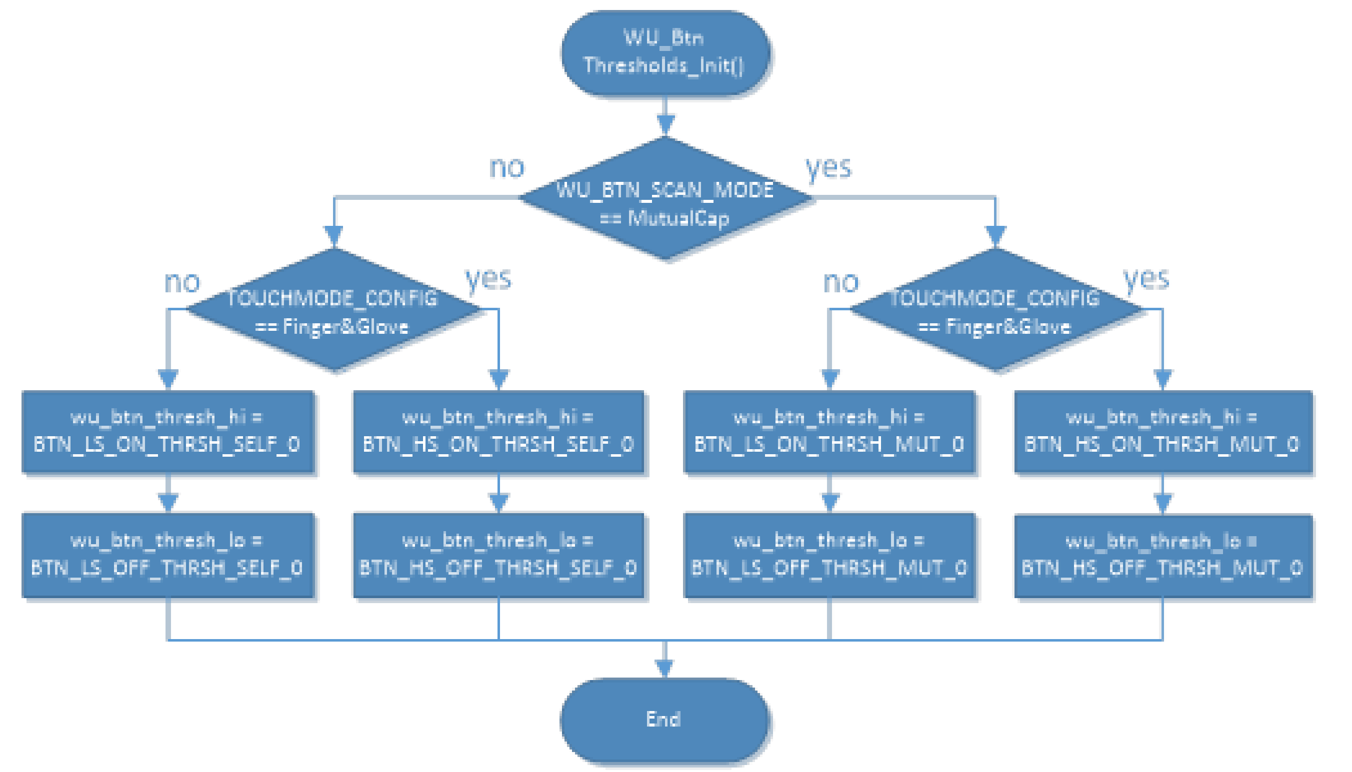


Figure 120 Wake-up button touch thresholds

Set the required debounce times for touchdown (**Wake up Button: WU_BTN_TD_DEBOUNCE**) and liftoff (**Wake up Button: WU_BTN_LIFTOFF_DEBOUNCE**). Both debounce parameters are the number of consecutive occurrences of the event; therefore, a debounce of 1 results in no debounce. During the touchdown debounce, the button is repeatedly scanned with no additional delay. The time taken (BUTTONSCAN_TIME) is dependent on the TX setup, but is normally around 500 µs. During the liftoff debounce (assuming the touch controller is still in the deep sleep state), the button scan rate remains at the wake-up button scan interval (**Wake up Button: WU_BTN_SCAN_INTERVAL**).

$$\text{Min } T_{RESP\_TOUCHDOWN} = \text{Wake-up Button: WU\_BTN\_TD\_DEBOUNCE} * \text{BUTTON}_{SCAN\_TIME}$$

Equation 66

$$\text{Max } T_{RESP\_TD} = \text{Wake-up Button: WU\_BTN\_SCAN\_INTERVAL} + \text{Min } T_{RESP\_TOUCHDOWN}$$

Equation 67

$$\text{Min } T_{RESP\_LIFTOFF} = \text{Max } T_{RESP\_LIFTOFF} - \text{Wake-up Button: WU\_BTN\_SCAN\_INTERVAL}$$

Equation 68

$$\text{Max } T_{RESP\_LIFTOFF} = \text{Wake-up Button: WU\_BTN\_LIFTOFF\_DEBOUNCE} * \text{WU\_BTN\_SCAN\_INTERVAL}$$

Equation 69

The following figure illustrates the touchdown and liftoff debounce, with debounce values of 3 (**Wake up Button: WU_BTN_TD_DEBOUNCE**) and 2 (**Wake up Button: WU_BTN_LIFTOFF_DEBOUNCE**):

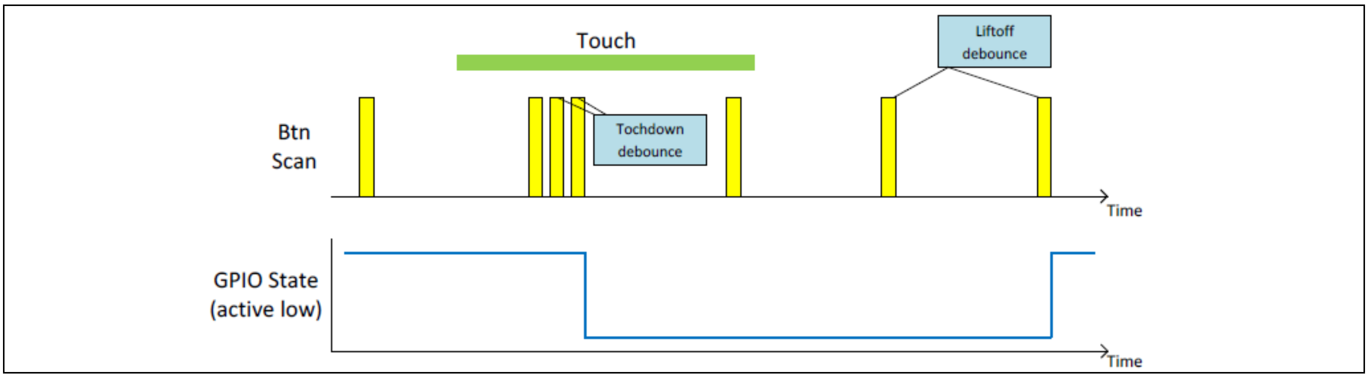


Figure 121 Wake-up button debounce

10.3.3 Wake-up button reset signal

A reset signal is available if the wake-up button is held for a configurable period. This is useful for resetting devices without the need for bringing them into the active state. This pin must be tested manually (using an oscilloscope or DMM), the TTHE does not provide this function.

The reset signal pin (**Wake up Button: HOST_RESET_PIN**) can only be assigned to P1[1]. However, the full complement of drive-modes (**Wake up Button: HOST_RESET_PIN_DM**) and polarities (**Wake up Button: HOST_RESET_PIN_LEVEL**) are available. The period the wake-up button must be activated for the reset signal to be asserted is configurable (**Wake up Button: HOST_RESET_PIN_PULSE_DELAY**) in units of seconds. The pulse width of the reset signal is set at 1 ms.

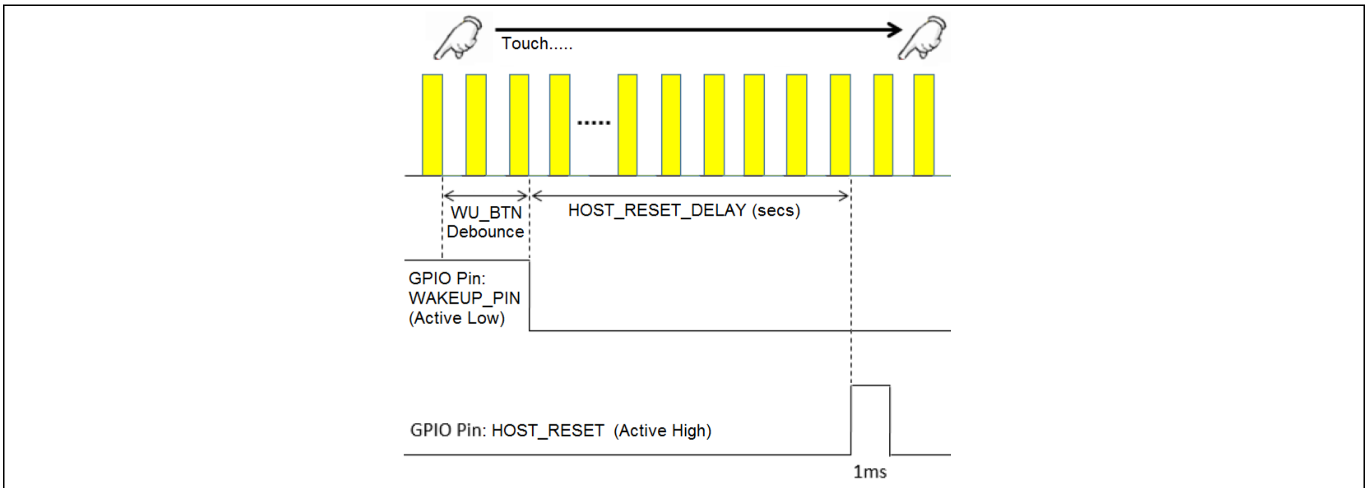


Figure 122 Wake-up button reset signal

10.3.4 Estimated power consumption

The power consumption of the wake-up button is linearly correlated to the following configuration parameters: **TSS: TX_PERIOD_BTN_xx** (self or mutual), **Wakeup: WAKEUP_SCAN_TX_PULSES**, and **Wakeup: WAKEUP_SCAN_INTERVAL**.

For example, with the button configured with a TX period of 102 kHz (235 clocks), 20 TX pulses, and a single 3 V power rail, the following power consumption was measured.

Table 64 Wake-up button power estimates

VDDA Mode		Pump Mode				Bypass Mode			
Meaurement line		VDDA		VDDD		VDDA		VDDD	
Measurement Mode		Self,uA	Mutual,uA	Self,uA	Mutual,uA	Self,uA	Mutual,uA	Self,uA	Mutual, uA
WakeUp Scan Interval	**100ms**	22.6	21.6	63	44.2	5	9.7	38.7	39.5
WakeUp Scan Interval	**150ms**	17.5	15.2	45.7	34.5	4.1	6.6	26.8	26.7
WakeUp Scan Interval	**200ms**	11.7	11.9	37	27.9	2.7	4.9	21.6	20.5
WakeUp Scan Interval	**250ms**	10.3	10.4	29.5	20.3	2.2	4.1	17.9	17
WakeUp Scan Interval	**300ms**	9.8	8.9	18.5	19.5	1.9	3.4	15.7	14.9
WakeUp Scan Interval	**350ms**	7.2	8.15	16.1	18.1	1.6	3	13.8	13.2

10.3.5 Parameters

Table 65 Glove button parameters

Configurable Parameter	Description	Selection
Wake-up Button: WU_BTN_ENABLE	Enable / disable the wake-up button.	Enabled / Disabled (default = Disabled)
Wake-up Button: WU_BTN_SCAN_MODE	Selects self-cap or mutual-cap scan mode for the wake-up button.	Self capacitance / Mutual capacitance (default = Self capacitance)
Wake-up Button: WU_BTN_TX_PULSES	Number of TX Pulses for the wake-up button scan.	1 – 255 (default 32)
Wake-up Button: WU_BTN_HOST_WAKEUP_PIN	GPIO pin to wake-up the host.	Disabled / P1[0] (default = Disabled)
Wake-up Button: WU_BTN_WAKEUP_PIN_DM	Drive mode of the host wakeup pin.	Strong drive / Open drain, drives low / Open drain, drives high / Resistive pull up / Resistive pull down (default = Open drain, drives low)
Wake-up Button: WU_BTN_WAKEUP_PIN_ACT_LEVEL	Active level of the host wakeup pin.	Active High / Active Low (default = Active Low)

Configurable Parameter	Description	Selection
Wake-up Button: WU_BTN_TD_DEBOUNCE	Number of consecutive scans for which Button signal must be above TOUCH_THRESHOLD_LO prior to being reported.	0 – 255 (default 3)
Wake-up Button: WU_BTN_LIFTOFF_DEBOUNCE	Number of consecutive refresh cycles for which Button signal must be below TOUCH_THRESHOLD_LO prior to report liftoff.	0 – 255 (default 2)
Wake-up Button: HOST_RESET_PIN	GPIO pin assigned the wake-up button reset function to reset the host if the wake-up button is held longer than a defined period.	Disabled / P1[1] (default = Disabled)
Wake-up Button: HOST_RESET_PIN_DM	Drive mode of the host wakeup button reset function.	Strong drive / Open drain, drives low / Open drain, drives high / Resistive pull up / Resistive pull down (default = Strong drive)
Wake-up Button: HOST_RESET_PIN_ACT_LEVEL	Active level of the host wakeup button reset function.	Active High / Active Low (default = Active High)
Wake-up Button: HOST_RESET_PIN_PULSE_DELAY	Period (in seconds) for which wake-up button must be active prior to the 1 ms reset pulse being triggered.	1 – 255 (default 15)
Wake-up Button: WU_BTN_SCAN_INTERVAL	Refresh interval of wake-up button (ms).	10 – 1000 (default 200)

11 Gesture tuning

11.1 Introduction

Some versions of the touch controller can detect and report gestures. When available and enabled, the touch controller monitors the touch data, applies algorithms to detect gestures, and reports them to the host.

Gestures are enabled by setting the parameter **Gesture: GESTURE_ENABLED** to “Enabled Standard” or “Enabled Extended”. When standard gestures are enabled, they can be monitored in the TTHE software in the Data Monitor window (View -> Tool Windows -> Data Monitor). Make sure gesture reporting is enabled in this window by clicking the Configure icon  and verifying that the “Gesture” box is selected. Click the green start arrow to begin monitoring gestures.

A history of gestures can also be seen by using the Event History tool window (View -> Tool Windows -> Event History). Click the green start arrow to enable monitoring.

The controller scans the panel at a regular rate, determined by **Device Setup: ACT_INTRVL0**. By monitoring activity over time, the controller can detect moving fingers and report them as gestures. The supported gestures are summarized in Table 66.

Gestures are reported in the operating mode registers. Standard gestures are reported using a Gesture ID value and a gesture count that tells how many times the gesture has been detected.

Extended gestures report not only the gesture itself, but also extended information about the gesture. The extended information is sent in the operating mode registers associated with the gesture. Details of the data for each extended gesture are described in the Technical Reference Manual, document number 001-99382.

Extended gestures cannot be monitored in TTHE. Instead, they must be monitored by viewing the I2C traffic using an analyzer or the Bridge Control Panel tool. Refer to the Technical Reference Manual for more information.

Table 66 Gesture summary

Gesture	Type	Gesture ID	Description
NO_GESTURE	Standard	0x00	No gesture detected.
ST_NORTH	Standard	0x10	Single-finger pan gesture in the north direction.
ST_NORTH_EAST	Standard	0x12	Single-finger pan gesture in the northeast direction.
ST_EAST	Standard	0x14	Single-finger pan gesture in the east direction.
ST_SOUTH_EAST	Standard	0x16	Single-finger pan gesture in the southeast direction.
ST_SOUTH	Standard	0x18	Single-finger pan gesture in the south direction.
ST_SOUTH_WEST	Standard	0x1A	Single-finger pan gesture in the southwest direction.
ST_WEST	Standard	0x1C	Single-finger pan gesture in the west direction.
ST_NORTHWEST	Standard	0x1E	Single-finger pan gesture in the northwest direction.
ST_CLICK	Standard	0x20	Single-finger single click (touch down and lift) gesture.
ST_DOUBLECLICK	Standard	0x22	Single-finger double click gesture.
ST_ROTATE_CW	Standard	0x28	Single-finger clockwise rotation gesture.
ST_ROTATE_CCW	Standard	0x29	Single-finger counter-clockwise rotation gesture.
TOUCHDOWN	Standard	0x2F	Single-finger touch down on panel.
MT_NORTH	Standard	0x30	Two-finger pan gesture in the north direction.

Gesture	Type	Gesture ID	Description
MT_NORTH_EAST	Standard	0x32	Two-finger pan gesture in the northeast direction.
MT_EAST	Standard	0x34	Two-finger pan gesture in the east direction.
MT_SOUTH_EAST	Standard	0x36	Two-finger pan gesture in the southeast direction.
MT_SOUTH	Standard	0x38	Two-finger pan gesture in the south direction.
MT_SOUTH_WEST	Standard	0x3A	Two-finger pan gesture in the southwest direction.
MT_WEST	Standard	0x3C	Two-finger pan gesture in the west direction.
MT_NORTHWEST	Standard	0x3E	Two-finger pan gesture in the northwest direction.
MT_CLICK	Standard	0x40	Two-finger single click gesture.
MT_ZOOM_IN	Standard	0x48	Two-finger pan gesture with fingers moving away from each other.
MT_ZOOM_OUT	Standard	0x49	Two-finger pan gesture with fingers moving toward each other.
LIFT_OFF	Standard	0x4F	Single-finger lift off from panel.
DT_ROTATE_CW	Standard	0x2B	Two-finger clockwise rotation gesture.
DT_ROTATE_CCW	Standard	0x2D	Two-finger counter-clockwise rotation gesture.
FLICK_NORTH	Standard	0x50	Single-finger combination gesture of touchdown, pan north, lift off.
FLICK_NORTH_EAST	Standard	0x52	Single-finger combination gesture of touchdown, pan northeast, lift off.
FLICK_EAST	Standard	0x54	Single-finger combination gesture of touchdown, pan east, lift off.
FLICK_SOUTH_EAST	Standard	0x56	Single-finger combination gesture of touchdown, pan southeast, lift off.
FLICK_SOUTH	Standard	0x58	Single-finger combination gesture of touchdown, pan south, lift off.
FLICK_SOUTH_WEST	Standard	0x5A	Single-finger combination gesture of touchdown, pan southwest, lift off.
FLICK_WEST	Standard	0x5C	Single-finger combination gesture of touchdown, pan west, lift off.
FLICK_NORTHWEST	Standard	0x5E	Single-finger combination gesture of touchdown, pan northwest, lift off.
RELEASE	Extended	0x80	Same as LIFT_OFF, except it can be reported in the case of transition from two fingers to one finger. Extended data includes X and Y position of last single-finger lift off.
CLICK	Extended	0x81	Same as ST_CLICK and ST_DOUBLECLICK, except triple click also triggers this gesture. Extended data includes X and Y position of the click and a field defining whether click was single, double, or triple.
PRESS	Extended	0x82	Same as TOUCHDOWN, but it is also reported when there is a transition from one finger to two. Extended data gives X and Y position of the press.

Gesture	Type	Gesture ID	Description
DRAG	Extended	0x84	Same as a single-finger pan gesture. Extended data gives X and Y location of the finger and flags indicating direction.
FLICK	Extended	0x85	Same as standard flick gesture. Extended data gives X and Y location of finger and flags indicating direction.
ROTATE	Extended	0x86	Same as two-finger rotate gesture. Extended data gives X and Y location of center point of rotation and direction and size of angle motion.
RAW2	Extended	0x87	This gesture is reported when there are two touches, and all two-finger gestures are disabled in the **Gesture: GEST_GROUP_MASK** parameter. Extended data includes center point between two fingers, distance between fingers, and angle of line formed by two fingers.
PRESS2	Extended	0x88	This gesture is reported when there are two fingers on the panel. Extended data gives location of center point and distance between fingers.
DRAG2	Extended	0x89	Same as two-finger pan gestures. Extended data gives location of center point and distance between fingers.
FLICK2	Extended	0x8A	Same as extended flick gesture, but with two fingers. Extended data gives location and distance between fingers.
ZOOM	Extended	0x8B	Reported by zoom motion. Extended data includes position of center point between fingers, distance between fingers, and change in distance between fingers.
RIGHTCLICK	Extended	0x8C	Triggered by click gesture of second finger on panel. Extended data includes location of center point and distance between fingers.
RAW2_PLUS	Extended	0x8D	This gesture is reported when there are more than two touches and all two-finger gestures are disabled in the **Gesture: GEST_GROUP_MASK** parameter. Extended data includes the center point between fingers, distance between fingers, and angle formed by fingers.

11.2 Gesture tuning

For a gesture to be valid, the motion or event must be strong enough to ensure that accidental gestures are not detected. For example, a very small motion should not be reported as a PAN gesture; only motion above a certain threshold should be. Therefore, tuning parameters are available to adjust the sensitivity of each gesture. Debounce is also provided for appropriate gestures; the gesture must be repeated a certain number of times before the gesture is deemed to be valid.

11.2.1 Enabling/disabling individual gestures

Some individual gestures, or groups of gestures, can be enabled or disabled using the **Gesture: GEST_GROUP_MASK** parameters. See Table 67 for details.

11.2.2 Pan gestures

A pan gesture is a motion of one or two fingers in a given direction. To be a valid pan gesture, both one- and two-finger motion must travel a minimum distance in X or Y (X AND Y for diagonal pans). These distances are configurable via the **Gesture: GEST_PAN_ACTIVE_DISTANCE_X** and **Gesture: GEST_PAN_ACTIVE_DISTANCE_Y** parameters. The parameters are specified in resolution units.

The pan gestures also have a debounce – there must be a given number of consecutive detected pan gestures before the gesture is reported. For single-finger pan gestures, the number is determined by the parameter **Gesture: GEST_DEBOUNCE_SINGLETOUCH_PAN_COUNT**. For two-finger gestures, it is determined by **Gesture: DEBOUNCE_MULTITOUCH**.

Additionally, for multitouch pan gestures, the parameter **Gesture: GEST_SETTLING_TIMEOUT_MSEC** sets the minimum duration for waiting before reporting two-finger pan gestures. This parameter prevents inadvertent gesture reporting for short-duration motion of two fingers on a panel.

11.2.3 Single-finger rotation gesture

A single-finger rotation is detected when a pan gesture changes from one direction to another. The parameter **Gesture: GEST_ROTATE_DEBOUNCE_LIMIT** defines the number of sequential pan gestures in a single direction, after changing from a different direction, that must be detected before the rotation gesture is cancelled.

11.2.4 Flick gestures

A flick gesture is reported when consecutive touchdown, pan, and liftoff gestures are detected. Note that if pan gestures are enabled, they will also be reported.

To be a valid flick, there are two requirements: the flick must meet minimum distance requirements for X or Y (X AND Y for diagonal flicks), and the entire gesture must be completed within a maximum time. The distances are defined by the parameters **Gesture: FEST_FLICK_ACTIVE_DISTANCE_X** and **Gesture: FLICK_ACTIVE_DISTANCE_Y**. The maximum time is defined by **Gesture: GEST_FLICK_SAMPLE_TIME**. Gestures that are too short in distance, or take too long in time, are not reported as flicks.

11.2.5 Single-touch click gestures

An ST_CLICK gesture is a consecutive touchdown and liftoff of a single finger. To be valid, it must meet constraints for distance and time.

- The distance from the location of the touchdown to the location of the liftoff must be less than a minimum X and Y distance. The minimum distances are determined by the **Gesture: GEST_CLICK_X_RADIUS** and **Gesture: GEST_CLICK_Y_RADIUS** parameters.
- The time between the touchdown and liftoff must be less than **Gesture: GEST_ST_MAX_CLICK_TIMEOUT_MSEC**.

An ST_DOUBLECLICK gesture is reported when two ST_CLICK gestures are detected, subject to additional constraints:

The two clicks must be close together. The distance from the first click to the second click must not exceed **Gesture: GEST_ST_MAX_DOUBLE_CLICK_RADIUS**.

The time between the two clicks must be more than **Gesture: GEST_ST_MIN_CLICK_TIMEOUT_MSEC** and less than **Gesture: GEST_ST_MAX_CLICK_TIMEOUT_MSEC**.

11.2.6 Zoom gesture

MT ZOOM gestures are defined by two fingers moving toward or away from each other. To be valid, the distance change in either direction (toward or away) must exceed **Gesture: GEST_ZOOM_ACTIVE_DISTANCE**.

Additionally, the parameter **Gesture: GEST_SETTLING_TIMEOUT_MSEC** sets the minimum duration for waiting before reporting zoom gestures. This parameter prevents inadvertent gesture reporting for short-term motion of two fingers on a panel.

11.2.7 Two-finger click gestures

An MT_CLICK gesture is defined by two fingers touching down and then lifting off. To be valid, both fingers must be on the panel for a minimum time defined by **Gesture: GEST_MT_MIN_CLICK_TIMEOUT_MSEC** and a maximum time defined by **Gesture: GEST_MT_MAX_CLICK_TIMEOUT_MSEC**.

11.2.8 Two-finger rotate gestures

Two-finger rotation must exceed an angular threshold before being considered valid. The minimum angle is defined by the parameter **Gesture: GEST_MULTITOUCH_ROTATION_THRESHOLD**. It is specified in units of 1/100 of a radian. They must also continue for consecutive scans, with the minimum number defined by the **Gesture: GEST_DEBOUNCE_MULTITOUCH** parameter.

11.2.9 Right-click gesture

A RIGHTCLICK gesture is triggered by the touchdown and liftoff of a second finger, while a first finger is on the panel. It is an extended gesture. To be valid, it must meet the distance of the ST_CLICK gesture, and have a duration that falls between **Gesture: GEST_RIGHTCLICK_MIN_TIMEOUT_MSEC** and **Gesture: RIGHTCLICK_MAX_TIMEOUT_MSEC**.

11.3 Parameters

Table 67 Gesture parameters

Configurable Parameter	Description	Selection
Gesture: GEST_PAN_ACTIVE_DISTANCE_X	Minimum active step distance (in resolution units) that must be exceeded in X direction before motion is considered as Pan Gesture.	1 – 255 (default = 10)
Gesture: GEST_PAN_ACTIVE_DISTANCE_Y	Minimum active step distance (in resolution units) that must be exceeded in Y direction before motion is considered as Pan Gesture.	1 – 255 (default = 10)
Gesture: GEST_ZOOM_ACTIVE_DISTANCE	Minimum active step distance (in resolution units) that must be exceeded before motion is considered as Zoom Gesture.	1 – 255 (default = 10)

Configurable Parameter	Description	Selection
Gesture: GEST_DEBOUNCE_MULTITOUCH	This parameter sets the number of sequential multitouch gestures that must be observed before the gesture is deemed valid.	1 – 255 (default = 2)
Gesture: GEST_FLICK_ACTIVE_DISTANCE_X	Minimum active step distance (in resolution units) that must be exceeded in X direction before motion is considered as Flick Gesture.	1 – 255 (default = 35)
Gesture: GEST_FLICK_ACTIVE_DISTANCE_Y	Minimum active step distance (in resolution units) that must be exceeded in Y direction before motion is considered as Flick Gesture.	1 – 255 (default = 35)
Gesture: GEST_FLICK_SAMPLE_TIME	Set the maximum time window that will be searched for the flick (in milliseconds).	1 – 255 (default = 80)
Gesture: GEST_DEBOUNCE_SINGLETOUCH_PAN_COUNT	Set the number of similar sequential pan gestures that should be performed before the pan motion is considered valid. This parameter is for the single touch pan motions.	1 – 255 (default = 2)
Gesture: GEST_MULTITOUCH_ROTATION_THRESHOLD	The multi-touch rotation threshold (in 1/100th radian) that must be exceeded during one scanning cycle before motion is considered as multi-touch rotation gesture.	1 – 255 (default = 4)
Gesture: GEST_ROTATE_DEBOUNCE_LIMIT	This parameter sets the number of sequential pan gestures in a consistent direction that have to be observed before the rotate gesture is deemed invalid.	1 – 255 (default = 20)
Gesture: GEST_ST_MAX_DOUBLE_CLICK_RADIUS	Sets the maximum pixel radius that the second click in a double click sequence can extend. If the second click occurs outside this radius the double click sequence is discarded.	1 – 255 (default = 50)
Gesture: GEST_CLICK_X_RADIUS	This parameter sets the maximum X-axis displacement for a ST Click.	1 – 255 (default = 30)
Gesture: GEST_CLICK_Y_RADIUS	This parameter sets the maximum Y-axis displacement for a ST Click.	1 – 255 (default = 30)
Gesture: GEST_MT_MAX_CLICK_TIMEOUT_MSEC	This parameter sets the maximum time that two fingers can be placed on the panel before being	1 – 65535 (default = 200)

Configurable Parameter	Description	Selection
	disqualified as a multitouch click event. The time is measured in milliseconds.	
Gesture: GEST_MT_MIN_CLICK_TIMEOUT_MSEC	This sets the minimum duration (in milliseconds) that two fingers need to be on the panel before a multitouch click event is registered.	1 – 65535 (default = 20)
Gesture: GEST_ST_MAX_CLICK_TIMEOUT_MSEC	Sets the maximum duration (in milliseconds) that a finger has to be on the panel before a single click event is considered to be valid.	1 – 65535 (default = 200)
Gesture: GEST_ST_MIN_CLICK_TIMEOUT_MSEC	This parameter sets the minimum duration that a click can stay on the panel to qualify as a single touch click. This can be used by applications to set how deliberately a single click operation must be performed. This helps filter out noisy events or very rapid clicks which are usually performed inadvertently. This parameter should be set lower than the single touch max click timeout parameter. The time is measured in milliseconds.	1 – 65535 (default = 20)
Gesture: GEST_ST_MAX_DOUBLECLICK_TIMEOUT_MSEC	The maximum allowable time between two sequential clicks such that a double click gesture will be reported. A double click gesture is a sequence of two single click gestures where the second single click gesture occurs within the ST Max DblClick Timeout of the first. When a ST Double-click gesture occurs a single click gesture is reported when the first touch lifts off the panel the double click gesture is reported if the second click is detected within the ST Max DblClick Timeout. Note that the two single click actions must also meet the single click timing requirements. The time is measured in milliseconds.	1 – 65535 (default = 400)
Gesture: GEST_ST_MIN_DOUBLECLICK_TIMEOUT_MSEC	This parameter is the minimum allowable time between two sequential clicks such that a double	1 – 65535 (default = 100)

Configurable Parameter	Description	Selection
	click gesture will be reported. A double click gesture is a sequence of two single click gestures where the second single click gesture occurs within the ST Max DblClick Timeout of the first. When a ST Double -lick gesture occurs a single click gesture is reported when the first touch lifts off the panel the double click gesture is reported if the second click is detected within the ST Max DblClick Timeout. Note that the two single click actions must also meet the single click timing requirements. The time is measured in milliseconds.	
Gesture: GEST_RIGHTCLICK_MIN_TIMEOUT_MSEC	This parameter sets the minimum duration (in milliseconds) that a right click can stay on the panel to qualify as a right click touch.	1 – 65535 (default = 20)
Gesture: GEST_RIGHTCLICK_MAX_TIMEOUT_MSEC	Sets the maximum duration (in milliseconds) that a right click can stay on the panel to qualify as a right click touch.	1 – 65535 (default = 200)
Gesture: GEST_SETTLING_TIMEOUT_MSEC	This parameter sets the minimum duration of how long to wait prior to decoding MT pan and zoom. The time is measured in milliseconds.	1 – 65535 (default = 40)
Gesture: GEST_GROUP_MASK: No Standard/RELEASE	Enables or disables the extended RELEASE gesture. When disabled, a liftoff gesture is reported when the first of two fingers is lifted from the panel. When enabled, the liftoff gesture is not reported. No standard gestures are affected.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: No Standard/CLICK	Enables or disables the extended CLICK gesture. No standard gestures are affected.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: No Standard/PRESS	Enables or disables the extended PRESS gesture. No standard gestures are affected.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: No Standard/DRAG	Enables or disables the extended DRAG gesture. No standard gestures are affected.	Enabled / Disabled

Configurable Parameter	Description	Selection
		(default = Enabled)
Gesture: GEST_GROUP_MASK: Single-Touch Pan/FLICK	Enables or disables the standard PAN and FLICK gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Multi-Touch Pan/ROTATE	Enables or disables the standard two-finger PAN and extended ROTATE gesture.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Single-Touch Click/RAW2	Enables or disables the standard single-touch CLICK and extended RAW2 gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Single-Touch Double Click/PRESS2	Enables or disables the standard ST_DOUBLECLICK and extended PRESS2 gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Multi-Touch Click/DRAG2	Enables or disables the standard MT_CLICK and extended DRAG2 gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Touchdown/FLICK2	Enables or disables the standard TOUCHDOWN and extended FLICK2 gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Single-Touch Rotate/ZOOM	Enables or disables the standard ST_ROTATE and extended ZOOM gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Multi-Touch Rotate/RIGHTCLICK	Enables or disables the standard MT_ROTATE and extended RIGHTCLICK gestures.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Zoom/RAW2 Plus	Enables or disables the standard ZOOM and the extended RAW2 gestures. All two-finger gestures must be disabled for RAW2 to be enabled.	Enabled / Disabled (default = Enabled)
Gesture: GEST_GROUP_MASK: Lift Off/No Extended	Enables or disables the standard LIFT_OFF gesture. No extended gestures are affected.	Enabled / Disabled (default = Enabled)

Configurable Parameter	Description	Selection
Gesture: GEST_GROUP_MASK: Single-Flick/No Extended	Enables or disables the standard FLICK gestures. No extended gestures are affected.	Enabled / Disabled (default = Enabled)
Gesture: GESTURE_ENABLED	Disables gesture reporting, enables or disables standard or extended gestures.	Gestures Disabled / Enabled Standard / Enabled Extended (default = Enabled Standard)

12 Panel ageing

Note that testing for panel ageing is currently under development. In future base firmware releases, the test accuracy will be improved, and simplified, by providing firmware assistance.

The ageing of a touchscreen's stack-up is a relevant problem for the automotive industry, as the life cycle of touchscreen device can be more than 20 years. This section introduces a method to simulation ageing so tuning parameters can be set to allow for a long product life.

The characteristics of the touch panel stack-up changes significantly with age. A key characteristic is the dielectric property which causes capacitance (Cm/Cp) drift, and as a result, significant drift in the raw counts. The main reason for the change is the ageing of materials used in the stack-up (overlay, glue, etc.). Other reasons are also possible, for example: delamination of the stack-up, water and air trapped in the stack-up, and gassing of the glue.

The expected capacitance drift due to ageing could be as high as 0.5 pf for mutual-cap and 5 pf for self-cap. Similarly, the expected capacitance drift due to temperature could be as high as 0.5 pf for mutual-cap and 5pf for self-cap. As these drifts can be in the same direction, the panel should be tuned to allow for capacitance changes of up to 1 pf for mutual-cap and 10 pf for self-cap. Note that these values are highly dependent on the stack-up.

12.1 Ageing test

12.1.1 Before ageing test

Before the ageing test is performed, the panel should be fully tuned for the performance (accuracy, linearity, response, etc.) and environment (temperature, emissions, ESD, etc.) required.

Capture the original Cm and Cp data for the panel. This can be performed using the TTHE. If not already enabled, enable the window by View > Tool Windows > Cm & Cp Data. With the panel running, refresh the Cm/Cp data using the "Cm & Cp Data" window's "Read" button. Then save it using the "Save" button.

Capture the original raw data for the panel. This can be performed using the TTHE. Set the "Display Mode" to "Heat Map". In the "Touch Display Settings" window, set the "Data Type" to "Raw Post Filter", the "Scan Type" to "Self", and the "Sensor Value Type" to "Average". Above the heat-map, click the "Collect" button and wait while it is "Collecting". In the project folder, save the log file (usually called "log.csv") with a good name (for example "yymmdd Ageing Test Original Raw Data.csv").

Capture the touch diff counts for the panel. This can be performed using the same method as the raw data above; except setting the "Data Type" to "Diff Counts", setting the "Sensor Value Type" to "Max Hold", and moving a touch across the entire panel before clicking the "Collect" button. Again, save the log file with a good name (for example "yymmdd Ageing Test Original Diff Data.csv").

12.1.2 Test simulation for estimation

Infineon provides a method of simulating the maximum drift due to panel ageing and temperature drift. This method can be used to estimate the maximum drift for the project. However, a physical ageing test should be performed when the final stack-up is available. The results of the physical test should be recorded for reference by future projects with a similar stack-up, so the simulation data can be more reliably viewed.

This simulation method is currently manual.

12.1.3 Test execution

Age the panel according to the customer or material specifications. This is usually achieved by baking the module for a specific time and temperature; however, the specific process can be supplier, customer, and project dependent.

IMPORTANT: After ageing, do not recalibrate the module.

Re-capture all the information from the previous section and verify performance is as expected:

- Panel performance:
  - Test full panel detection of the correct finger types when testing with the thickest glove, smallest finger, largest finger, and smallest large object.
- Cm & Cp Data:
  - Ensure the drift is not greater than expected (usually 1pf and 10pf as described in the introduction to this chapter).
- Raw data:
  - Ensure raw data is not saturating (less than +/- 2^{15}).
- Diff counts:
  - Ensure diff counts are similar to those recorded before ageing.

12.2 Tuning for ageing

While there is no specific tuning for ageing. The ageing tests above show the expected worst-case touch performance at the extreme of product life.

If touch works well after the ageing test, then no further action is needed. If touch does not work well, then the panel should be re-tuned. Examples of parameters that can be helpful in this re-tuning are the attenuator override (e.g. **Calibration: ATTENUATOR_MC**) and the IDAC balancing target (e.g. **Calibration: BAL_TARGET_MC**).

13 Manufacturing test parameters

At startup, the touch controller can perform shorts and/or opens POSTs (power-on self-tests). Often these tests are also part of the manufacturing process. The POSTs that are performed are controlled by the **MFG: POST_SHORT_OPEN_CTRL** parameter. The POSTs to select are project dependent.

The results of the POST are stored in System Information registers, viewable in TTHE by selecting View, then System Information. Refer to the technical reference manual, Infineon document 001-99382, for details of the registers.

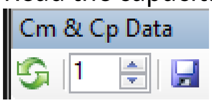
If the **TSS: VDDA_MODE** parameter is set to "Bypass Mode", then the **MFG: VDDA_LEVEL** must be set to the voltage supplied to the touch controller's VDDA pin.

13.1 Opens test: minimum capacitance

There are five parameters that define the minimum capacitance allowable before triggering an opens test failure:

- **MFG: MUTUAL_MIN_CAP**
- **MFG: SELF_TX_MIN_CAP**
- **MFG: SELF_RX_MIN_CAP**
- **MFG: BUTTON_MUTUAL_MIN_CAP**
- **MFG: BUTTON_SELF_MIN_CAP**

The same tuning procedure should be used for each scanning method used:

1. Connect a "known good" panel, which we can rework with an open sensor (see step 5).
2. Open the TTHE's capacitance window (View -> Tool Windows -> Cm & Cp Data).
3. Read the capacitance data by clicking the Read icon, and save it to a CSV file using the Save… icon:


4. In the CSV file, find the minimum value of Cm and/or Cp for the whole panel.
5. Using the same panel, make a HW open sensor at the end of both a single TX and a single RX:
- If single-routing is used: Create the opens at the end of the longest sensors.
- If double-routing is used: Create the opens at the end of the longest traces (next to the first sensor).
6. Write down the values of Cm and/or Cp for open sensors.
7. Set the minimum mutual-cap and/or self-cap values between the values calculated in step 4 (minimum good capacitance) and step 6 (known open-sensor capacitance); note that the exact value to set depends on:
- Capacitive variance of the particular project due to the manufacturing process, ageing process, and so on.
- Required strictness of the opens test.

13.2 Opens test: capacitance dispersion

There are four parameters that define the maximum capacitance dispersion before triggering an opens test failure:

- **MFG: CM_DISPERSION_PANEL**
- **MFG: CP_DISPERSION_PANEL**
- **MFG: CM_DISPERSION_BTN**
- **MFG: CP_DISPERSION_BTN**

Capacitance dispersion is the variance of the capacitance across all sensors. The parameters are the allowable percentage variance, and is applied to all four capacitance types (combination of mutual-cap/self-cap and touch-panel/button).

If the range of capacitance, for a particular scan type, is greater than this percentage then an opens test failure is triggered. The "Capacitance Dispersion" of a panel is calculated by the following formula:

$$\text{Capacitance Dispersion} = \left(\frac{(C_{MAX} - C_{MIN})}{C_{AVG}}\right) * 100\%$$

Equation 70

Tuning is dependent on both the capacitive variance of the particular project (due to the manufacturing process, ageing process, etc.) and the required strictness of the opens test.

It is strongly recommended to set the limits tight during manufacturing testing, so any minor issues, or process variation, can be detected. However, the TTHE settings should be relatively loose, as only a catastrophic failure should be reported.

13.3 Shorts test tuning

The shorts test is controlled by the **MFG: ILEAK_MAX** parameter. This parameter defines the maximum allowable current leakage from a sensor before a shorts test error is triggered. The ILEAK data for each sensor is output in the manufacturing test log file for the shorts test (added in TTE/MTK version 1.8.5).

The value to set the **MFG: ILEAK_MAX** parameter is dependent on both the capacitive variance of the particular project (due to the manufacturing process, ageing process, etc.) and the required strictness of the shorts test.

On CYAT8168 (TSG6XL) devices, there is an additional parameter **MFG: ILEAK_DISCARD_PULSES** to discard the initial (more unreliable) pulses. It is recommended to leave this value at the default.

Note that the measured leakage can be affected by display noise, and by the protective covers used in the manufacturing line. It may be necessary to have multiple configurations. Each manufacturing stage should be examined to determine if a custom ILEAK setting is required. ILEAK should also be tested with best-case and worst-case display noise images, and the tuning parameter set accordingly.

It is strongly recommended to set the ILEAK limit tight during manufacturing testing, so any minor issues, or process variation, can be detected. However, the TTHE setting should be relatively loose, as only a catastrophic failure should be reported.

13.4 Parameters

Table 68 Manufacturing test parameters

Configurable Parameter	Description	Selection
MFG: MUTUAL_MIN_CAP	Minimal Allowed Capacitance of Mutual-cap Sensors in units of fF. Only used by the opens test.	0 – 10000 (default = 800)
MFG: SELF_TX_MIN_CAP	Minimal Allowed Capacitance of Self-cap TX Sensors in units of fF. Only used by the opens test.	0 – 200000 (default = 8000)
MFG: SELF_RX_MIN_CAP	Minimal Allowed Capacitance of Self-cap RX Sensors in units of fF. Only used by the opens test.	0 – 200000 (default = 8000)
MFG: BUTTON_MUTUAL_MIN_CAP	Minimal Allowed Capacitance of Mutual-cap Buttons in units of fF. Only used by the opens test.	0 – 300000 (default = 5000)
MFG: BUTTON_SELF_MIN_CAP	Minimal Allowed Capacitance of Self-cap Sensors in units of pF. Only used by the opens test.	0 – 500000 (default = 50000)
MFG: POST_SHORT_OPEN_CTRL	Enable/Disable Shorts and Opens Test for POST.	Disabled All / Shorts Only Enabled / Opens Only Enabled / Enabled All (default = Enabled All)
MFG: CM_DISPERSION_PANEL **MFG: CP_DISPERSION_PANEL** **MFG: CM_DISPERSION_BTN** **MFG: CP_DISPERSION_BTN**	Maximum allowed dispersion of capacitance. In percent. Only used by the opens test.	0 – 10000 (default = 300)
MFG: ILEAK_MAX	Maximum allowable (non-short-circuit) sense pin leakage current to/from external sinks/sources (in nA). Possible values are from 0 nA to 24000 nA. Only used by the shorts test.	0 – 24000 (default = 1300)
MFG: ILEAK_DISCARD_PULSES CYAT8168 (TSG6XL) only	Note, CYAT8168 (TSG6XL) only. Only used by the shorts test. The number of initial pulses to ignore before applying the shorts test.	1 – 200 (default = 2)
MFG: VDDA_LEVEL	VDDA power supply voltage in unit of mV. Only used by the Cm test if the power mode is "bypass mode".	2000 – 5500 (default = 2800)

14 Manufacturer and designer user parameters

Some tuning memory is reserved to store bytes at the discretion of the manufacturer and designer. Typical examples could be date code, design location code, manufacturing date, and so on. There are 64 bytes classified as manufacturing values, and 32 classified as design values, although in reality there is no distinction between the two classes.

Table 69 Manufacturing and Design User Parameters

Configurable Parameter	Selection	Configurable Parameter	Selection
MFG Values: MFG_VAL0	0 – 255 (default = 0)	**MFG Values: MFG_VAL32**	0 – 255 (default = 0)
MFG Values: MFG_VAL1		**MFG Values: MFG_VAL33**	
MFG Values: MFG_VAL2		**MFG Values: MFG_VAL34**	
MFG Values: MFG_VAL3		**MFG Values: MFG_VAL35**	
MFG Values: MFG_VAL4		**MFG Values: MFG_VAL36**	
MFG Values: MFG_VAL5		**MFG Values: MFG_VAL37**	
MFG Values: MFG_VAL6		**MFG Values: MFG_VAL38**	
MFG Values: MFG_VAL7		**MFG Values: MFG_VAL39**	
MFG Values: MFG_VAL8		**MFG Values: MFG_VAL40**	
MFG Values: MFG_VAL9		**MFG Values: MFG_VAL41**	
MFG Values: MFG_VAL10		**MFG Values: MFG_VAL42**	
MFG Values: MFG_VAL11		**MFG Values: MFG_VAL43**	
MFG Values: MFG_VAL12		**MFG Values: MFG_VAL44**	
MFG Values: MFG_VAL13		**MFG Values: MFG_VAL45**	
MFG Values: MFG_VAL14		**MFG Values: MFG_VAL46**	
MFG Values: MFG_VAL15		**MFG Values: MFG_VAL47**	
MFG Values: MFG_VAL16		**MFG Values: MFG_VAL48**	
MFG Values: MFG_VAL17		**MFG Values: MFG_VAL49**	
MFG Values: MFG_VAL18		**MFG Values: MFG_VAL50**	
MFG Values: MFG_VAL19		**MFG Values: MFG_VAL51**	
MFG Values: MFG_VAL20		**MFG Values: MFG_VAL52**	
MFG Values: MFG_VAL21		**MFG Values: MFG_VAL53**	
MFG Values: MFG_VAL22		**MFG Values: MFG_VAL54**	
MFG Values: MFG_VAL23		**MFG Values: MFG_VAL55**	
MFG Values: MFG_VAL24		**MFG Values: MFG_VAL56**	
MFG Values: MFG_VAL25		**MFG Values: MFG_VAL57**	
MFG Values: MFG_VAL26		**MFG Values: MFG_VAL58**	
MFG Values: MFG_VAL27		**MFG Values: MFG_VAL59**	
MFG Values: MFG_VAL28		**MFG Values: MFG_VAL60**	
MFG Values: MFG_VAL29		**MFG Values: MFG_VAL61**	
MFG Values: MFG_VAL30		**MFG Values: MFG_VAL62**	
MFG Values: MFG_VAL31		**MFG Values: MFG_VAL63**	

Configurable Parameter	Selection	Configurable Parameter	Selection
User Values: VAL0	0 – 255 (default = 0)	**User Values: VAL16**	0 – 255 (default = 0)
User Values: VAL1		**User Values: VAL17**	
User Values: VAL2		**User Values: VAL18**	
User Values: VAL3		**User Values: VAL19**	
User Values: VAL4		**User Values: VAL20**	
User Values: VAL5		**User Values: VAL21**	
User Values: VAL6		**User Values: VAL22**	
User Values: VAL7		**User Values: VAL23**	
User Values: VAL8		**User Values: VAL24**	
User Values: VAL9		**User Values: VAL25**	
User Values: VAL10		**User Values: VAL26**	
User Values: VAL11		**User Values: VAL27**	
User Values: VAL12		**User Values: VAL28**	
User Values: VAL13		**User Values: VAL29**	
User Values: VAL14		**User Values: VAL30**	
User Values: VAL15		**User Values: VAL31**	

15 Appendix

15.1 Channel frequency response

The analog to digital converters within the touch controller do not typically operate in an ideal environment. There is noise usually associated with the display, chargers, and as well as any other coupled noise through a touching object (Figure 123). When designing to operate in the presence of noise such as coupled display noise it is useful to understand the characteristics of the converter, to understand how the converter is going to handle the noise. Thus, this section briefly introduces and discusses the analog-to-digital converter's characteristics in the frequency domain. This will help when making decisions about sampling frequency and integration pulses.

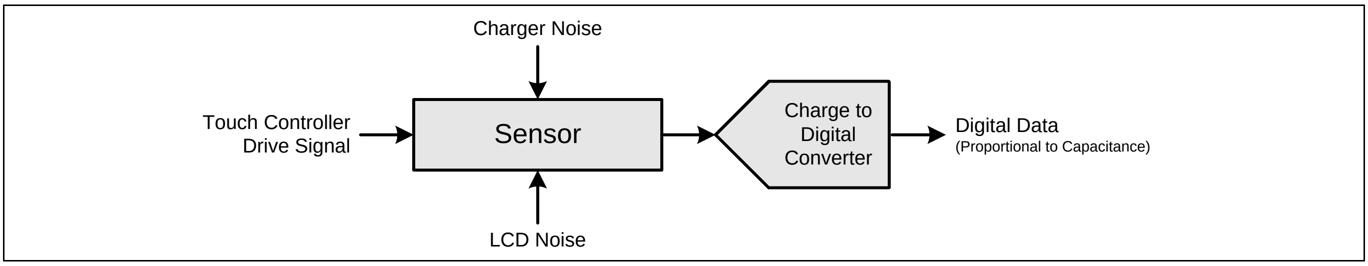


Figure 123 Injected signals into the converter

15.1.1 General frequency response

Figure 124 shows an example response of the converter. Note the large peak that is the sampling rate of the converter. Thus, the plot suggests that the converter will pass noise that exists throughout the spectrum, but around the sampling rate the converter is particularly sensitive.

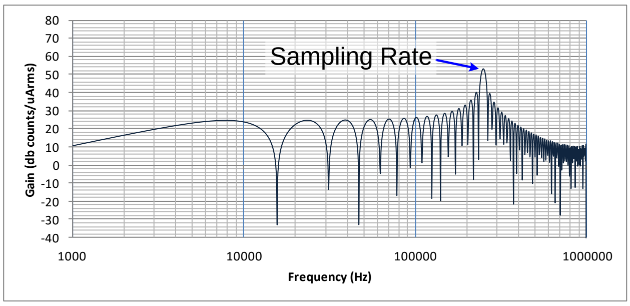


Figure 124 ADC frequency response example

The converter is sensitive at multiple frequencies. The converter passes information throughout the spectrum with potentially multiple sensitive areas.

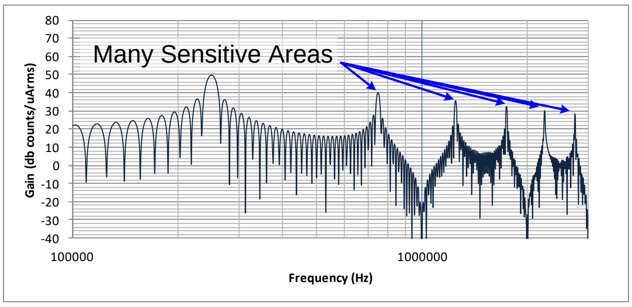


Figure 125 ADC frequency response example

15.1.2 Channel bandwidth

Increasing the number of pulses can significantly narrow the sensitive band of the converter while increasing the signal to noise sensitivity. Thus, if the noise is managed such that it does not land on the sampling rate or any of its harmonics, out of band noise will have diminishing effect on the signal of interest as pulses are increased. Figure 126 shows how dramatically the number of pulses affects the bandwidth of the channel. If there are strong noise harmonics, it may be possible to adjust the number of pulses to narrow the channel bandwidth to fit between the noise bands in the system.

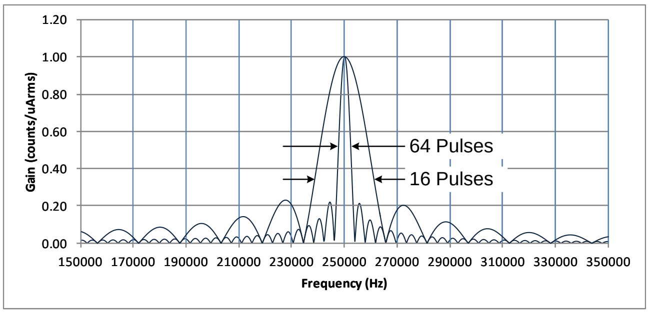


Figure 126 Bandwidth vs. pulses

15.1.3 Signal to noise transfer characteristics

If the signal of interest is the sampling rate, the rate at which the sensor is stimulated to measure charge through the converter, then all other frequencies relate to noise. Figure 127 and Figure 128 shows that there is a clear floor where noise will pass through the converter, thus if there is display noise, charger noise, or any other noise then it is desirable to insure the signal of interest (i.e. the sampling rate) is well above the noise at any other frequency. Note that the harmonics of the sampling rate are also considered frequencies of interest when measuring charge is concerned.

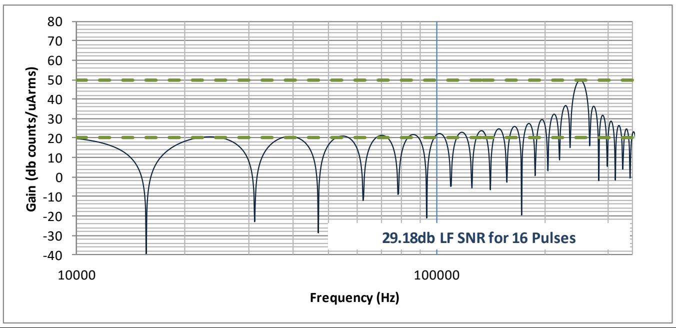


Figure 127 Signal to noise transfer for 16 pulses

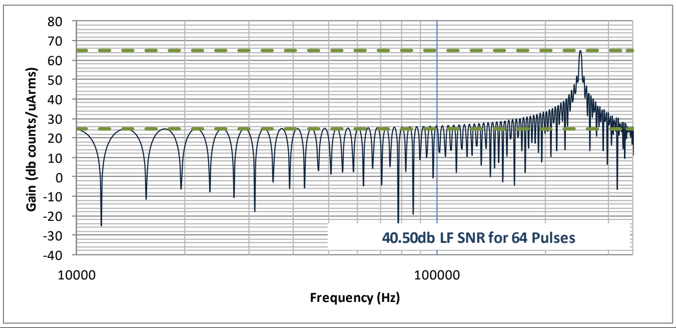


Figure 128 Signal to noise transfer for 64 pulses

15.1.4 Integration options

As suggested before there are harmonics of the sample rate that are sensitive, and this sensitivity is programmable in the touch controller. There are two integration options built into the touch controller. With full integration, the channel is sensitive to all harmonics of a square signal (3rd, 5th, 7th, etc.). In addition, it also has an equal number of heavy attenuation on all even harmonics. Figure 129 shows the frequency response.

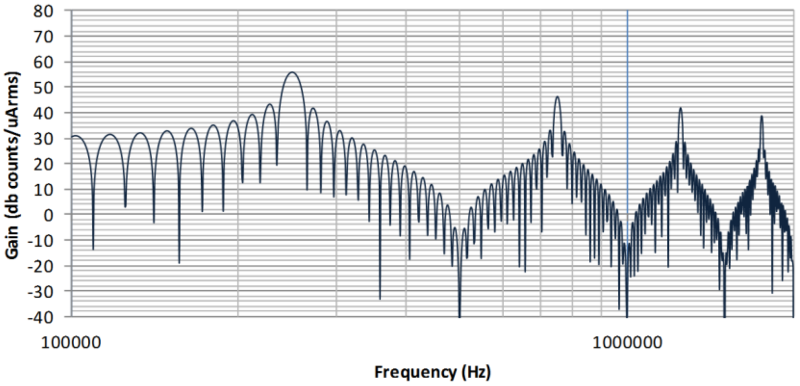


Figure 129 Full integration

Using the limited integration option allows making some tradeoff with the attenuation to be able to eliminate strong sensitivity to some harmonics. In this case, the integration internal to the converter is cut to 67% of the TX period to yield reshaping of the frequency response. As shown in Figure 130 the channel is less sensitive to the 3rd harmonic (as well as other higher harmonics). However, it becomes somewhat more sensitive to even harmonics in addition to a slight loss in gain.

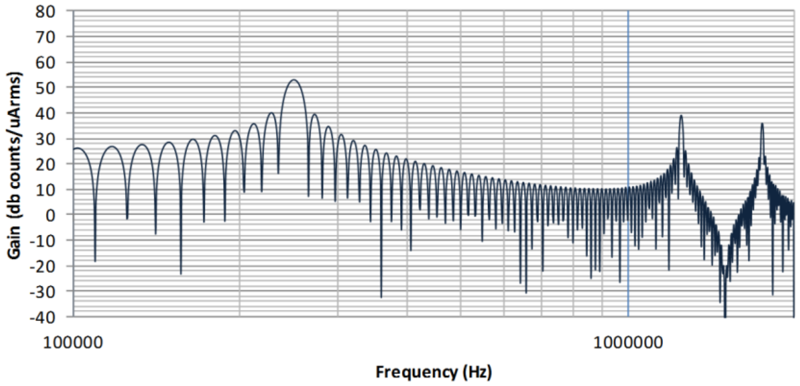


Figure 130 Limited integration

15.1.5 Sensor response

In an ideal situation, a square wave is injected into the touch sensor with its usual harmonics at the 1st, 3rd, 5th, etc. Likewise, noise is also injected. The sensor naturally filters out much of the information injected as part of the touch controller stimulation, but there is still information that is passed. The sensor acts as a band-pass filter, and its most sensitive area may be near or above the sampling rate of choice as shown in Figure 131. This may not necessarily be an issue, but one should not count on the sensor to attenuate all noise. The channel is sensitive at multiple frequencies.

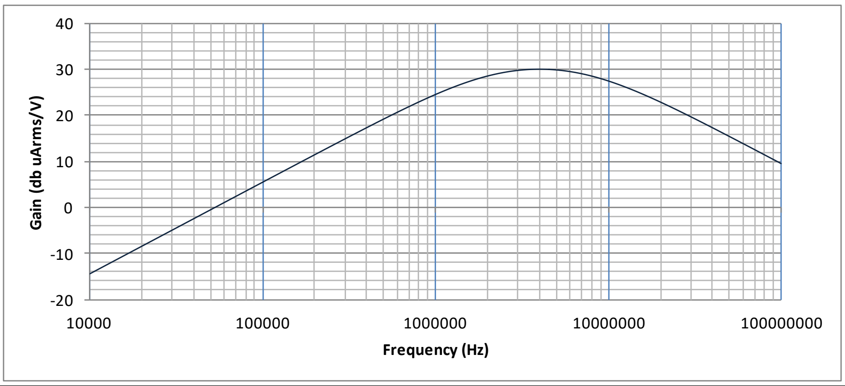


Figure 131 Sensor transfer characteristics example

Revision history

Document revision	Date	Description of changes
**	2015-08-19	Initial release
*A	2016-04-01	Added sensors pin assignment Updated parameters to the actual FW Added EMI/EMC emissions tuning
*B	2017-02-22	Add gesture section Update glove Add table entries for all available parameters Add CYAT8165 (TSG6L) support Add details and tuning steps for slew rate control Add refresh rate section Move noise section before advanced tuning section Move calibration description to introduction section and dynamic calibration to advanced tuning section
*C	2017-04-04	Update for new CYAT8165 (TSG6L) firmware release Add: **TSS: TX_SPREADER_STEP_NUM**, also updated spreader section Add: **Wake-up Button: WU_BTN_xxx**, also added new section Fix minimum IDAC and check the part numbers
*D	2017-10-20	Update for new CYAT8165 (TSG6L) firmware release • Add Stuck Touch Timeout • Add WF Water State Timeout • Add Attenuator (MC, SC-TX, SC-RX, BTN-MC, BTN-SC) • Add Host Reset Pin (Enable, DM, Active-level, Pulse-delay) • Remove Wake-Up-Btn-Touch-Thresh Hi/Lo Update title to by “CYAT816X” from “CYAT81X” Add section on display synchronization Many changes for formatting, spelling, grammar and clarity Update Gesture section
*E	2018-04-25	Change document name to include slider MPN Update all parameters for TSG6XL base FW 1.3 Fix parameter ranges and default values Add slider support Add detail to the baseline reset debounce section Add detail to the temperature compensation section Fix touch suppression in finger detection parameter table
*F	2019-01-02	Update GIDAC description and limits Fix spreader step to be applied twice per pulse Remove CA level-1 (previously removed from FW) Add section on ageing
*G	2019-06-29	Major doc reorg: move all background info to Appendices Update all parameters for TSG6L base FW 1.4

Document revision	Date	Description of changes
		Update all parameters for TSG6XL base FW 1.4 Add tuning for burst noise Rewrite hover section Recognizing and tuning channel saturation Tuning for fast swipe Tuning for VDDA start-up delay and pump-delay Improve charger armor's self-cap content TX spreader information fixed Note that wake-on-button pins must be tested manually Add more data to the ageing section Equation for TX-Patterns fixed for TSG6L Restore tuning info for glove buttons
*H	2022-05-25	Rebranding to PSoC™ Automotive Multitouch Add Channel Saturation section Clarify "Total Time" in response time table Fix spreader example, and make half-period clearer Improve Aggressive Temperature Compensation section wording Verify RAM variable addresses/formats
*I	2023-11-05	When copying a TTHE project, check custom values (section 3.1) Discard time calculation is round down (section 6.5.4) xxx_SCALE parameter equations should be verified (section 7.3) Fix eq-34 (Z9->Z8) in section 7.3 Note MC BL scan can trigger SC-LFT->MC-LFT (section 7.4) Clarify when SC thresh used, and sum of SNS_WIDTH (section 7.4) Updated and corrected CMF in section 8.2.3.1 Add recommended panel XY filter settings (section 8.8.1) When tuning edge, disable XY filters (section 8.8.6)
*J	2024-04-17	Updated to IFX format Add warning for non-default params in Initial Setup (section 3.3.1) Attenuator set to specific value, not automatic (section 6.4.3) Define INT pin setup (section 3.1) HW bring-up check error signal (section 5.3) POST limits should be loose (MTK limits tight) (section 13) Add wake-up-button power estimates (section 10.3.4) Reformat wet finger tracking section, add overview (section 8.12)

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2024-04-17

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-00156 Rev. *J