AZD137 - User Interface Application Note

A brief introduction to the Azoteq user interfaces and their functions

Contents

1 Introduction

Azoteq offers various multi-sensor solutions to meet the needs of the industrial designer, specifically in the domain of capacitive, inductive and Hall-effect technologies. Said technologies are accompanied by comprehensive user interface (UI) platforms, ensuring that the unique requirements of the application are met.

The definition of the term 'UI' in the context of this document is the *set of on-chip algorithms and their related settings that are in service of a greater overall function*. Example functions include wear detection, capacitive sliders or gesture recognition. The *user interface* term then refers to the fact these settings may be adjusted via the appropriate Azoteq graphical user interface (GUI) or microcontroller unit (MCU). Furthermore, the result of the relevant algorithm may be read by the GUI or MCU as flags (shown in Figure 1.1) or as analogue data in certain cases.

This document serves as an introduction to the various UIs, with a brief explanation of each. The designer is encouraged to consult the appropriate Azoteq IC datasheet for more detailed information.

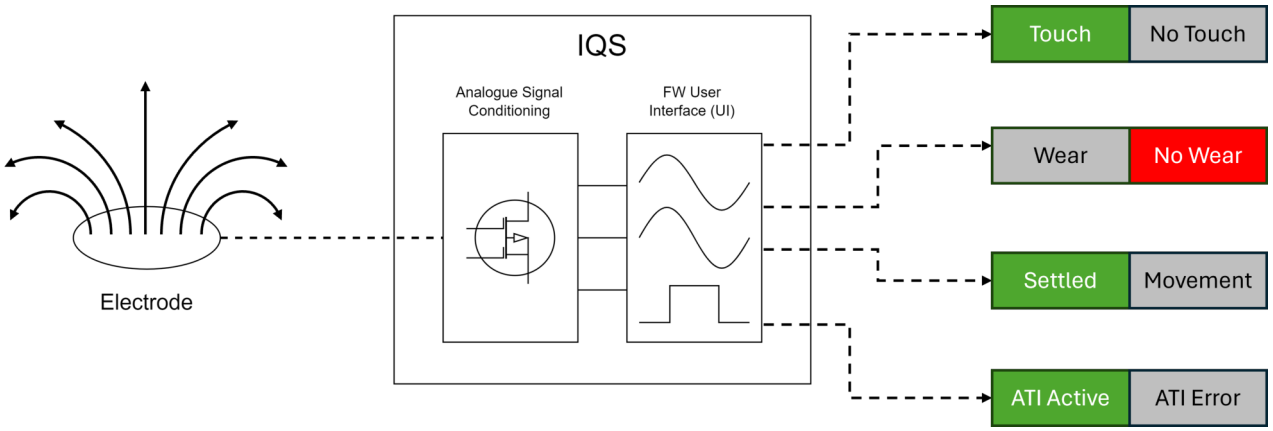


Figure 1.1: Simple Diagram of a Generic IQS Device and its Main Components

2 Touch and Proxmity UI

This section is dedicated to the touch and proximity UI. The concepts of touch and proximity need to be well understood, as they form the basis of the UIs explained from Section 3 onwards. Note that the term *Prox* is shorthand for *Proximity*, and will henceforth be used in this document. Also, the terms *signal*, *capacitive signal* and *counts* are synonymous, and will thus be used interchangeably.

2.1 Introduction

Prox and *touch* refer to different levels achieved by the same capacitive signal and will therefore be explained together. A prox level of a signal is the deviation from its baseline when the user is near a capacitive button or sensing pad. A touch level, however, is the deviation that occurs when the user makes physical contact with the pad instead. Due to the nature of capacitance measurements, the touch level of a signal will always be greater than its prox level. Thus, a numeric threshold may enforced to distinguish these levels, and certain on-chip algorithms may be activated once these thresholds are surpassed. These algorithms include:

- Filter/LTA halting
- Debouncing
- Hysteresis

Each capacitive signal measured by the IC has a Long-Term-Average (LTA), as introduced in AZD004, which tracks slow, environmental changes in the signal and serves as a reference for interactions with the sensor pad. The LTA is typically "halted" (locked/frozen) when a prox is detected to capture the user's influence on the sensor.

Refer to Figure 2.1 for a simple visual explanation of the above concepts. The purple line shows the capacitive signal level (measured in *counts*) over time. At time instant t_1, a user approaches the sensor pad and moves away again. Since the disturbed signal crosses the prox threshold, a prox event is successfully registered by the IC. At t_2, the user momentarily touches the electrode. This signal is greater than the prox signal and crosses the touch threshold. Note that the LTA (blue line) remains constant, thereby acting as a reference for the state before signal change.

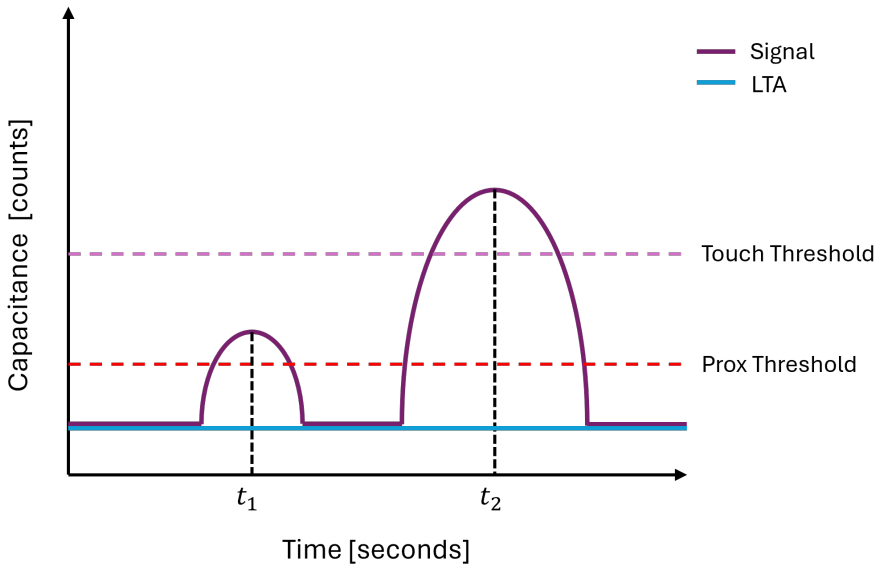


Figure 2.1: Illustration of Prox and Touch Signal Levels

Every sensing electrode has a region of sensitivity surrounding it, which may be thought of as its *sensing region* and is illustrated in Figure 2.2 as a dome shape. The Cx copper plate is the electrode, whereas the outer ring is connected to electrical ground. A bigger electrode will have a larger sensing region and vice versa. The sensing region itself is an imaginary boundary of the concentrated electric field that propagates from the Cx to the ground ring. When a user disturbs the dome, a change in signal will be measured by the IC, and is proportional to the volume of the disturbance. Note that the size, shape and placement of the Cx and ground conductors on the PCB will influence the shape of the sensing region. While the PCB shown in Fig. 2.2 is well-defined and easy to visualise, more realistic implementations are less well-defined and will lead to a distorted sensing region.

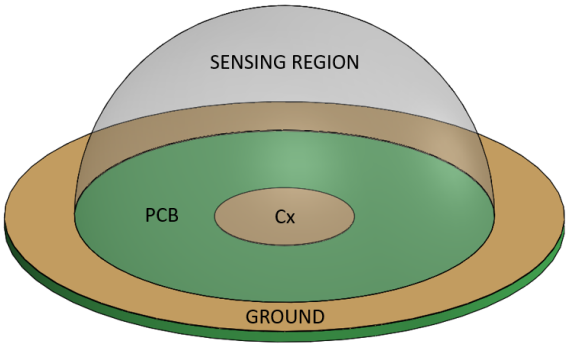


Figure 2.2: Electrode Showing Its Sensing Region With a Surrounding Ground Ring

2.2 Touch and Prox UI Applications

The following list includes example applications where only the *touch* part of the UI is utilised:

- Keypad buttons
- Laptop trackpads
- Volume sliders
- Light switch dimmers

Some applications will rely exclusively on the prox part of the UI to meet their need for user interactions. Care should be taken when designing the electrodes for these applications, as prox signals are significantly smaller than touch signals, making them harder to detect consistently. Example applications include:

- Headphone wear detection
- Non-contact soap dispenser
- Keyboard backlighting
- Low power wake-up feature in any device

A notable application of the Prox part of the UI in the above list is the low-power wake-up feature, in that it allows for significant power saving in the main device. As an example, refer to the keyboard backlighting application. If the backlighting is kept on, it will lead to excessive and unnecessary power consumption. However, if the Prox UI is implemented, the backlighting may be turned on or off depending on whether a user is nearby, thus prolonging its battery life.

2.3 UI Flow Diagram

The flow diagram of the Touch and Prox UI is given in Figure 2.3. The *delta* (Δ) referred to in the figure is given by the formula:

$$\Delta = |Signal - LTA|. \tag{1}$$

For a visualisation of the above formula, refer to the prox and touch events in Figure 2.1, where non-zero deltas can be seen at t_1 and t_2. The prox and touch states may be set with a timeout so that the delta that formed is cleared after a predefined period. In the case of the diagram in Figure 2.3, the device may also be switched to a low power mode after timeout events.

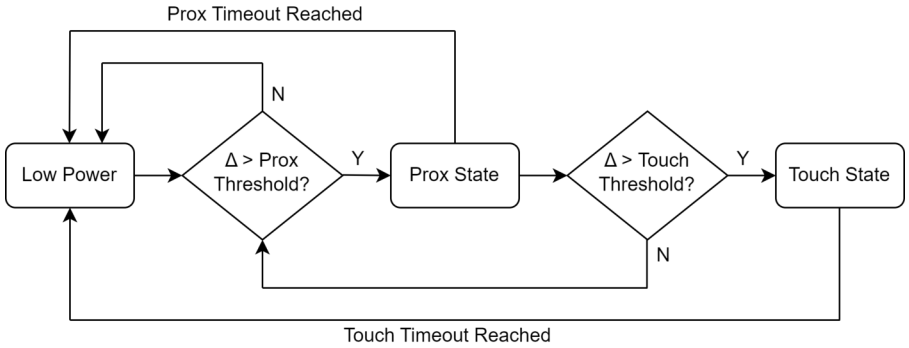


Figure 2.3: Prox Touch Flow Diagram

The prox threshold on the IQS devices is typically defined as a counts value. For example, consider an application with an LTA value of 800. It is required that a prox state is registered at 820. Then, the prox threshold is set at 20 counts. If the environment changes such that the LTA is 850, then the prox state will only register at 870. Thus, the prox threshold is defined as the number of counts by which the capacitive signal must differ from the LTA.

The touch threshold is set similarly, except that it is typically defined in terms of percentage. Consider again an application with an LTA of 800. It is required that a touch event be registered at or above 1000 counts. Then the touch threshold must be 25%, since

$$\frac{|1000 - 800|}{800} = 0.25. \tag{2}$$

2.4 Design considerations

The following key settings should be noted:

- **Prox Threshold**: This is usually chosen as a small value, but should be higher than the worst-case noise environment for the application.
- **Touch Threshold**: The touch threshold percentage should be selected that a touch triggers after a proximity event. It is usually significantly higher than the prox threshold.
- **IIR Filter (Beta Filter)**: A low beta filter value means less filtering, thus faster response. A high beta filter value means more filtering, thus a slower response.

Please refer to AZD004 and AZD125 for layout and other design considerations.

3 Slider and Gesture UI

This section is dedicated to sliders and gestures, restricted to Azoteq's ProxFusion® generic sensor series devices such as IQS7222A/C/D and the IQS3230xx(Axx) for single finger contact movement on sliders, miniature trackpads, small touchscreens or similar touch-sensitive display/interface emulation areas. The goal is to provide accurate finger tracking information and identify specific gestures like tap, hold, swipe or flick actions to assign product UI functionality.

3.1 Typical slider and gesture applications

The application of sliders- and gesture-based UI control can be utilised extensively in various products. A few common examples are listed below:

- Audio product volume control.
- Small rudimentary trackpad/touchscreen-equipped products for navigational control.
- Remote controls and home entertainment systems.
- Lighting and dim control for home automation, recreational- and outdoor lamps, toys etc.
- Portable electronics and PC peripherals.

3.2 Gesture descriptions

3.2.1 Tap gesture

A tap gesture is validated based on the duration of a finite touch on any channel or group of channels. The following conditions need to be met for a valid tap gesture:

1. The touch condition must exceed the minimum gesture time setting.
2. The touch condition should clear (touch release) before the maximum tap time is reached.
3. Any resulting change in co-ordinates shouldn't exceed the maximum tap distance setting.

Note:

- Only single tap gestures are detected on IQS7222x and IQS323 devices. Successive tap events for double and triple tap recognition should be monitored by the host controller (MCU) with applicable checked dead /air time in between taps.
- This excludes the IQS7211E with special gesture recognition for double and triple taps.

3.2.2 Hold gesture

A hold gesture is defined as a prolonged touch condition, similar to a stationary tap without releasing. The following conditions need to be met for a valid hold gesture:

1. The touch condition should persist (without interruption) for a period longer than the minimum gesture time and should also surpass the maximum tap & swipe times.
2. The continuous touch period should exceed the minimum hold time setting.
3. Any resulting change in co-ordinates shouldn't exceed the maximum tap distance setting.

Note:

- Hold events persist for as long as the touch remains (or until a touch timeout is reached).
- The minimum hold time should always be set longer than the tap and swipe time settings.

3.2.3 Swipe and Flick gestures

The difference between a flick and a swipe gesture is defined as follows:

- **Swipe:** A sliding movement of the finger in a certain direction on the touch-sensitive surface.
- **Flick:** A sliding movement of the finger in a certain direction on the touch-sensitive surface requiring the *removal or lifting of one's finger contact* with the surface after the travel.

The following conditions must be met for a valid *swipe* gesture:

1. The touch condition must exceed the minimum gesture time setting.
2. The touch condition should persist uninterrupted during the swipe movement execution and should be completed before the maximum swipe time is reached.
3. The resultant relative slider co-ordinate change achieved should exceed the minimum swipe distance setting.

The following conditions need to be met for a valid *flick* gesture:

1. The touch condition must exceed the minimum gesture time setting.
2. The touch condition should persist uninterrupted during the swipe movement execution and should be completed *with the touch released* before the maximum swipe time is reached.
3. The resultant relative slider co-ordinate change achieved should exceed the minimum swipe distance setting.

Note:

- A swipe event followed by a flick event upon release may be detected in quick succession for a single gesture movement if the conditions for each gesture are successfully met at that instance.

Figure 3.1 describes how gestures are logically distinguished based on touch status, time and achieved relative co-ordinate change.

3.3 Layout and design considerations

3.3.1 Slider co-ordinate calculation

Gesture detection accuracy and reliability are dependent on the slider layout and setup. The resultant channel data is processed upon touch interaction to calculate the slider (or trackpad) co-ordinates used as input to the Gesture UI. There are several algorithms to calculate slider co-ordinates. One of the simplest and fastest algorithms, employed on the featured ProxFusion® sensors is a weighted average calculation described by the following formula:

$$Co\text{-}ordinate = \frac{\sum_{i=0}^{N-1} x_i a_i}{\sum_{i=0}^{N-1} x_i} \times \frac{2^b}{(N-1)}, \tag{3}$$

where:

- N is the number of channels used in the slider (typically $N = 3$ or 4),
- x_i is the delta values of the i^{th} sensor channel,
- a_i is the weight assigned to the i^{th} sensor channel, and
- b is the bit resolution.

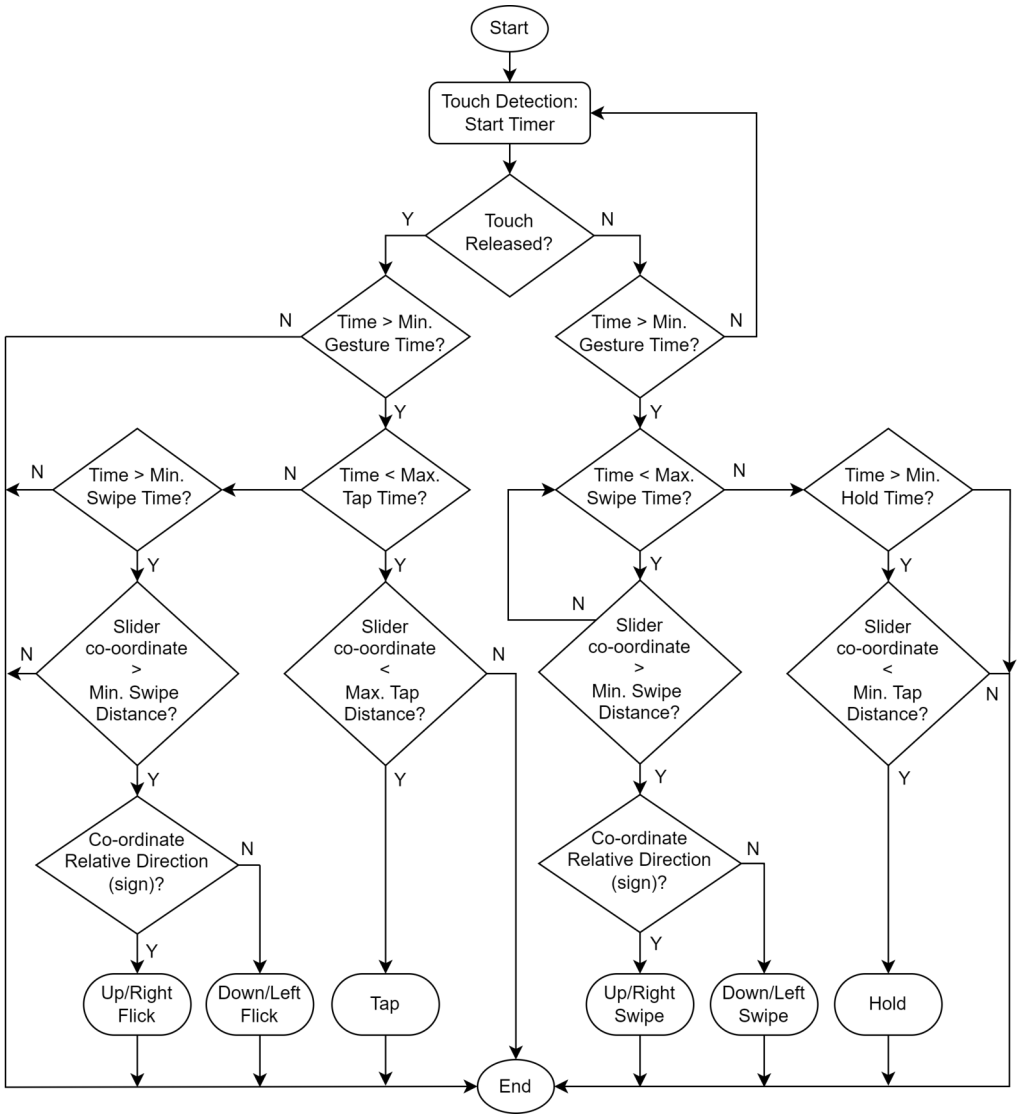


Figure 3.1: Flow chart of gesture UI state detection

3.3.2 Slider specifications

Table 3.1 may be used as a reference for typical slider resolution and range, channel delta, as well as co-ordinate trimming specifications.

Table 3.1: Slider Specifications

Specification	Minimum	Typical	Maximum
Number of channels	2 channels	3 channels	4 channels
Full co-ordinate/pixel range	1 pixel / mm	±256 pixels / channel	1000 pixels / channel or 4000 absolute maximum
Delta per channel	Above prox/touch threshold	~50 to 200 counts	<target counts
Co-ordinate deviation	0 pixels / co-ordinates	±10% of full resolution	< min. gesture limits
Trimming calibration setup to reach upper and lower full co-ordinate range	0 pixels / co-ordinates trimmed	Residual co-ordinate amounts at the start and end of slider (>0 and <max. co-ordinate)	255 pixels / co-ordinates is the abs. max. amount that can be trimmed

3.3.3 Linearity of sliders

The layout of a slider, specifically the interleaving of channels which a slider consists of, contributes to the linearity of positional co-ordinates achieved from finger movements across the slider length. To test or quantify linearity a best-fitting line is used to determine co-ordinate output as a function of the exact finger position on the slider's total active length. The maximum deviation of the recorded points from the best fitting line represents the maximum linearity error as shown in Figure 3.2.

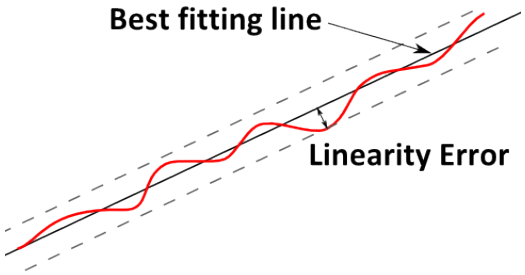


Figure 3.2: Linearity error illustration

Figure 3.3 shows a typical response of co-ordinate output from a single finger's travel over the length of a slider. The layout and number of electrodes (sensor elements or channels forming the slider) may be arbitrary, but for illustration purposes, four electrodes (x1 to x4) are indicated on the x-axis as an example. The deviation from perfect linearity in the co-ordinate output should be within acceptable limits to meet the linearity specification of the slider. Non-linearity and reduced co-ordinate ranges may occur inherently but should be minimised by conforming to the slider's specifications (refer to Table 3.1). The entire output co-ordinate range for a slider can be achieved with an additional post-processing trim step.

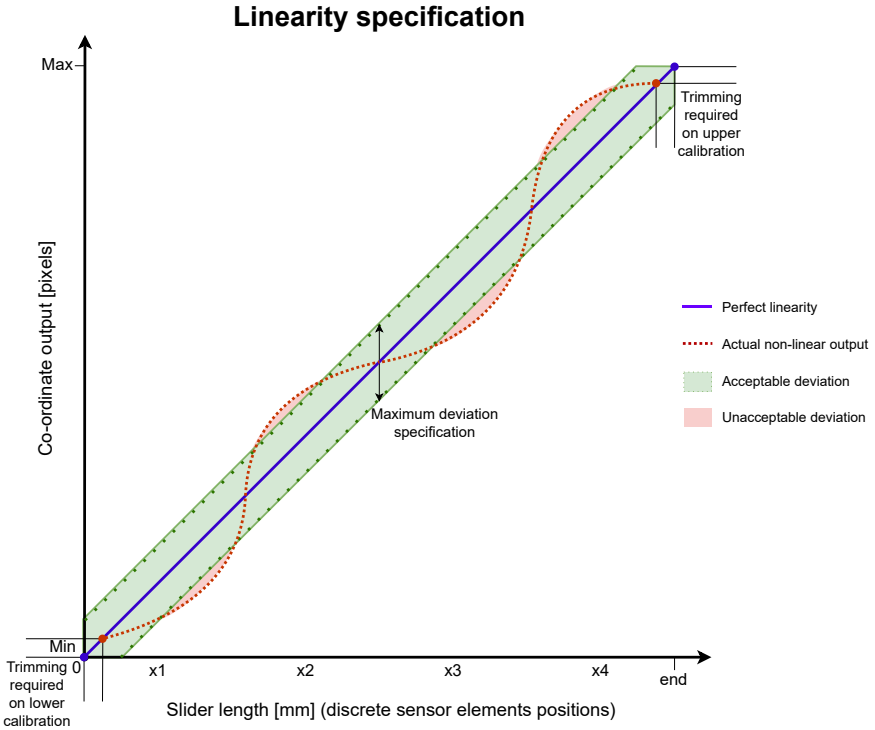


Figure 3.3: Slider co-ordinate linearity and range specification

3.4 Slider settings

3.4.1 Number of slider channels

This setting defines the number of channels used in the slider co-ordinate calculation. All channels used in the slider should be enabled and their individual *delta links* assigned.

3.4.2 Enable status link/pointer

This register is used to activate slider co-ordinate calculation based on a channel's *prox/touch* status.

3.4.3 Delta links

The order of associated channel delta linking registers in the *delta link* fields determines the order of channels' processing and directionality assignment of co-ordinate and gesture outputs. The delta link order should match the physical slider layout. For example, CH0 may be assigned to the centre CRx pad in a three-channel slider hardware layout for low-power wake-up purposes. The delta links can then be assigned in the order $[CH1_{deltalink}; CH0_{deltalink}; CH2_{deltalink}]$ to process the channels for slider and gesture calculations in the correct (logical) order.

3.4.4 Slider resolution

The *slider resolution* defines the full range of slider co-ordinates or output values (before *upper-* and *lower calibration trimming*). The *gesture configuration/setup* registers must be set in accordance with the *slider resolution*.

3.4.5 Slider (Upper/Lower) calibration - Trimming

Due to boundary conditions at the edges of a slider or trackpad, it is unlikely that the co-ordinate extreme values will be achievable (0 and max slider/X/Y resolution). To achieve this, the edges can be trimmed with a configurable calibration amount (Upper (X&Y) / Lower (X&Y) calibration) on-chip. For example, say Lower calibration is set to 0, and a finger on the left of the slider gives a minimum co-ordinate output of 48, and a maximum of 964 for a finger to the far right (for slider resolution set to 1000 as an example), then a lower calibration of 50 and an upper calibration of 40 could be used to trim away the 'dead' area, and the full 0 to 1000 range will be achievable.

3.4.6 Dynamic filter

Relative to the speed of movement of co-ordinate change, the slider output IIR filter dynamically adjusts the amount of filtering (damping factor) performed. When fast movement is detected, and quick response is required, less filtering is done. Similarly, when a co-ordinate is stationary or moving at a slower speed, more filtering can be applied to eliminate co-ordinate jitter.

The damping factor (beta) is adjusted depending on the speed of movement. Two of these parameters are adjustable to fine-tune the dynamic filter if required:

- Slow/Static beta
- Bottom Filter speed

The speeds are defined as the distance (relative change in co-ordinates/pixels of the full *slider resolution*) travelled in one cycle (pixels/cycle). The configured *slow/static beta* will decrease linearly depending on the current speed for samples in between the bottom and top speed settings.

The concept is illustrated in Figure 3.4, which shows that at movement speeds below the bottom filter speed limit (or simply bottom speed), slow/static filtering is performed, while at speeds higher than the top speed limit, no filtering is done to ensure the user input is still correctly captured. Between these limits, the damping factor is adjusted.

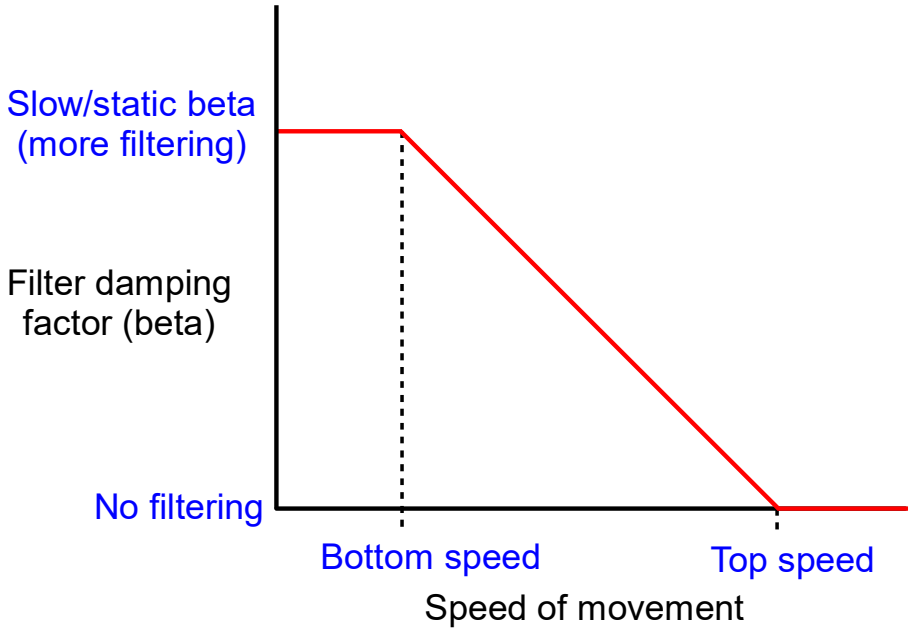


Figure 3.4: Slider co-ordinate dynamic filter

3.4.7 Wheels

The IQS7222C *does not* have a gesture recognition UI, but is the only device that offers the capability of wheel (circular slider) output. The co-ordinate calculator is modified when *wheel enable* setting is configured to change the typical slider co-ordinate range to wrap at the end and start points. The wheel layout should consist of 3 or 4 equally sized and interleaved elements in a circular pattern with start and end elements alongside each other. Refer to AZD125 for more design guidelines.

3.5 Gesture settings

Gestures may be enabled/disabled individually or appropriate event mask bits may be set if certain gestures are not needed and to prevent accidental events.

The IQS7222*D* has two additional bit settings denoted as:

- Strict X swipe
- Strict Y swipe

This imposes an additional requirement on swipe and flick gestures, in the specific direction, requiring that the X/Y co-ordinate travel distance needs to be at least twice that of the co-ordinate travel experienced in the other axis. The trackpad canvas shown in Figure 3.5 illustrates examples of swipes that would be successfully classified as strict Y swipes (blue area) and strict X swipes (green area).

The gesture co-ordinate thresholds (maximum tap distance and minimum swipe distance) should always be less than the full resolution setting to operate sensibly. The minimum and maximum gesture time settings should be set appropriately considering the power mode report rates and delays that

might occur due to waking from lower power modes.

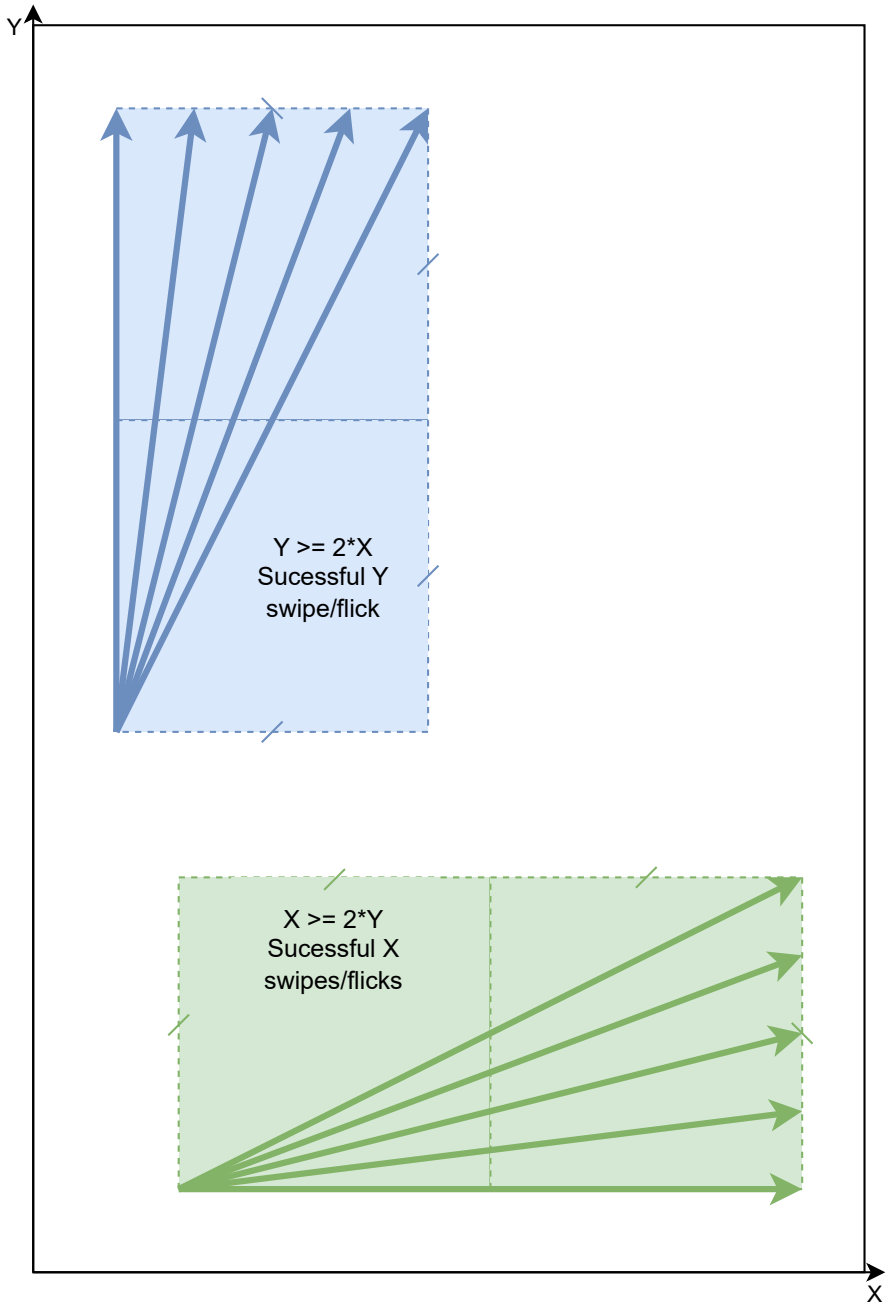


Figure 3.5: Trackpad Canvas Showing successful Strict X/Y Swipes/Flicks for IQS7222D

4 DYCAL™ UI

4.1 Introduction

The Azoteq DYCAL™ (Dynamic Calibration) user interface allows for dynamic threshold adjustment depending on the device entering touch or exiting touch.

Capacitive sensing is sensitive to capacitive changes even below 100fF in many capacitive sensing applications. With this in mind, the system is also similarly sensitive to all the dielectric changes that have an impact on the measured capacitance.

Temperate and humidity changes are the main factors that contribute to drift in the measured capacitance. In applications where the “touch” is expected to last for a long time, capacitance drift plays a big role in the accuracy of the sensor in the long term. The DYCAL™ algorithm reduces the capacitive drift impact during touch mode (TM) and ensures a reliable release condition even in challenging environments.

4.2 Overview

DYCAL™ compensates for sensor drift during prolonged activation. It is a special form of hysteresis that can track slow varying environmental change even while the sensor is in a touch state (or TM), which is ideal for various long-term activation applications, such as wear detection. The DYCAL™ algorithm is offered in two different types, as detailed in Table 4.1 below.

Table 4.1: Original DYCAL™ and DYCAL™ 2 Details

Detail	Original DYCAL™	DYCAL™ 2	Comment
Power-on state	Non-TM only	Programmable option (TM or Non-TM)	Example case: the device is worn, then power is turned on. DYCAL™ 2 offers release detection in such cases where the original DYCAL™ does not offer release detection.
Release threshold	Dynamically calculated - see Figure 4.3	Fixed percentage of the reference	DYCAL™ release characteristics vary with the touch approach speed. DYCAL™ 2 release characteristics are not linked to approach speed.
Raw signal	Long-term signal amplitude varies between TM and non-TM	The signal is always recalibrated to an optimal level to detect a touch or release.	With original DYCAL™ the signal is at a less sensitive level during TM and at a more sensitive level during non-TM.

It should be noted that newer IQS devices use the DYCAL™ 2 UI and will therefore be discussed first. However, some legacy IQS devices still use the original DYCAL™ UI, and therefore its explanation is also included in this section.

4.3 DYCAL™ 2

4.3.1 Visual Description

To explain the function of DYCAL™ 2, a lid open/close example application is illustrated in Figure 4.1. In this example, the state of the lid is determined and sensed over long periods. The DYCAL™ 2 algorithm enables the user to determine an assumed power-on state that will ensure a safe and sensible output whenever the device is reset on purpose (e.g. MCU Firmware update) or by accident (e.g. electrostatic discharge). It should be noted that the “lid state” (shown in green) will automatically correct itself during use (see the first “close lid” event after power-on). Also, time-out events

ensure that re-calibration is done via the Auto-ATI routine. With this re-calibration, even during an active/touch state, the system ensures that sensitivity and performance are under control while the total capacitance of the system may drift during varying environmental conditions.

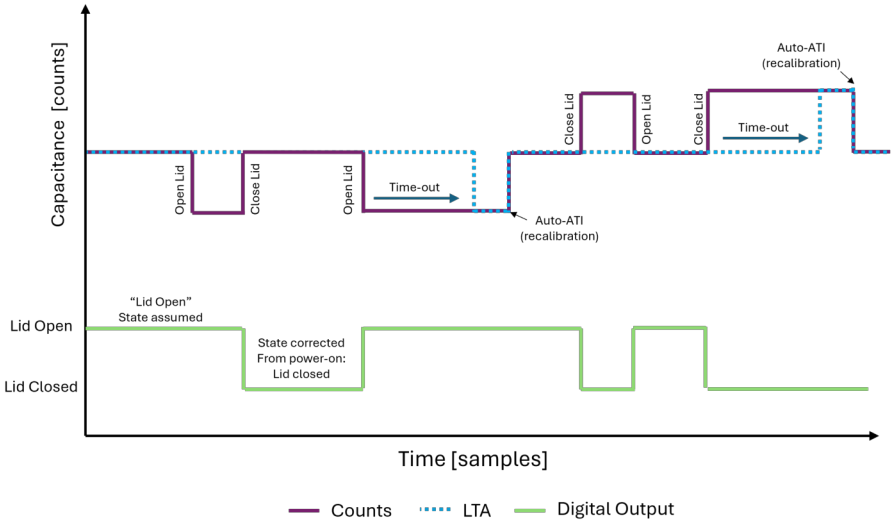
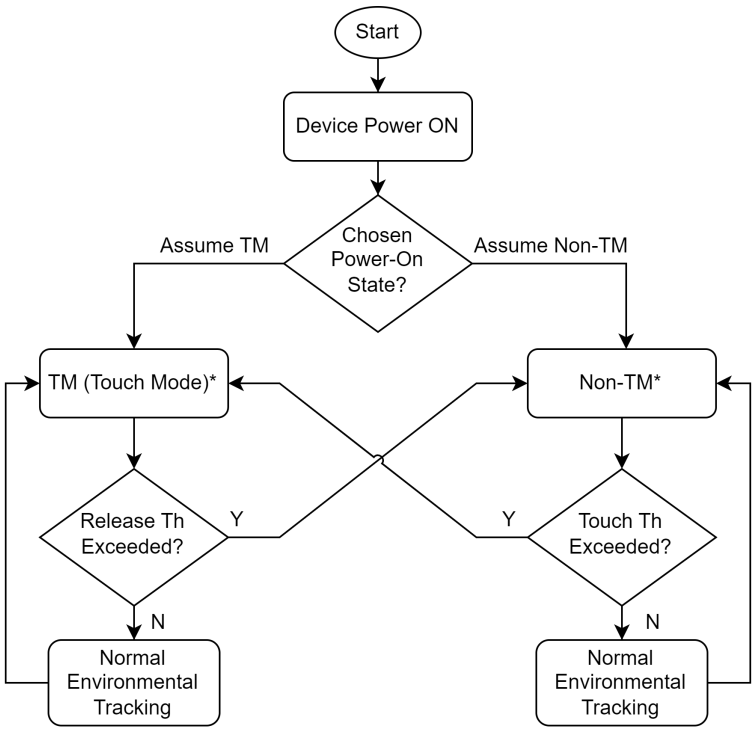


Figure 4.1: DYCAL ™ 2 visual description

4.3.2 Flow Diagram

A flow diagram is presented in Figure 4.2, which shows how the UI transitions from TM to non-TM, and vice versa.



* Will reseed and Re-ATI after state change + preset time

Figure 4.2: DYCAL ™ 2 UI Flow Diagram

4.3.3 Threshold selection

DYCAL™ 2 offers a simple threshold structure and includes linearisation options to enable a touch and release threshold via a single parameter. Linearisation is enabled by default to negate the non-linear effects of capacitance increases and decreases. The DYCAL™ 2 devices have a hysteresis register option to prevent excessive triggers when the SNR is low due to non-optimal design or harsh environmental conditions.

4.4 Typical Applications

The following are typical applications suited to the DYCAL™ 2:

- On-ear / Over-ear headphone wear detection
- Watch wear detection
- Open-close detection
- In/out of water detection

4.5 Original DYCAL™

4.5.1 Visual Description

Figure 4.3 provides a visual description of how the original DYCAL™ functions. The counts signal (or CS) is given by the black short-striped line, with its LTA in dark blue. In the first part of a graph, a prox event causes the counts to drop past the prox threshold (denoted by P_{THR} in green), leading to an OUT pin state of 1. However, when the counts drops past the touch threshold T_{THR}, the DYCAL™ UI performs the following is true:

- The device is now in TM.
- The OUT state is set to 1.
- The LTA recalibrates and follows the counts to its new, lower value.
- P_{THR} is re-calculated based on the new LTA value.
- Since the device is in TM, a release threshold REL_{THR} (light blue line) is enforced instead of T_{THR}, and is calculated relative to the delta Δ between the CS and LTA.

Thereafter, when the counts signal increases and becomes greater than REL_{THR}, the following is true:

- The device is now in non-TM.
- The OUT state is set to 0.
- The LTA re-calibrates and follows the CS signal back upwards to its new value.
- P_{THR} is re-calculated relative to the LTA.
- T_{THR} is now used again and follows the LTA.

4.5.2 Flow Diagram

A flow diagram of the original DYCAL™ is illustrated in Figure 4.4. It is notably similar to the operation of the DYCAL™ 2 UI, with the key difference that DYCAL™ 2 will perform a re-seed and re-calibration to its target counts value after a preset time-out, as shown in Figure 4.1. This re-calibration step allows DYCAL™ 2 to maintain optimal sensitivity regardless of state.

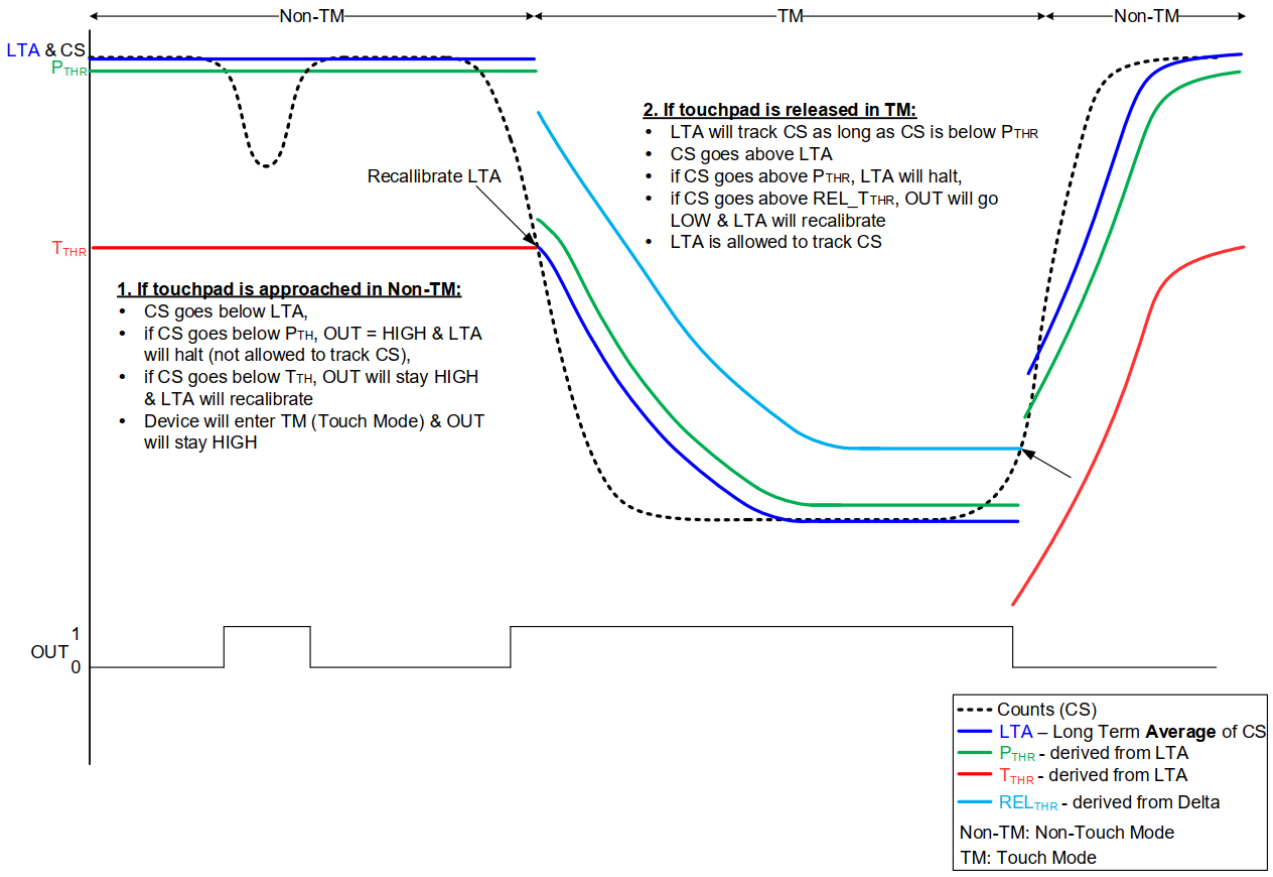


Figure 4.3: DYCAL™ (original) visual description

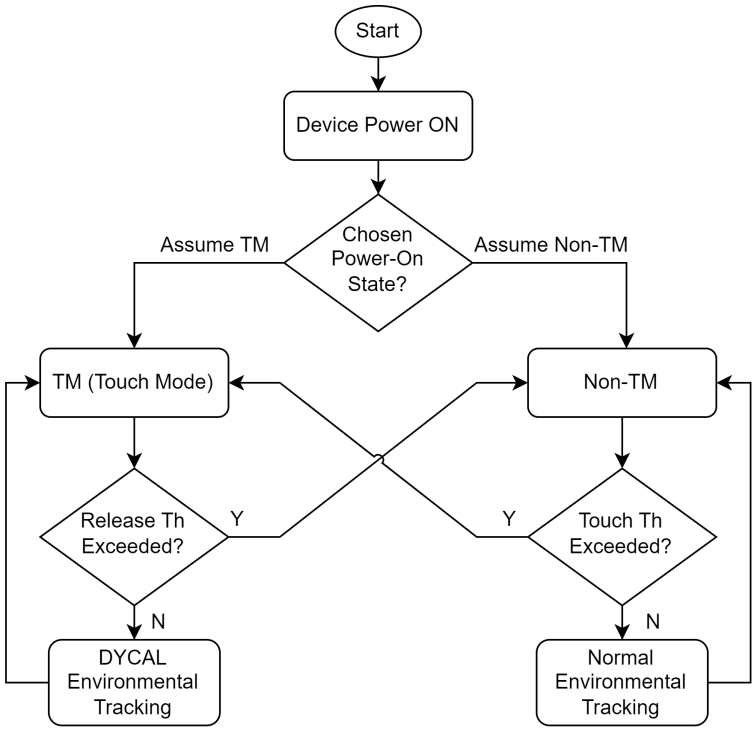


Figure 4.4: DYCAL™ UI Flow Diagram

4.5.3 Threshold selection

The touch threshold T_{THR} is selected by the designer to obtain the desired touch sensitivity. A touch event is triggered if the counts diverge more than the selected threshold from the LTA for a predetermined amount of consecutive sampling cycles (debounce). Thus, in the non-TM state, the counts signal must diverge more than the T_{THR} value below the LTA. When operating in TM, the counts must diverge more than the release threshold REL_{THR} above the LTA. With Δ defined in (1), the following conditions are used to determine if a touch or release event occurred:

NO-TOUCH STATE: $\Delta \leq T_{THR}$
TOUCH STATE: $\Delta \geq REL_{THR}$

Original DYCAL™ devices have the option to increase REL_{THR} when in TM. Small variations caused by moving a finger/hand on a touchpad will prevent unintended exits to non-TM, making the solution more robust.

After entering TM, as soon as the LTA follows to within 16 counts, an Entry Delta Δ_{entry} is calculated as:

$$\Delta_{entry} = |LTA_{entry} - LTA_{current}|. \tag{4}$$

This calculated Δ_{entry} value is then used to calculate REL_{THR}:

$$REL_{THR} = \alpha\Delta_{entry}, \tag{5}$$

where α is the chosen percentage value, typically between 0.75 and 0.85.

If upon entry the LTA value is already within 16 Counts, the Δ_{entry} is taken as the calculated touch threshold value. It is important to note that in this case (a slow threshold crossing), REL_{THR} may differ significantly from a faster threshold crossing. Depending on the threshold level compared to the maximum typical signal variation, a slow crossing will result in a much more sensitive release than a fast/normal crossing. This is especially true if T_{THR} triggers “in the air” or is more than 1mm away from the sensing pad (proximity applications). It is recommended to test all typical interaction speeds when considering a DYCAL™ solution. DYCAL™ 2 does not have a similar “touch speed” related effect.

4.6 Conclusion

Both DYCAL™ and DYCAL™ 2 devices are in production and each offers unique benefits in response and current consumption. DYCAL™ 2 devices are recommended for applications that may be sensitive to the approach-speed-dependent release threshold of DYCAL™ devices (typically proximity trigger applications).

5 Movement UI

5.1 Introduction

The Movement UI is a solution to determine if an object detected by the proximity sensor of a device is caused by a user or by a stationary object. When the device is placed near a stationary object, the counts will be stationary meaning there is no user interaction, whereas the counts would vary constantly if a user is interacting with the device.

The output of the Movement UI is typically linked to re-setting event re-seed timeouts. The basic algorithm of the movement UI is given in Figure 5.1.

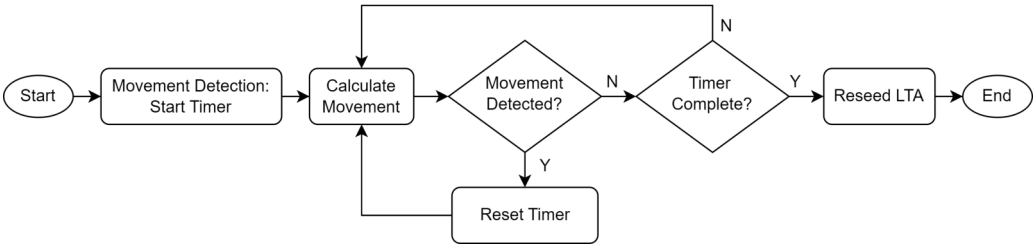


Figure 5.1: Movement UI Logical Flow Diagram

5.2 Technical Details and Operation

The movement is calculated using an estimate of the gradient of the counts, which is given by the absolute delta between the counts and a filtered version of the counts. The IIR beta filter introduces a time delay on the signal which allows for the gradient estimation.

Figure 5.2 shows the gradient estimation of an activation event with a stationary object. After the activation event, the gradient is below the threshold, allowing the event to time out. Figure 5.3 shows an event activation with the user interacting with the device. The continuous triggering of the movement threshold will repeatedly reset the event timeout, allowing the device to remain in activation.

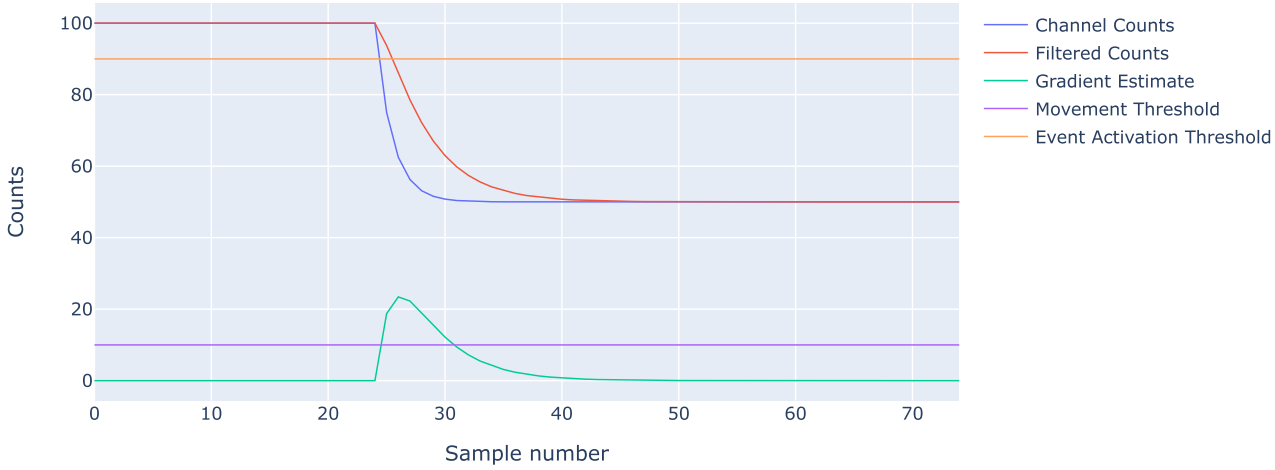


Figure 5.2: Gradient estimation illustration and movement threshold with no movement

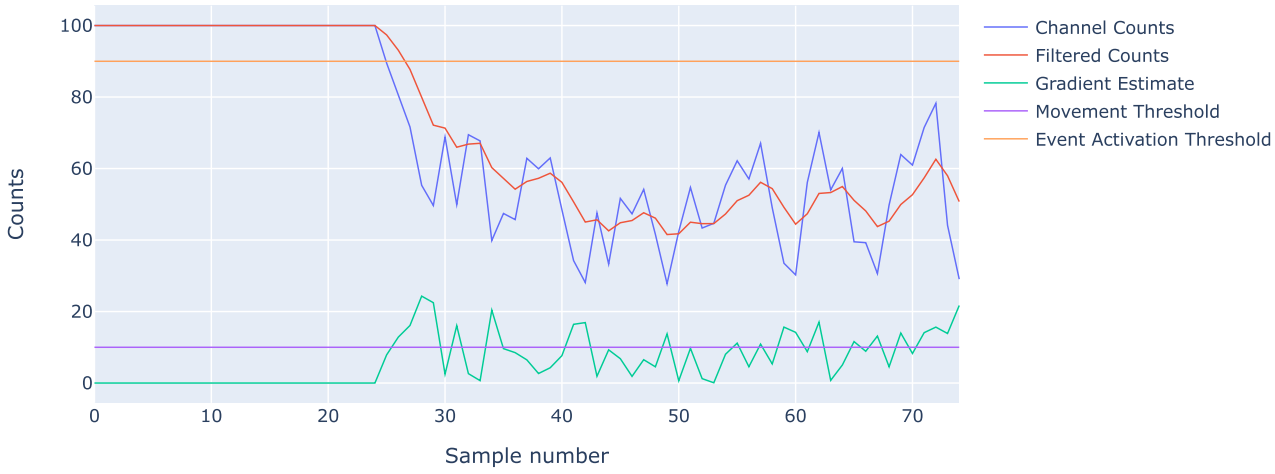


Figure 5.3: Gradient estimation illustration and movement threshold with movement

5.2.1 Report Rate and Beta Coeficient

Since the filtered signal uses an IIR beta filter, the gradient estimation can vary according to the beta coefficient selection and report rate. The signal will filter faster when the beta coefficient is lower or when the report rate is quicker. The signal being filtered faster would cause a smaller gradient estimation since there is a smaller delta between the signal and the filtered signal which would require a smaller movement threshold. This concept is illustrated in Figure 5.4.

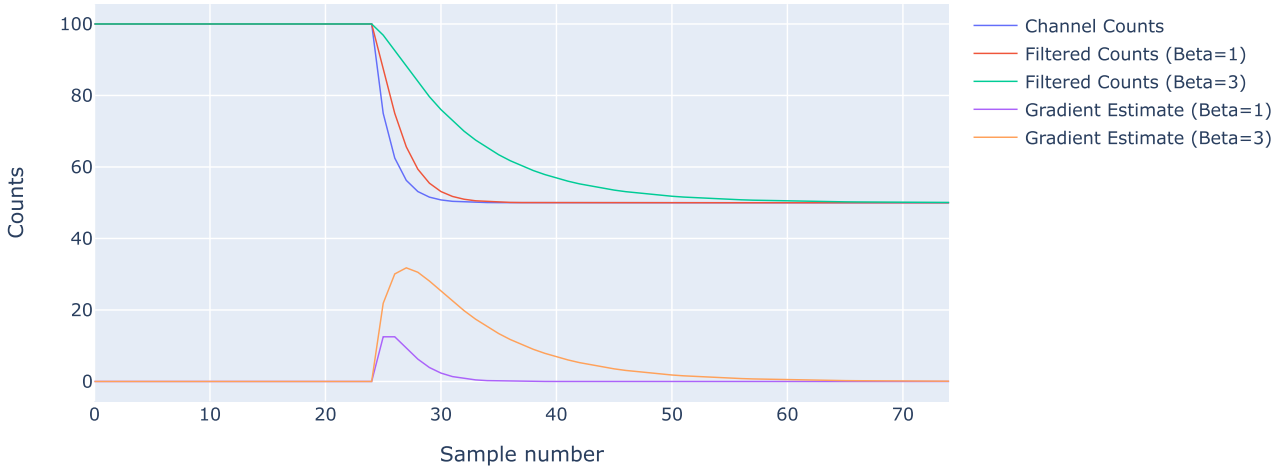


Figure 5.4: Gradient estimation with different Beta coefficients

6 Follow UI

Some ICs offer a functionality called *Follow UI*, which allows for long-term 'in-touch' tracking to compensate for temperature drifts. An example application for this functionality is wear detection, where a device is being worn for extended periods, requiring continuous capacitive sensing. This section will provide a brief explanation of how to set up this functionality, starting with the electrode configuration.

6.1 Electrode Configuration

In its simplest form, the Follow UI system requires a *sensing* and a *reference* electrode, where both function independently as self-capacitance electrodes. As the name suggests, the purpose of the sensing electrode is to detect user interaction with the device and should therefore be designed to be sensitive to human presence. The reference electrode, however, should be designed such that:

- It is not affected by user interaction.
- It is subject to the same temperature changes experienced by the sensing electrode.

The latter requirements may be achieved by using the following guidelines:

- The reference electrode must be smaller in size than the sensing electrode. This will ensure that the reference electrode remains insensitive to human interaction. It is typical for the reference electrode to be a PCB trace with a width of 0.2 mm. To avoid a too-sensitive reference electrode, its width should be equal to or less than half of the thickness of the overlay. Refer to Figure 6.1 for an illustration.

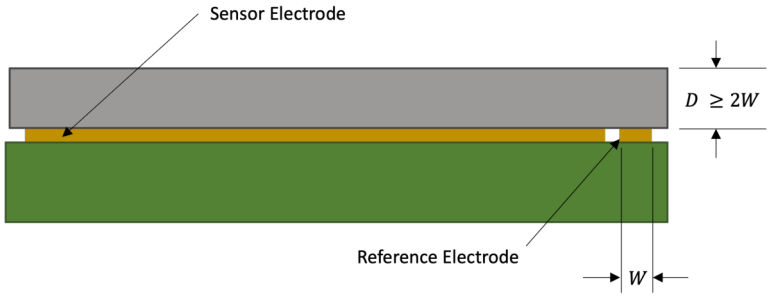


Figure 6.1: Illustration of the approximate width sizing of the reference electrode

- The reference electrode should be placed such that it spans the same region as the sensing electrode. This will ensure that temperature changes over the entire sensing region will be detected by the reference electrode. Refer to Figure 6.2 for a simple example of how the reference electrode may be routed to cover the sensing region.

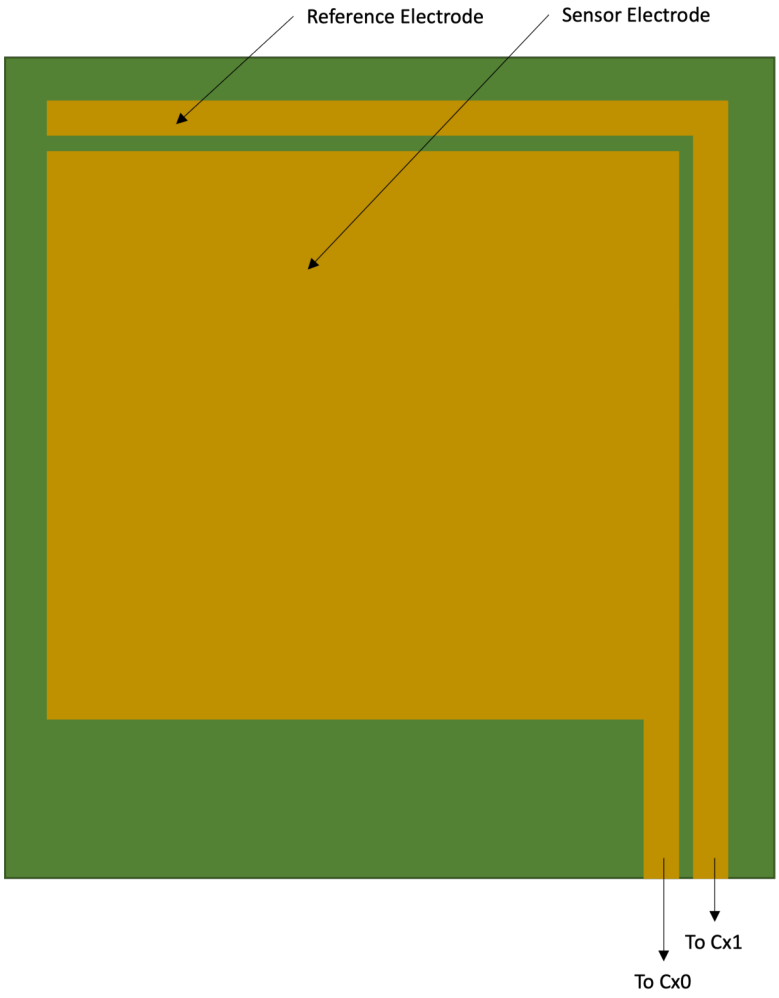


Figure 6.2: Illustration of the approximate PCB coverage of the reference electrode

- The sensor and reference electrodes must be connected to separate Cx pins, and must have separate channels. For example, the sensor electrode may be connected to Cx0 and assigned to Channel 0, while the reference electrode is connected to Cx1 and assigned to Channel 1.

6.2 GUI Configuration

The exact GUI configuration procedure will depend on the IQS device being used, but generally, there are four main steps to be performed when configuring the Follow UI:

- **Appoint a Follower Channel**. First, one of the channels needs to be designated as a follower. The channel of the sensor electrode must be selected as the **Follower**.
- **Appoint a Reference / Leader Channel**. Next, the channel that is used for the reference electrode must be selected as the **Reference** or **Leader**.
- **Select a Follow Event**. The event that will trigger the following action must also be selected. This may be when a specific channel goes into a proximity or touch state.
- **Adjust the Follow Weight**. The follow weight is a numeric value that determines how aggressively the follower channel will follow the reference/leader channel. The method to determine the follow weight is explained in Section 6.2.1.

Note that the above list is only a general list of actions required to enable the Follow UI. Please refer to the applicable IQS device's application note, datasheet or setup guide for detailed instructions.

6.2.1 How to Determine the Follow Weight

The following steps may be taken to determine the correct follow weight for your application.

1. **Perform a Temperature Characterisation Test**.
   The first step is to determine the temperature response of the application within the desired operating temperatures. This will require an environment within which the temperature can be reliably controlled. It is required to choose a *Reference Environmental Temperature* (T_{ref}) at which the electrodes are calibrated before the test. Typically, the reference temperature $T_{ref} = 25$ °C. Also, make sure of the following:

   - Both the sensor and reference channels are activated. This can be confirmed by monitoring their respective counts values in the Azoteq GUI.
   - Both the sensor and reference channels have completed their ATI procedure at T_{ref}. When this is completed, the ATI for both the Sensor and Reference channel should then be disabled.
   - The channels may not be in activation state.
   - Optionally, using the logging function in the GUI can help with the post-processing of the test results.

   Now, change the temperature of the environment to the maximum (T_{max}) or minimum (T_{min}) operating temperature, and take note of the counts values of both the sensor and reference channels as they reach this temperature. It is expected that the counts values of the sensor and reference channels will have increased or decreased with the change in temperature.

2. **Calculate the Follow Weight**
   By using the results from the temperature characterisation test, the follow weight W_f (in percentage) can be calculated using the following formula:

$$W_f = \frac{|N_{s,i} - N_{s,f}|}{|N_{r,i} - N_{r,f}|} \times 100, \tag{6}$$

   where:

   - $N_{s,i}$ is the counts value of the sensor channel at T_{ref}
   - $N_{s,f}$ is the counts value of the sensor channel at T_{max} or T_{min},
   - $N_{r,i}$ is the counts value of the reference channel at T_{ref} and,
   - $N_{r,f}$ is the counts value of the reference channel at T_{max} or T_{min}.

3. **Validate the Follow Weight**
   To validate the correctness of the follow weight, repeat the temperature test from Step 1, while the sensor channel is in activation or proximity state (or whichever condition is required for the sensor channel to initiate the following sequence.) During this test, it should be observed that the counts delta at T_{ref} and T_{min} or T_{max} should be the same. If this is not the case, then slight adjustments must be made to the follow weight until this is true.

6.3 Example Illustration of Follow Weight Calculation

To demonstrate the procedure of how to determine the follow weight value, an example will now be given.

For this example, we will use the electrode shown in Figure 6.2, which has a sensor electrode con-

nected to Cx0 and a reference electrode connected to Cx1. We will assume that this PCB is already built into a watch device as the application. In the Azoteq GUI, we assign the sensor electrode to Channel 0, and the reference electrode to Channel 1. Then, Channel 0 is set as the follower channel, while Channel 1 is set as the reference channel. Since it is a watch application, we select the follow event as a touch state. Finally, we use the ATI to set Channel 0 to 1000 counts and Channel 1 to 700 counts, at room temperature (25 °C).

Note that the counts target used here for Channel 0 is higher than the value for Channel 1. This is an extra step of caution to ensure that the reference electrode does not detect user interaction. If Channel 1's counts target was also set at 1000, it would be more sensitive and there would be a slight chance that the reference channel can be influenced by the user. Thus, by setting it lower at a target of 700, we ensure that the reference channel remains isolated from the user.

We want the watch to be operational up to 50 °C, so we will perform a controlled temperature sweep on the watch from T_{ref} = 25 °C up to T_{max} = 50 °C. Figure 6.3 below shows a typical result that may be observed in the 'SCOPE' view in the Azoteq GUI. For samples 0 to 100, the device was kept at 25 °C. Then, from samples 101 to 375, the device was heated to 50 °C, where it remained up to sample 500. It can be seen that both the sensor (blue) and reference (red) channel counts have decreased, but blue more so than red. It is this difference in rate of decrease that is captured by the follow weight, so that we may predict the LTA of the sensor channel by using the reference channel.

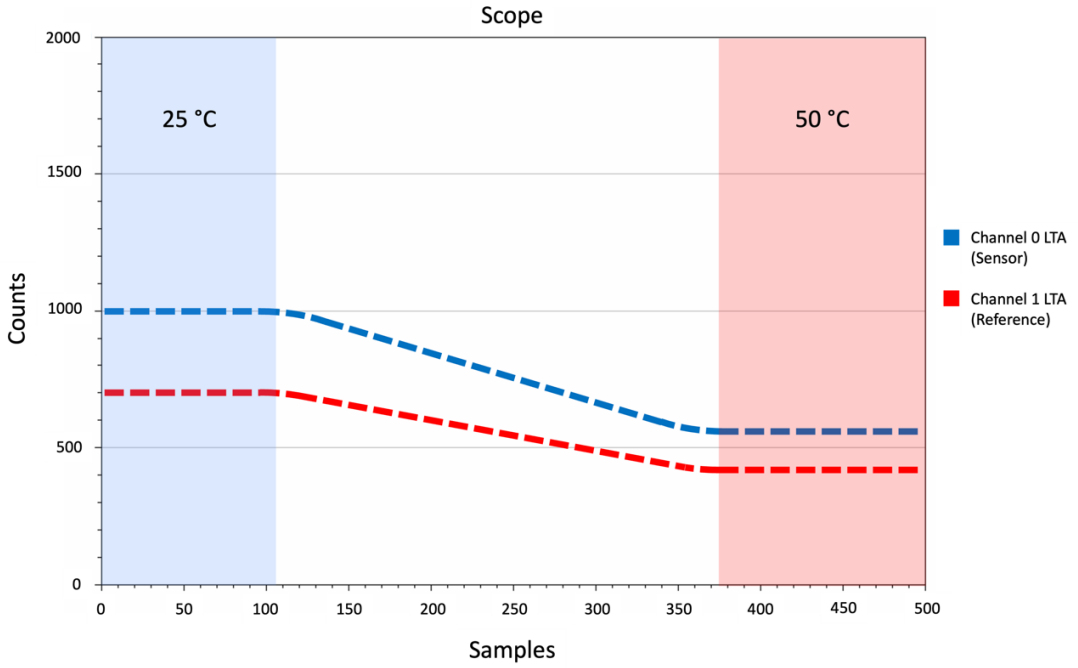


Figure 6.3: Example Scope View of a Device in a Temperature Characterisation Test

Using these results, we can calculate the follow weight W_f:

$$W_f = \frac{|1000 - 550|}{|700 - 400|} \times 100 = 150\%, \tag{7}$$

which means that for every 1 count change in the reference channel, we expect a 1.5 count change in the sensor channel within the tested operating range.

Finally, we must validate the follow weight for when there is a touch event. After applying the follow weight as determined in (7), the test is repeated, but with Channel 0 in touch. Figure 6.4 shows the desired result of the repeated test. At sample 50 the device is put in touch and it is seen how a counts delta is formed between the Channel 0 signal and LTA lines at T_{ref} = 25 °C.

Once again, at sample 100 the device is heated and here two LTAs are shown to demonstrate the effect of the Follow UI. The light blue line follows the signal drift to maintain the counts delta, and is controlled by the reference channel and the assigned follow weight. The green line is the LTA without the Follow UI, which will remain halted because of the touch event. At sample 450, at 50 °C, the touch is removed and it is shown how Channel 0 recovers to the level of the LTA that followed, indicating a successful touch release. Without the Follow UI, the device would not have registered a released touch event, since the green line did not follow the temperature change. Lastly, according to the third point of Section 6.2.1, we must verify that the counts delta at T_{ref} and at T_{max} are the same, which is shown in Figure 6.4 to be true.

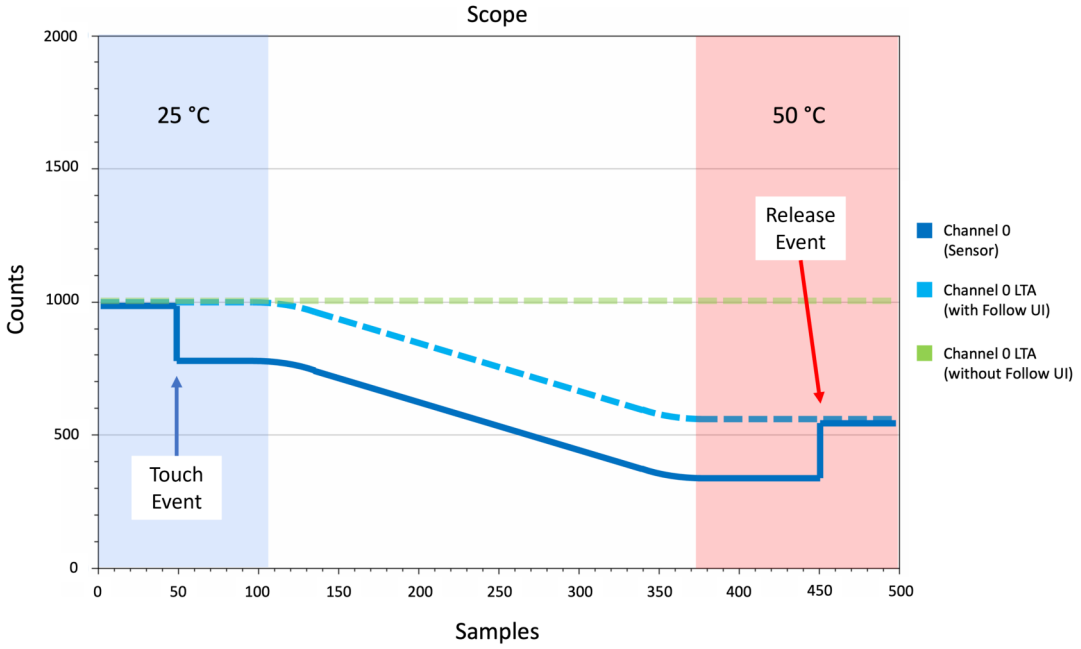


Figure 6.4: Example Scope View of a Device During the Validation Test

It should be noted that it is best to calculate the follow weight for several device samples before determining the final follow weight to account for production variation.

6.4 Follow Weight Troubleshooting

This section discusses some solutions to common anomalies that may arise during the follow weight setup process.

- **There is little or no drift occurring in the sensor channel.**
  In this case, it is not necessary to use the Follow UI for the application. However, it is recommended that the 'no drift' result is confirmed on multiple samples before concluding that the system is drift resistant.

- **There is significant drift in the sensor channel, but no drift in the reference channel.**
  The reference channel's sensitivity needs to be improved. This can be done by increasing its resolution factor, or the base value of the channel may be lowered slightly if the resolution factor increase alone is not sufficient. In the event that a lower base and a higher resolution factor do not work, the reference electrode itself must be redesigned to be larger or better positioned to observe the environmental changes. After these changes, it is important to verify that the reference channel is still immune to user interaction.

- **The sensor channel reaches a timeout state before reaching T_{min} or T_{max}.**
  The sensor channel needs to be adjusted to be less sensitive. To achieve this, the resolution factor must be lowered. Then, re-run the test to verify that a timeout does not occur, and also verify that the sensor channel is still sensitive enough to detect the user.

7 Release UI

A typical touch channel will track changes in the environment by keeping track of a Long Term Average (LTA) of the measured touch signal. Upon touch entry, the value of the LTA is taken as a reference, and the measured signal is compared to said reference to determine if the channel should remain in touch.

For short-term touches, this is an effective solution, because the external conditions are not likely to change significantly from when the reference was taken. However, for long-term touches, the external conditions are more likely to change significantly, making the reference inaccurate for the current conditions.

Similar to Wear UI discussed in Section 8, the Release UI tracks long-term touch states by exiting a touch state based on the rate at which the measured signal changes rather than by comparing the signal to a fixed reference. This accounts for drift in the measured signal, which could otherwise result in a stuck touch condition or an incorrect touch release.

The UI is best used in wear or grip detection applications, where long-term touches are a common occurrence. However, the UI is not suited for applications where rapid temperature changes are expected in the environment, since temperature spikes also lead to a high rate of change in capacitive signal similar to human interaction.

7.1 Implementation

The UI is typically configured by adjusting the following settings:

- Activation LTA Filter
- Activation Settling Threshold
- Delta Snapshot Delay
- Release Delta Percentage

The UI works by tracking an additional LTA, called the Activation LTA ($LTA_{activation}$). Like the standard LTA, the Activation LTA is a long-term average of the measured touch signal. Unlike the standard LTA, the Activation LTA is continuously updated, even when the channel is in a proximity or touch state. the Activation LTA is filtered with an adjustable software filter.

Once a touch condition is detected, the Activation LTA is continuously compared to the measured touch signal. When the difference between the measured signal and the Activation LTA is less than the *Activation Settling Threshold* for longer than the *Delta Snapshot Delay*, the delta between the standard LTA and the touch signal is recorded and stored as the Delta Snapshot ($\Delta_{snapshot}$), which is taken once the measured touch signal has stabilised.

To exit the touch state, a percentage γ of $\Delta_{snapshot}$, defined as the *Release Delta Percentage*, is compared to the difference between the measured signal and the Activation LTA.

If the following condition is satisfied, the channel is re-seeded and therefore any touch or proximity states are exited:

$$|Signal - LTA_{activation}| > \gamma\Delta_{snapshot} \tag{8}$$

The Activation LTA constantly tracks the measured signal. Therefore, for the touch state to be cleared,

the measured signal must change rapidly, such that the Activation LTA lags far enough behind the measured signal to make the above condition true.

7.2 Practical Example

Figure 7.1 shows an annotated log of a mutual capacitive channel with the Release UI enabled. Counts are shown in light green, the LTA in dark green, the Activation LTA in blue and the Delta Snapshot in purple.

The Release UI is configured with the following parameters:

- Activation Settling Threshold = 20 counts
- Delta Snapshot Delay = 20 samples
- Delta Snapshot Release Percentage = 50%
- ATI Target = 400 counts

Note that these parameters are for demonstration purposes only and should not be used as a reference for production designs. In particular, the filter for the Activation LTA has been set to filter slowly so that key events can be identified.

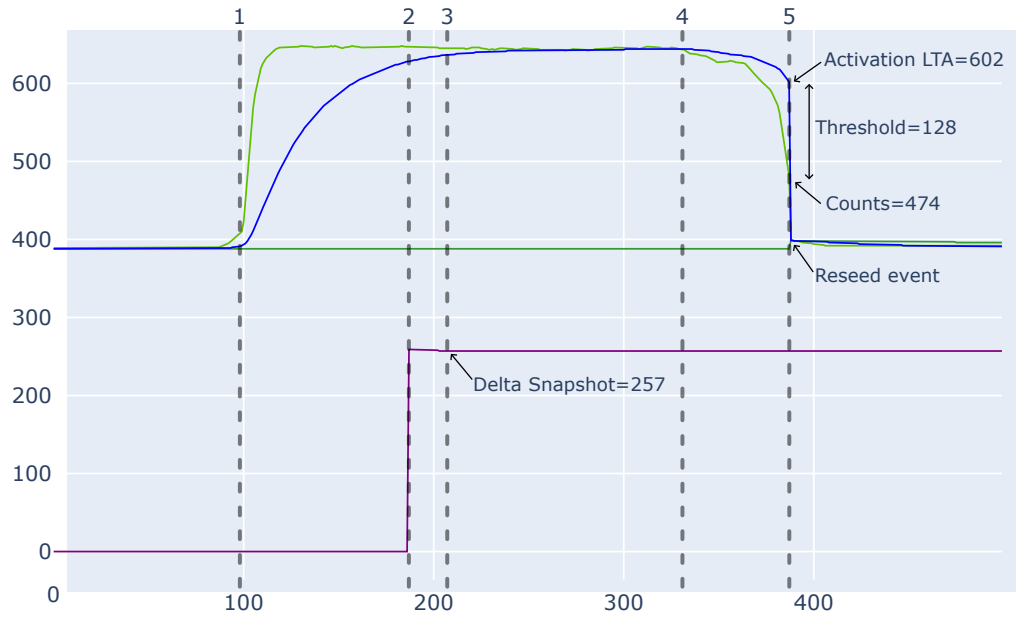


Figure 7.1: Release UI touch entry and exit

Initially, there is no interaction with the channel. The counts are near the ATI target and both the LTA and Activation LTA have stabilised at this value. A user induces a touch on the sensing electrode. Since this is a mutual capacitive channel, the counts increase. At some point, marked by the vertical dotted line labelled '1', the channel enters touch. The LTA is frozen but the Activation LTA continues to track the counts.

The vertical line marked '2' indicates the first time at which the delta between the Counts and Activation LTA is less than 20 while the channel is in touch. It is here that the Delta Snapshot logic becomes active because the delta is less than the Activation Settling Threshold. Between the lines marked '2' and '3', the Delta Snapshot is continuously updated with the difference between the Counts and the standard LTA. At the time marked by the vertical line labelled '3', the Delta Snapshot is frozen since

the delta between the Activation LTA and Counts has been less than 20 counts for 20 consecutive samples, as per the Activation Settling Threshold and Delta Snapshot Delay settings.

A percentage of the delta snapshot serves as the release threshold. In this case, the Release Delta Percentage is 50% and the Delta Snapshot was taken as 257 counts. Therefore, the release threshold is 50% of 257, which is 128 counts.

Between the dotted lines marked ‘3’ and ‘5’, the delta between the Activation LTA and Counts is less than 128. At this time, the Activation LTA is still tracking the counts, so the channel will remain in touch as long as the counts only drift slowly.

At the time marked by the dotted line labelled ‘4’, the user begins to release the touch. Initially, the release is slow and the Activation LTA can track the counts. The user makes a sudden release at the time marked by the line labelled ‘5’. The counts drop rapidly, leaving the Activation LTA lagging.

The delta between the Activation LTA and Counts exceeds the release threshold of 128 counts, triggering a channel re-seed. The re-seed snaps both LTAs to the current counts, and the channel exits touch.

7.3 Configuration

- **Activation LTA Filter** - A slower filter will make it easier to exit touch since the measured signal will outrun the Activation LTA more easily. The filter also has an effect on when the Delta Snapshot is taken. The slower the filter, the more stable the measured signal must be before the snapshot is taken.
- **Activation Settling Threshold** - The threshold must be chosen to ensure that the Delta Snapshot is taken once a stable touch condition has been detected. If the threshold is too large, the Delta Snapshot may be taken while the user is inducing short-term touch events on the channel. This could result in false releases of the touch state or stuck conditions on the channel. If the threshold is too small, the Delta Snapshot may never be taken, and the channel will effectively behave as if the Release UI is disabled.
- **Delta Snapshot Sample Delay** - If the delay is too short, the Delta Snapshot could be taken before the touch condition has stabilised. If the delay is too long, the Delta Snapshot may never be taken.
- **Release Delta Percentage** - The larger the percentage the quicker the measured signal must change to exit touch once the Delta Snapshot has been taken.

The Release UI settings are interdependent and must be adjusted together. For example, increasing the Release Delta Percentage while simultaneously making the Activation LTA Filter slower may have no effect on how fast the counts must change to exit touch, but would affect how stable the touch condition must be before the Delta Snapshot is taken.

7.4 Low Power Considerations

Typically, the sampling period is increased in low-power modes. Increasing the sampling period will slow the on-chip filtering. Therefore, a separate filter beta value is required for the Activation LTA filter in low-power modes. The low power filter beta must be less than the normal power filter beta, such that the filter behaves similarly in both normal and low power modes.

Typically, Azoteq devices implementing software filtering will provide a low-power filter beta option, and will select the appropriate beta value based on the current power mode.

8 Wear UI

The *WearUI* is an alternative solution to the Follow UI described in Section 6. As such, Wear UI also aims to stabilise the effects of environmental change by monitoring not only the capacitive signal but also the rate of change in the signal. Note that the *WearUI* platform is exclusive to the IQS7223C IC, and more specifically, CH0 on the IC.

8.1 Typical Applications

As the name suggests, Wear UI is aimed at wear detection in portable devices. However, the UI performs best in wear applications that are the least susceptible to sudden signal spikes when worn (explained further in Section 8.2). Therefore, Wear UI may be used for:

- Over-ear headphones
- On-ear headphones
- TWS

It is *not* recommended for use in:

- Smartwatches
- Smart glasses

For smartwatches and smart glasses, it is best to use *Follow UI*.

8.2 Flow Diagram & Description

Figures 8.1 and 8.2 show simplified flow diagrams of the Wear UI system. The process of entering a wear state is illustrated in Figure 8.1, which indicates the UI's focus on both the capacitive signal, indicated by 'CH0 ACTIVATION', and the rate of change denoted by 'CH0 MOVEMENT'. If CH0 is in activation and no movement is detected, the wear flag is set. However, should movement occur during the process of entering wear, the wear flag will only not be set, and will instead be delayed until the capacitive signal settles (no movement).

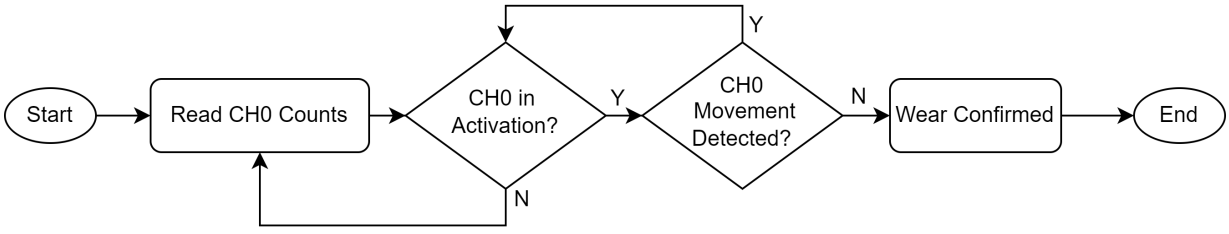


Figure 8.1: Flow chart of entering the wear state with Wear UI

When the device is already in a wear state, Figure 8.2 is applicable. Each time the counts for CH0 are sampled, the IC checks movement. If no movement is detected, the LTA follows the change in counts. This is the distinguishing ability of the Wear UI - it can actively adjust the LTA while in a wear state using only one channel. Thus, when temperature drift occurs, which is associated with a slow change in counts (low movement), the LTA follows such that the original capacitive signal is maintained. Conversely, when sufficiently large movement is detected, which is typically associated with user interaction with the device, the LTA will halt in preparation for a wear release. However, it will only release if after halting the LTA it is found that CH0 is not in activation anymore.

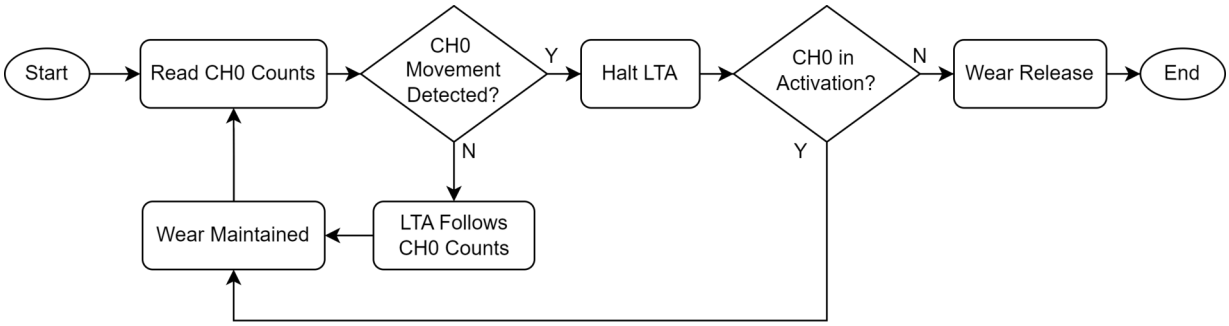


Figure 8.2: Flow chart of maintaining wear and exiting wear

Figure 8.3 shows another illustration of the Wear UI system in the form of the counts plotted over time, and is divided into 8 time periods to explain the Wear UI function. Periods one to three correspond to the flow chart in Figure 8.1, where the device goes from a 'no wear' state to a 'wear' state. In time slot one, the device is idle and CH0 ≈ CH0 LTA. During time slot two, the device is being put into a wear state, but the wear flag is not yet set because there is still fast movement in the CH0 counts signal. The wear flag is only set in period three, where CH0 is in activation and the fast movement has disappeared.

Periods four to eight correspond to the flow chart in Figure 8.2. In period four, slow movement (of signal drift) occurs in the CH0 counts. However, because this is slow movement and not fast movement, the Wear UI will adjust the CH0 LTA to keep the counts delta between CH0 and its LTA, thus preserving the capacitive wear signal. A movement threshold is used by Wear UI to distinguish between slow and fast movement (explained in Section 8.4). Then in period five, fast movement occurs for a brief movement, leading to the LTA being halted. After CH0 stabilises, the LTA is again able to adjust to the slow movement in period six. In period seven, as the user removes the devices, the fast movement of CH0 halts the LTA, allowing CH0 to decrease to its activation threshold. Note that the wear state is maintained here since CH0 is still in activation despite the fast movement. Finally, in period eight, CH0 recovers to the level of the LTA below the activation threshold, and the movement stabilises. This is then registered as a 'no wear' state.

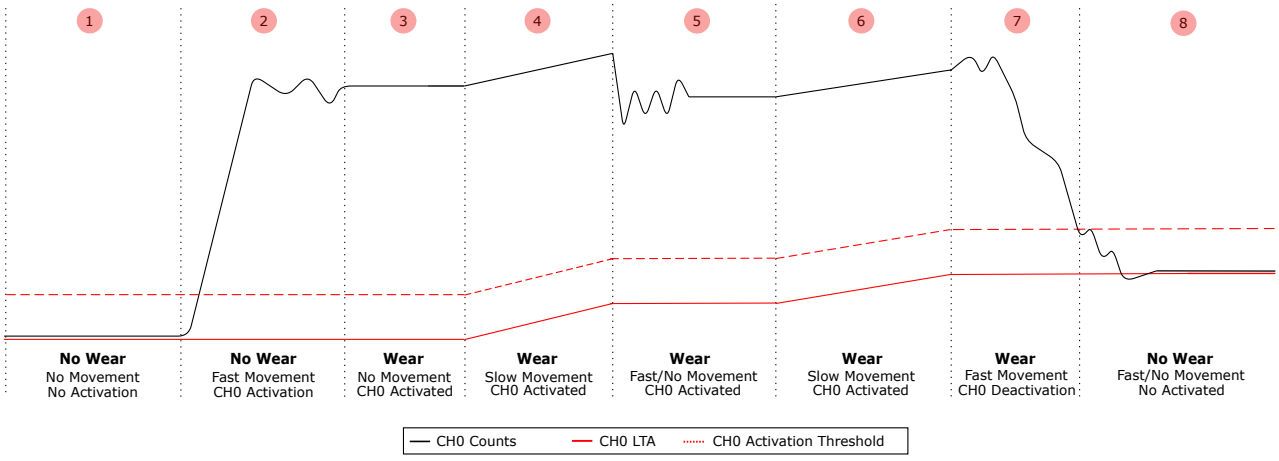


Figure 8.3: Plot of the typical CH0 response for wear detection

Recall from Section 8.2 that the Wear UI is susceptible to signal spikes. Examples of signal spikes

are shown in periods two and seven in Figure 8.3, and is defined as a fast and big change in CH0 counts. When using Wear UI, it is assumed that these spikes only occur when the user interacts with the device, that is, puts the device on or takes it off. However, in some applications, these spikes may occur for other reasons. For example, if the application is a smartwatch and the user performs vigorous exercise, these spikes will commonly occur while the device is being worn, which may be misinterpreted by the Wear UI as a possible release event. Thus, the LTA might be halted when it should have been following to maintain the capacitive signal and if this happens frequently, it will lead to a false 'no wear' state. For this reason, the use of Wear UI should be reserved for applications where random signal spikes are less likely, such as firm-fitting over-ear headphones.

8.3 Layout And Design Configuration

The Wear UI is compatible with all capacitive sensing methods offered by Azoteq and therefore requires no special layout or routing considerations. The general best practice design guidelines for capacitive sensing as noted in AZD135 should still be adhered to for optimal performance.

8.4 Key Settings

Figure 8.4 shows the Wear UI settings dialogue box. The most important settings are indicated by the red blocks and are discussed below. The other settings shown here are the typical settings to be used for most applications.

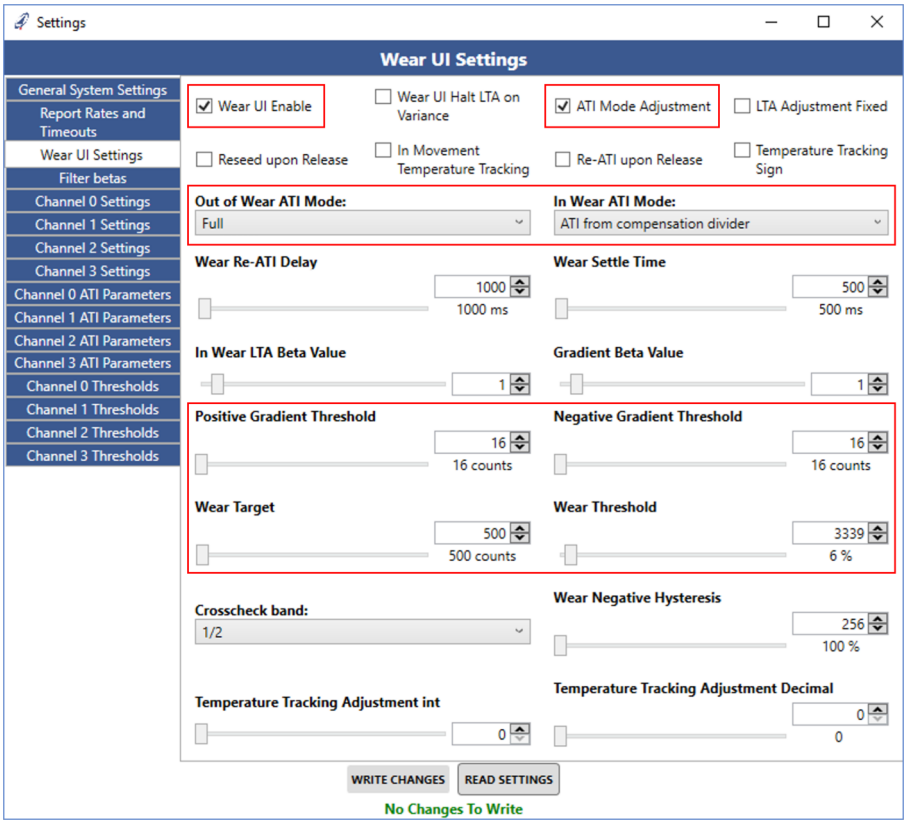


Figure 8.4: Key settings for Wear UI

The following key settings need to be considered:

- **Wear UI Enable**. This check box must be selected for the UI to function.

- **ATI Mode Adjustment**. The Wear UI allows for custom ATI settings. More specifically, it is possible to have different ATI modes for the 'no wear' and 'wear' states. Refer to the next item for more information.
- **ATI Modes**. If *ATI Mode Adjustment* is enabled, the 'no wear' ATI mode can be set, which will override the ATI mode set in the channel settings as long as the UI is enabled. Also, the device will be allowed to ATI while in wear. For the latter, it is recommended to use *ATI from compensation divider*.
- **Positive Gradient Threshold**. This is the threshold value used to distinguish between fast and slow movement as shown in Figure 8.3. Here it is shown to be 16 counts. This means that if CH0 increases by more than 16 counts from one sample to the next, it is regarded as *fast movement*. If it is less than 16 counts, it is *slow movement*. This threshold value will be unique for every application, and should be determined by observing the gradient plot in the SCOPE view of the IQS7223C GUI.
- **Negative Gradient Threshold**. This performs the same function as the *Positive Gradient Threshold*, but is applicable for decreases in CH0 counts. It is typical to have this set equal to the Positive Gradient Threshold.
- **Wear Target**. This is the same as the ATI target found in the Channel 0 settings. If Wear UI is active, the Wear Target overrides the ATI target.
- **Wear Threshold**. Acts as the activation threshold as shown in Figure 8.3, and is the percentage deviation required by CH0 counts from its LTA to go into activation. This value must also uniquely be determined for each application through testing.

9 Power-on Detection Process

9.1 Introduction

This section is dedicated to the design and system explanations for the power-on detection technology. We begin by defining power-on detection within the scope of the discipline:

Power-on detection is the ability to perform wear detection directly after powering up, where the wear state was initiated during the period when the device was powered off.

A simplified illustration of the concept is shown in Figure 9.1.

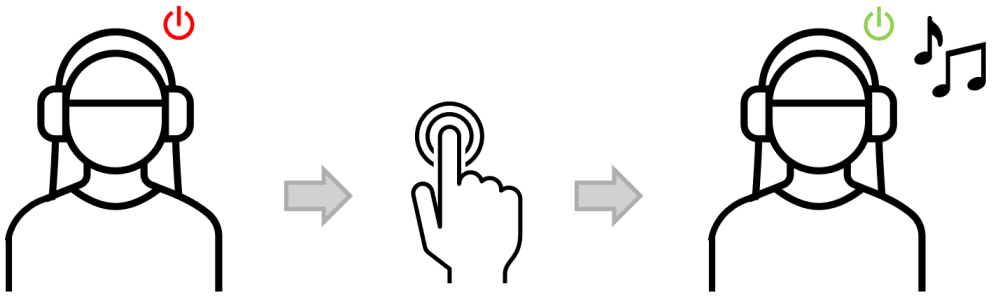


Figure 9.1: Simple illustration of power-on detection with headphones

9.2 Typical Application

As an example, consider the headphone application for wear detection. Assume that when the headphone is turned off, the IQS IC is also turned off. Now there exist two main scenarios during which wear detection must occur. The first is when the user powers on the headphones, thus also powering on the IQS IC, and then places the headphones on their head during which the wear detection will occur. The second scenario is when the user places the headphones on their head *before* powering on.

In the case of the first scenario, the wear state will be successfully detected using the existing algorithms on the IC, specifically the ATI system. This is because when the device is powered on, an ATI procedure can be quickly executed to calibrate the device for the no-wear state, thus allowing timely detection of the wear state that follows.

In the second scenario, the device is already in wear state when it is powered on. So if the device follows the ATI procedure, the resulting device configuration will calibrate to the wear state, and not to the no-wear state as in scenario 1, which means the wear state will not be detected. To solve this problem, Azoteq recommends two possible approaches, explained in the next section.

9.3 Power-on Detection Algorithm

9.3.1 Approach 1 - IQS Always-on Mode

The first approach, namely the *IQS Always-on Mode*, requires simply that the IQS chip remains powered on, even when the MCU is powered off or in a deep sleep mode. This ensures that the IQS device always keeps track of environmental conditions, and as such correctly identifies a wear state, regardless of *when* the MCU itself is powered on.

This approach is made feasible by the ultra-low power (ULP) mode available on the IQS chips, which leads to minimal battery drainage when the device is not in use. Furthermore, some IQS devices feature an *event mode* which further assists in minimising power consumption. For more details on these features, refer to the relevant IQS chip datasheet.

The UIs that are compatible with this approach are:

1. DYCAL™ 2
2. Follow UI
3. Release UI

Given the fact that the IQS does not switch off, this approach may be infeasible for applications with small batteries, such as smartwatches and TWS earbuds. However, this may be suitable for applications such as headphones, which have comparatively large batteries.

Lastly, it is important to note that if the device gets stuck in a given state, the IQS will rely on external indicators to clear the state, such as a charger connection. This 'state reset' routine then needs to be added to the main MCU.

9.3.2 Approach 2 - State Assumption

If it is not possible to follow the first approach, a second approach namely *state assumption* may be used. The state assumption approach closely follows the DYCAL 2 algorithm, in that when the device is powered on, a state (for example a wear state) is assumed, which if incorrect, will automatically correct itself. For more information, refer to Section 4.3 as well as Fig. 4.2.

10 Revision History

Release	Date	Comments
v1.0	2024/04/25	Initial release

Contact Information

	USA	Asia	South Africa
Physical Address	11940 Jollyville Suite 120-S Austin TX 78759 USA	Room 501A, Block A T-Share International Centre Taoyuan Road, Nanshan District Shenzhen, Guangdong, PRC	1 Bergsig Avenue Paarl 7646 South Africa
Postal Address	11940 Jollyville Suite 120-S Austin TX 78759 USA	Room 501A, Block A T-Share International Centre Taoyuan Road, Nanshan District Shenzhen, Guangdong, PRC	PO Box 3534 Paarl 7620 South Africa
Tel	+1 512 538 1995	+86 755 8303 5294 ext 808	+27 21 863 0033
Email	info@azoteq.com	info@azoteq.com	info@azoteq.com

Visit www.azoteq.com
for a list of distributors and worldwide representation.

Patents as listed on www.azoteq.com/patents-trademarks/ may relate to the device or usage of the device.