Application Report

LDC211x and LDC3114 Internal Algorithm Functionality

ABSTRACT

This application note details the operation and usage of the LDC2112, LDC2114, and LDC3114 internal algorithms. It covers what conditions are addressed by a given algorithm and the parameters that control each algorithm. With the information provided here, the system designer can obtain the optimum performance out of a given design.

Table of Contents

1 Introduction......2
2 Scan Rate and Sampling Interval......2
2.1 Low Power Mode and Normal Power Mode......3
2.2 Button Sequencing and Error Handling......3
3 Data Polarity and Timeout......5
3.1 Button Timeout......5
4 Internal Algorithms Overview......6
5 Baseline Tracking......8
5.1 Baseline Increment......8
5.2 Baseline Tracking Reset......10
5.3 Button Actuation Time......11
5.4 BTPAUSE......12
5.5 Fast Tracking Factor......13
6 Gain, Hysteresis, and Threshold......14
6.1 Threshold and Hysteresis......15
7 Multi-Button Algorithms......15
7.1 Max Win......15
7.2 Anti-Common Mode......17
7.3 Anti-Twist Factor......18
7.4 Anti-Deform Factor......20
8 Summary......21
9 Revision History......21

List of Figures

Figure 2-1. Sample Interval and Scan Rate......2
Figure 2-2. Sampling Flow Chart......4
Figure 3-1. DPOLx functionality with Inductive and Capacitive Sensing......5
Figure 4-1. Algorithm Sequence......7
Figure 5-1. Baseline Increment produces a decaying output code......8
Figure 5-2. Effect of different values of Baseline Increment......9
Figure 5-3. Baseline Tracking Reset during button press......10
Figure 5-4. Effect of Baseline Increment on Minimum Actuation time......11
Figure 5-5. BTPAUSE Functionality......12
Figure 5-6. Effect of Fast Tracing on Output Code......13
Figure 6-1. LDC211x and LDC3114 Gain Factor vs. Programmed Register Field Setting......14
Figure 6-2. Compare Thresholds based on HYST setting......15
Figure 7-1. Max-Win functionality......16
Figure 7-2. Anti-Common Functionality......17
Figure 7-3. Example Twist Force Applied to Device......18
Figure 7-4. Anti-Twist Functionality......19
Figure 7-5. Deformation Can Cause Positive or Negative Offsets......20
Figure 7-6. Anti-Deform Functionality......21

List of Tables

Table 2-1. LDC211x and LDC3114 Sample Control....3
Table 4-1. LDC211x and LDC3114 Internal Algorithms....6
Table 5-1. Baseline Tracking Clear and Reset....10

Trademarks

All trademarks are the property of their respective owners.

1 Introduction

The Texas Instruments LDC211x and LDC3114 devices can monitor changes in a sensor resonant frequency to determine whether a stimuli corresponds to a user interaction such as a button press. Refer to the *Inductive Touch System Design Guide for HMI Button Applications* application report for details on how to construct a mechanical system suitable for inductive sensing button applications, sensor design guidance, and an overview of device configuration. The LDC211x and LDC3114 use several internal algorithms to compensate for environmental shifts, determine whether the sensor signal change corresponds to a button press, and also to correct for mechanical cross-talk between multiple buttons. This document details the operation and configuration of the algorithms.

2 Scan Rate and Sampling Interval

Once the LDC211x or LDC3114 powers on, it continuously samples the enabled channels and returns to as shown in Figure 2-1. The active channels, sample time interval, and button scan rate are all configurable parameters. Refer to Table 2-1 for a listing of the device parameters that control the Scan Rate and Sample Interval. A higher Scan Rate produces a more responsive interface, but generally at the cost of higher average supply current consumption. Note that the scan rate can vary by up to ±30% from the nominal value across devices.

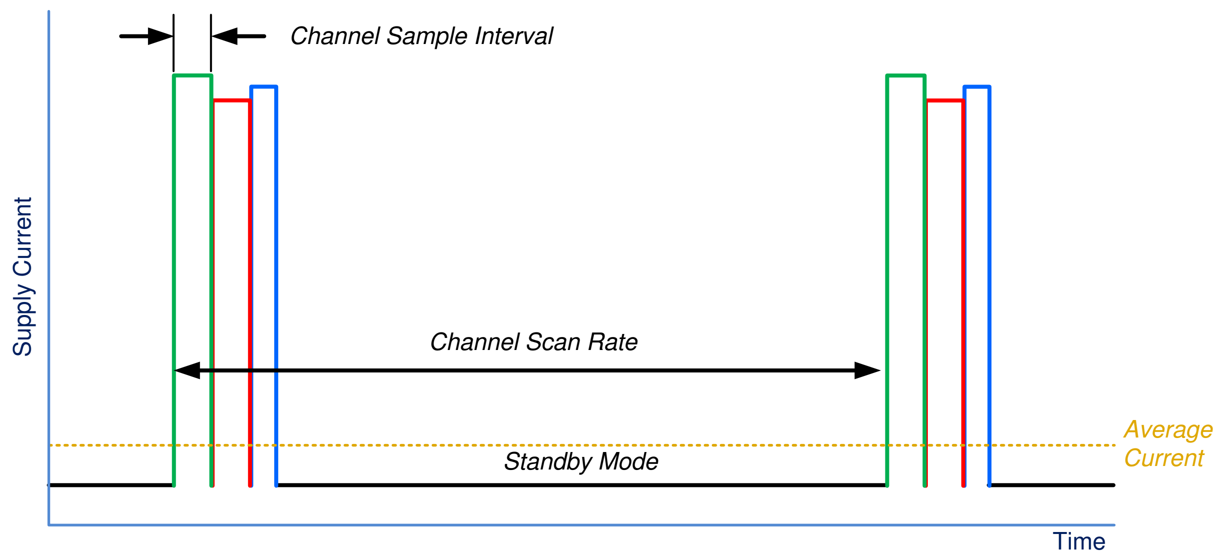

Figure 2-1. Sample Interval and Scan Rate

Table 2-1. LDC211x and LDC3114 Sample Control

Device Parameter	Configuration Effect	Details
SENCYCx	Button Sample Interval for channel x	Each channel has dedicated setting. Sensor frequency is used to determine proper setting.
CNTSCx	Button Sample Interval for channel x	Each channel has dedicated setting. Sensor frequency, LCDIV, and SENCYCx is used to determine proper setting
LCDIV	Button Sample interval for all channels	Common setting for all channels
LPWRB pin	Button Scan Interval	When set High, device samples based on NPSR value. When set Low, device sample based on LPSR value.
NPSR	Button Scan Interval	When LPWRB pin is High, this field sets the device scan rate from 10 sps to 80 sps.
LPSR	Button Scan Interval	When LPWRB pin is Low, this field sets the device scan rate from 0.625 sps to 5 sps.

The Button Sample Interval is a function of the SENCYCx, LCDIV, and the Sensor Oscillation Frequency. The sample time, t_{SAMPLE} is set by:

$$t_{SAMPLE} = 128 \times (SENCYCx+1)\times 2^{LCDIV} \div f_{SENSOR\ x}$$

The recommended sample interval is 1 ms, as this provides a good balance on noise vs. supply current. Shorter sample intervals may have reduced SNR. Variations in sensor frequency greater than 2.5% may need to use different settings for SENCYCx. If the sensor frequency increases significantly and the SENCYCx & LCDIV settings are not changed, then a reduction in sensitivity can occur. Alternatively, if the sensor frequency decreases by ~30% from the configured frequency, it is possible for internal counters in the device to over-range, resulting in improper operation.

2.1 Low Power Mode and Normal Power Mode

The LDC211x and the LDC3114 operate in one of two modes – Low Power Mode and Normal Power Mode. In Low Power Mode, the maximum supported scan rate is from 0.625 to 5 sps. In Normal Power Mode, the scan rate is 10 to 80 sps. The primary difference between the two modes is the Baseline Increment, which is discussed below. The buttons enabled in Low Power Mode must be a subset of the buttons enabled in Normal Power Mode.

2.2 Button Sequencing and Error Handling

The LDC211x and LDC3114 start sampling channel 0 and check to see if it is enabled. If the channel is enabled, it starts the sensor oscillation and checks whether the sensor is oscillating. If so, it measures the sensor frequency, otherwise the device flags an error. It then increments to the next channel, and repeats the process. Figure 2-2 represents this sequence as a flowchart.

If the sensor cannot oscillate in a stable manner, the device flags the error in STATUS:LC_WD and continues onto the next channel. Potential sources of a LC_WD error include the sensor frequency exceeds the sensor maximum frequency of 30 MHz, sensor frequency is below the minimum specified sensor frequency, or the sensor R_P is less than the minimum supported R_P.

The LDC211x and LDC3114 devices are intended to maintain operation under a wide range of conditions, and may not generate an error condition even if the sensor is not within the specified operating region. Refer to Figure 4-1 for details on the Execute Algorithms operation shown in Figure 2-2.

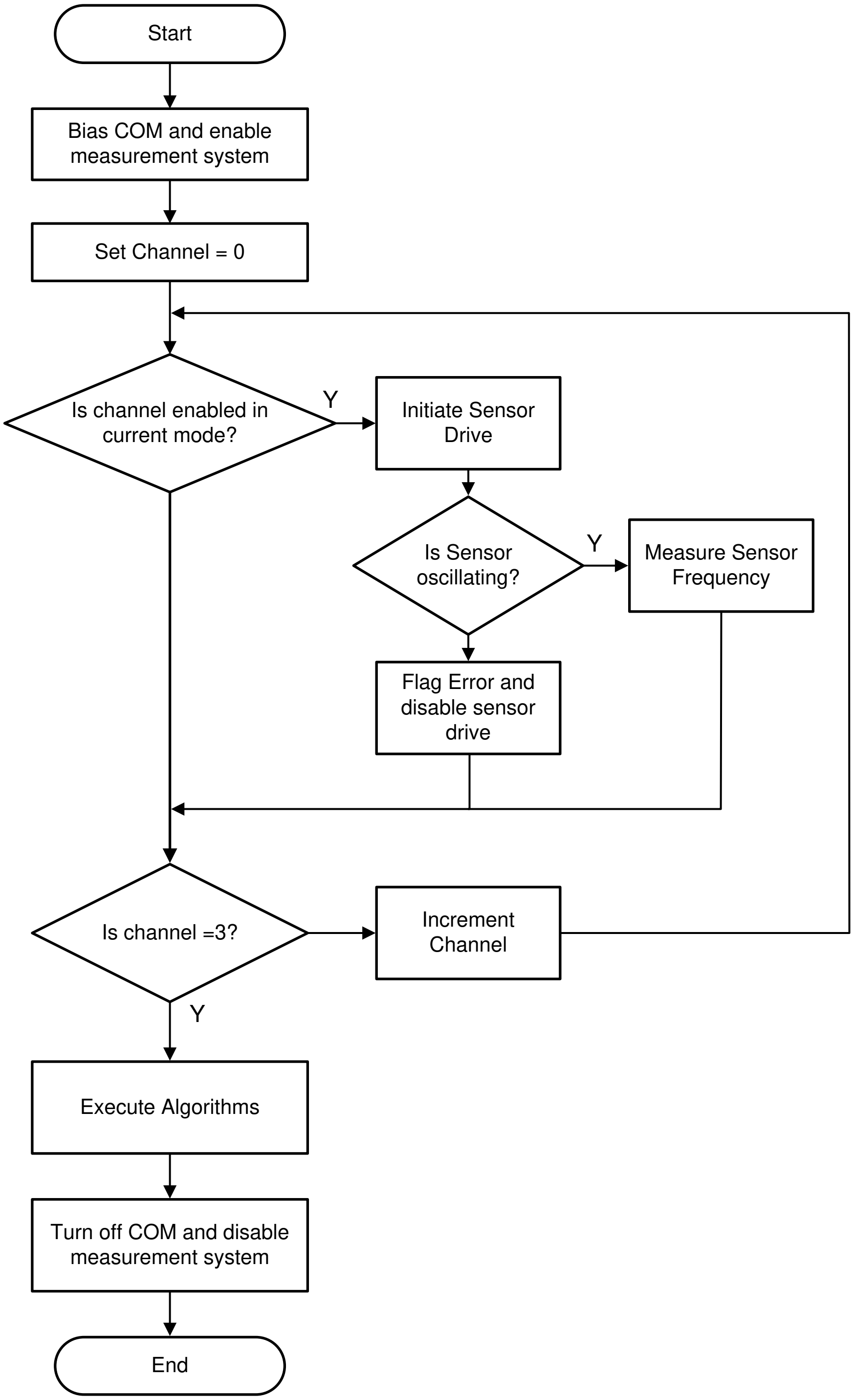

Figure 2-2. Sampling Flow Chart

3 Data Polarity and Timeout

The LDC211x and LDC3114 can detect positive or negative frequency shifts as button events. This is controlled per channel by the DPOLx setting, which adjusts the polarity of data processing. Figure 3-1 shows how this can be configured to support either inductive or capacitive sensing by the same device, or even a combination where some buttons operate as inductive sensors and the others operate as capacitive sensors. For a capacitive sensor, setting DPOLx to 0 inverts the sign of the internal data processing. This results in a signal which operates in the same sign as an inductive sensor, and so all other internal algorithms operate consistently.

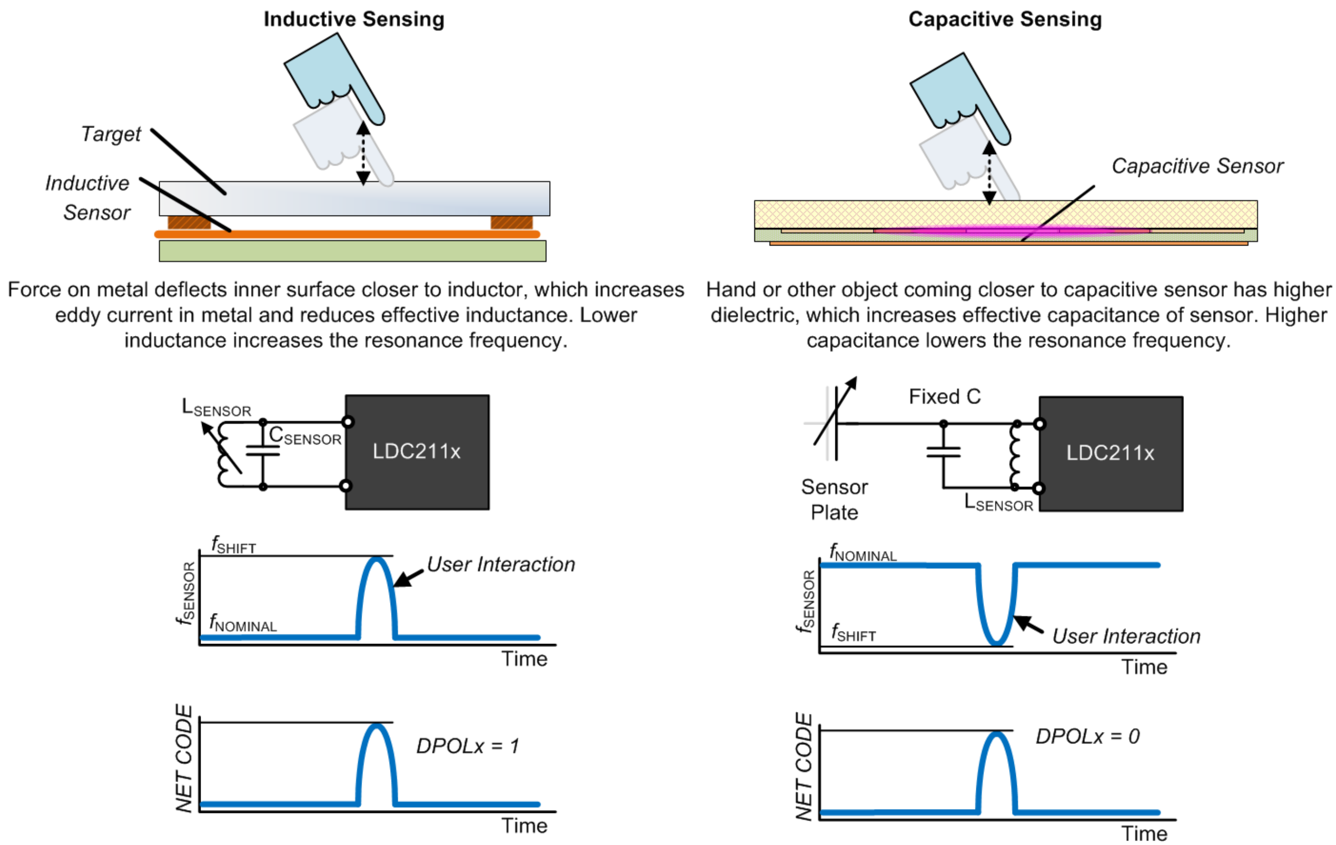

Figure 3-1. DPOLx functionality with Inductive and Capacitive Sensing

3.1 Button Timeout

The maximum supported continuous button actuation is ~50 seconds. If the button is asserted for a longer time interval, the device internal state machine automatically resets operation and the baseline tracking value automatically resets, which deasserts the button. When a timeout occurs, the device will flag the event by setting STATUS:TIMEOUT to 1.

For the LDC3114, the baseline tracking algorithm may be disabled by using the device in raw data access mode.

4 Internal Algorithms Overview

The LDC211x and LDC3114 devices take the output measurements for each channel and apply a sequence of several algorithms to the conversion results. Some of the algorithms are optional, and some can be configured per-channel. Table 4-1 below lists the internal algorithms and the configuration options applicable for each algorithm.

Table 4-1. LDC211x and LDC3114 Internal Algorithms

Algorithm	Always Applied or Optional?	Configuration	Per Channel	Functionality
Data Polarity	Always	DPOLx	Y	Select Inductive or Capacitive sensing operation
Baseline Tracking	Always	NPBI/LPBI	N	Compensates for environmental shifts in inductance and capacitance.
Fast Tracking	Optional	FTF	Y	Provides faster recovery for negative swings in output code values.
Gain	Always	GAINx	Y	Adjusts sensitivity of channel. Scales between 1x and 232x in 64 settings; average step delta is 9%.
Threshold Compare	Always	N/A	N	Centered at 128. Effectively adjusted by use of GAINx.
Hysteresis	Always	HYST	N	Sets button actuation or deactivation thresholds. Centered at fixed Threshold Compare level of 128. Adjustable from 0 to 60 in steps of 4.
Baseline Tracking Pause	Optional	BTPAUSEx	Y	Disables Baseline Tracking when OUTx value exceeds THRESHOLD+HYSTERSIS. Baseline Tracking is disabled in raw data access mode of LDC3114.
Max Win	Optional	MAXWINx	Y	Compares OUTx across selected channels and deasserts OUTx signal for all but the highest value channel.
Anti-Common	Optional	ANTICOMx	Y	Suppress common-mode signal present on multiple channels which are due to mechanical cross-talk.
Anti-Deform	Optional	ANTIDFORMx	Y	Compensate the baseline tracking for non-ideal recovery from mechanical stresses.
Anti-Twist	Optional	ANTITWIST	N	Suppress inverse signals present on multiple channels which are due to mechanical cross-talk.

The various algorithms are applied in the sequence shown in Figure 4-1. For algorithms which are optional, if the algorithm is not enabled, then the channel data is not modified in by block.

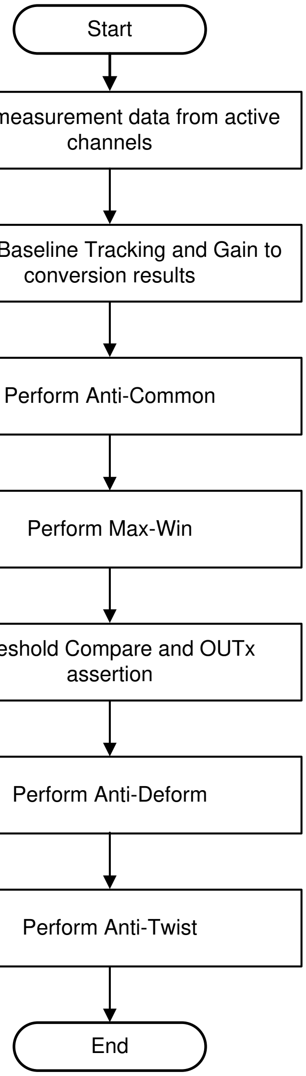

Figure 4-1. Algorithm Sequence

5 Baseline Tracking

5.1 Baseline Increment

Button detection applications can be effectively implemented by looking at a high-pass filtered version of the input signal, due to the typical stimuli. Refer to Figure 5-1. In the LDC211x and LCD3114 devices, this high-pass filtering is implemented using a subtracted baseline tracking which outputs the shift from the nominal code. This baseline tracking is designed to ignore slow changes in the output code, as these are generally due to environmental shifts which occur over the span of seconds, while a user interaction typically occurs on the order of 50 ms.

The LDC211x and LDC3114 use a linear offset approach, which always drives the output code towards 0.

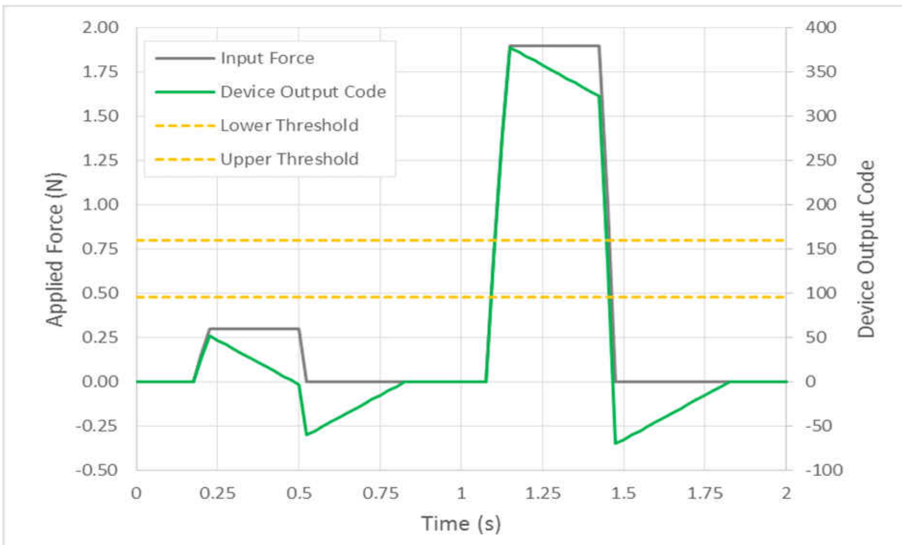

Figure 5-1. Baseline Increment produces a decaying output code

Each channel of the LDC211x has its own Baseline value, but the baseline increment is the same for all channels. The internal baseline setting of the LDC211x and LDC3114 is updated for each sample, using the following pseudo-code:

```
If raw_data[channel] > baseline[channel] then
baseline[channel] = baseline[channel] + base_increment
iIf raw_data[channel] < baseline[channel] then
baseline[channel] = baseline[channel] - base_increment
```

The net code is then calculated from the raw code with:

```
net[channel] = raw_data[channel] - baseline[channel]
```

Where the net code is the output code of the device. Note that this net value will be scaled by the Gain_Factorx and modified by other algorithms before storage in the output register. The baseline increment value can be adjusted across a range of 1 to 128, in 8 settings. A higher setting will bring the output code to 0 in less samples, as displayed in Figure 5-2.

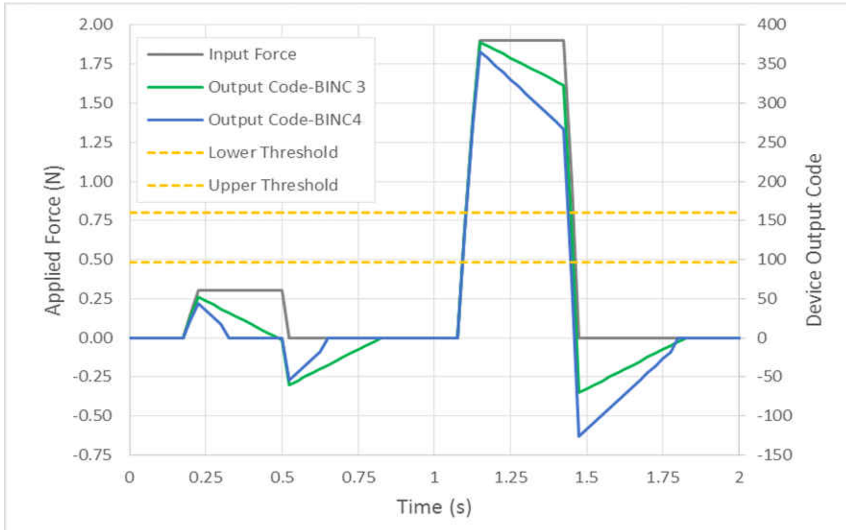

Figure 5-2. Effect of different values of Baseline Increment

The baseline value is reset whenever the device changes modes (e.g. from Config Mode to Low Power Mode, or from Low Power Mode to normal power mode). Note that baseline tracking is applied for every sample. Faster sample rates have an effectively higher baseline tracking rate per unit time. The Baseline Increment for Normal Power Mode is:

Baseline Increment per Sample = Gain_Factorx × 1.827 ÷ $2^{(7\text{-NPBI})}$

Where:

- *Gain_Factorx* is the linear gain value selected for the channel based on the GAINx field setting, and
- *NPBI* is the value programmed into the NPBI field

To compensate for the slower sampling interval used in low power mode, the Low Power mode Baseline Increment is effectively 8x larger for the same setting:

Baseline Increment per Sample(Low Power Mode) = Gain_Factorx × 1.827 ÷ $2^{(4\text{-LPBI})}$

Where:

- *Gain_Factorx* is the linear gain value selected for the channel based on the GAINx field setting, and
- *LPBI* is the value programmed into the LPBI field

The effective Baseline Increment in either mode is not restricted to an integer value, as the LDC2114 has higher internal resolution than is represented by the output code.

5.2 Baseline Tracking Reset

The baseline tracking is reset under several conditions. When a baseline tracking is reset, the baseline tracking value is set to the current measured raw_data. If a button is pressed when a baseline tracking reset occurs, the button press will be deasserted. The OUTx pin will immediately be deasserted. A baseline tracking reset can result in reduced sensitivity to subsequent button presses until the baseline tracking returns to 0.

In Figure 5-3, a baseline reset occurs at t=0.6 s. The OUTx button then immediately deasserts. The second button press occurring at t=1.5 s is not detected due to the offset in the baseline tracking. A stronger button press would be detected at t=1.5 s. The third button press at t=2.25 s is detected, as the baseline tracking has returned close enough to 0.

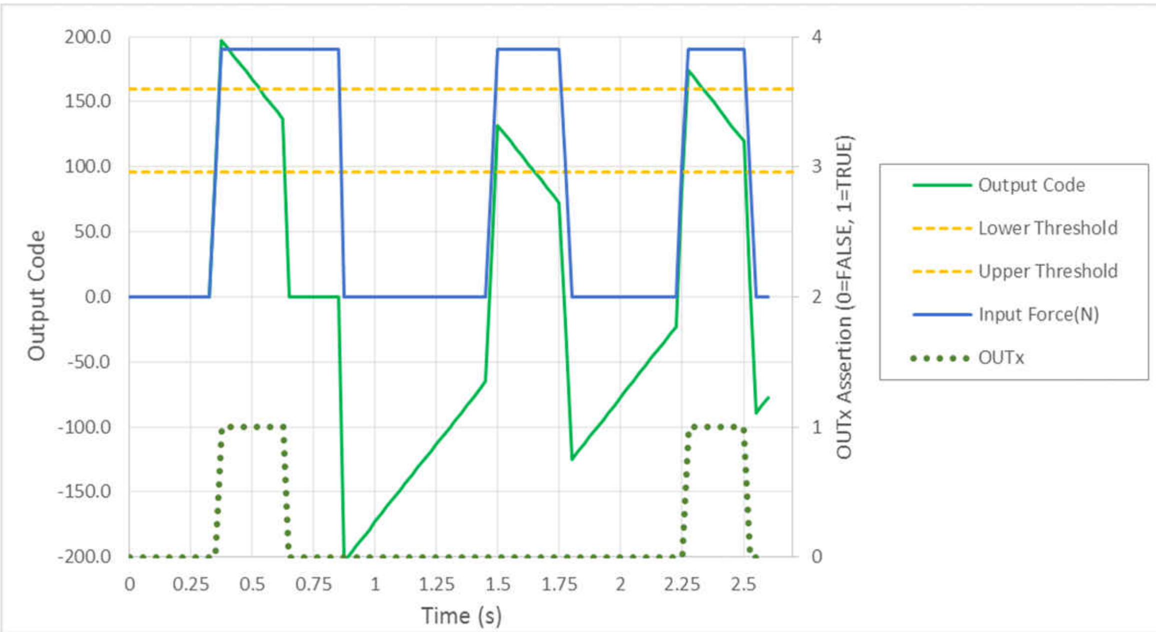

Figure 5-3. Baseline Tracking Reset during button press

Some operations reset the baseline tracking for a subset of channels while other operations reset all channels. Another type of reset is a clear – when the baseline tracking is cleared, a button press will take 2 scan intervals to detect a button press event.

Table 5-1. Baseline Tracking Clear and Reset

Condition	Channels Affected	Comment
LPWRB pin toggle	All channels	Clears baseline. If a button remains pressed during toggling of LPWRB, the button press will be deasserted.
Soft Reset	All channels	Clears baseline
Exit from Config Mode	All channels	Clears baseline
Continuous Button press > 50 sec	All channels	Reset baseline
Button Press detected in Max Win group	Unasserted channels in Max-Win group.	Reset baseline
Button Press detected in Anti-Deform	Unasserted channels in Anti-Deform group.	Reset baseline
Anti-Twist threshold is exceeded	All channels	Reset baseline

5.3 Button Actuation Time

The minimum button actuation time is the hysteresis value divided by the baseline increment. Figure 5-4 shows the output codes and OUTx assertion for the same stimuli with different baseline increment settings. Larger values of baseline increment can effectively “turn-off” a button before the actual stimuli has been removed.

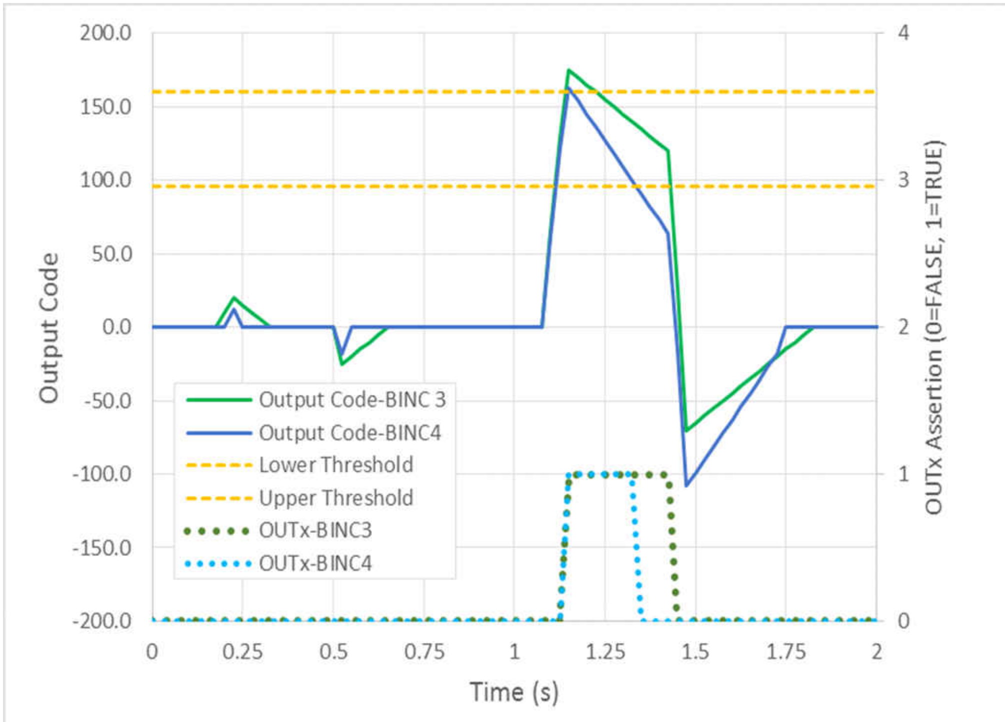

Figure 5-4. Effect of Baseline Increment on Minimum Actuation time

5.4 BTPAUSE

BTPAUSE is an optional adjustment to the baseline tracking, which inhibits the baseline tracking operation for the duration of a button press. This functionality applies to the internal data used by the LDC211x and LDC3114 and not just the OUTx pin operation. This can keep a button actuated for a longer time, and also be useful for some applications which post-process data. BTPAUSE can be enabled on all channels or any subset of channels.

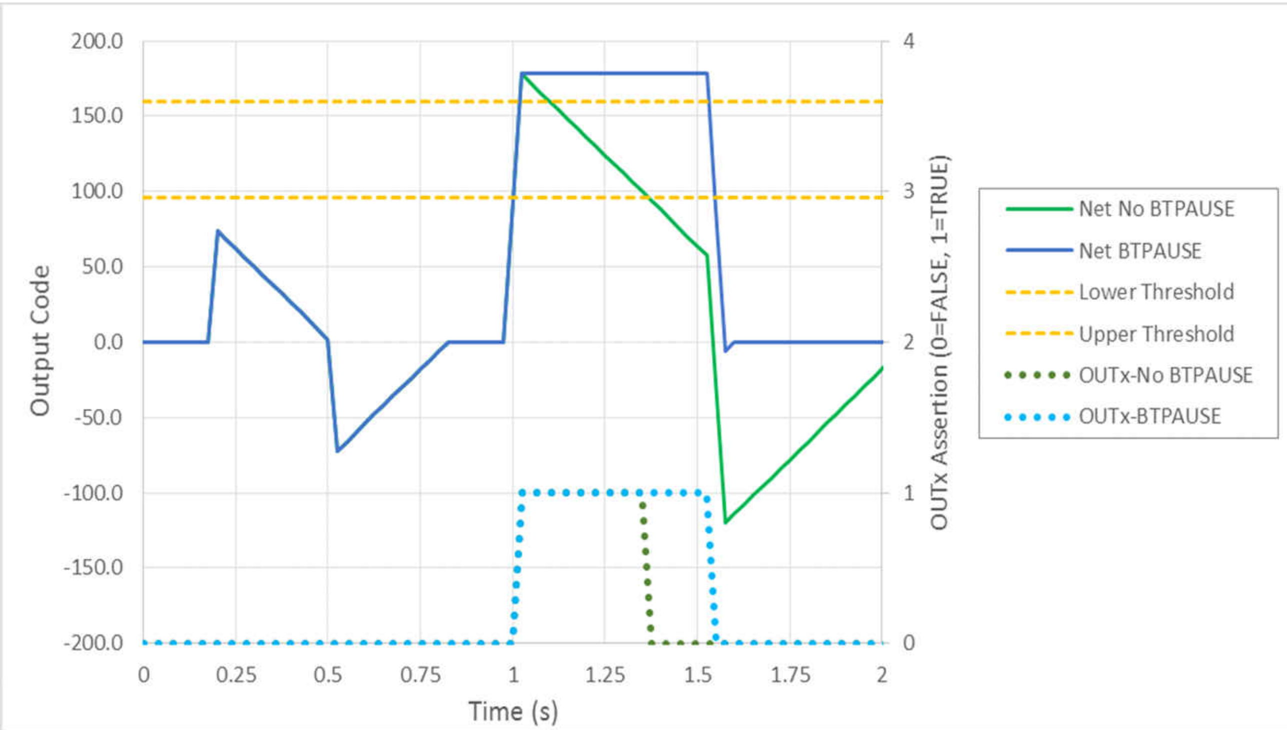

Figure 5-5. BTPAUSE Functionality

5.5 Fast Tracking Factor

The negative swings which occur after a button actuation can reduce sensitivity for multiple sequential button presses. While BTPAUSE is often a more effective algorithm for applications in which this is an issue, Fast Tracking Factor provides an increased Baseline Increment when OUTx is negative. Figure 5-6 compares the operation with and without Fast Tracking Enabled for an example stimuli. Each channel can be set with an independent setting for Fast Tracking – FTFx field can be set between 0 and 3.

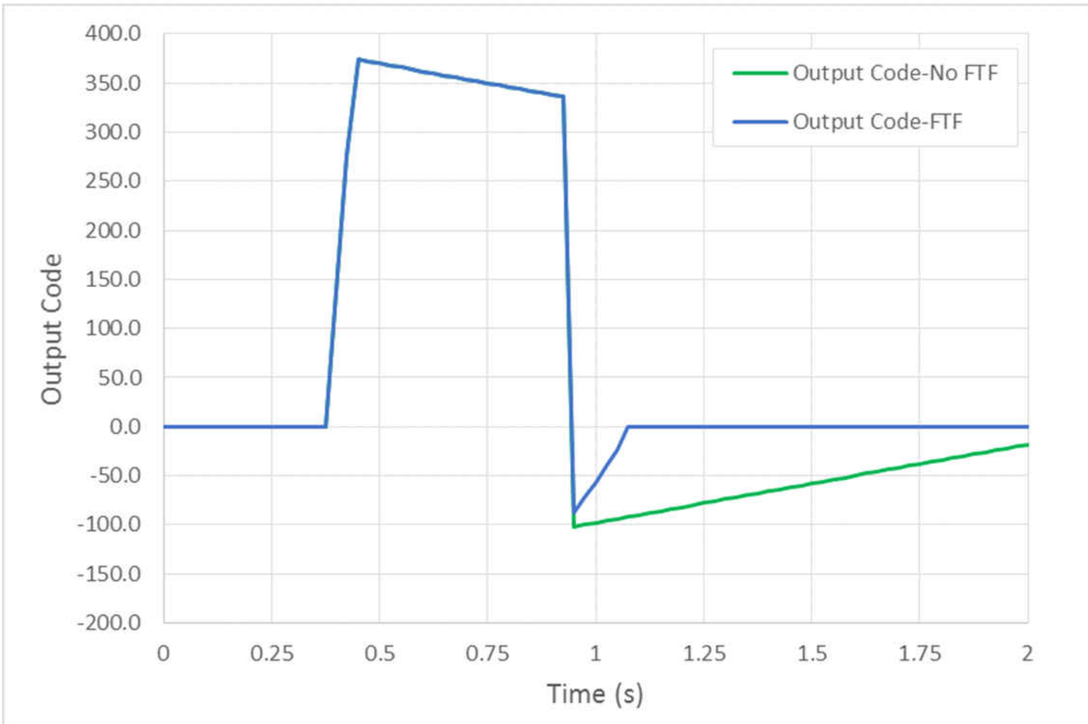

Figure 5-6. Effect of Fast Tracing on Output Code

The following pseudo-code shows the effective change in Baseline Tracking when Fast Tracking Factor is enabled:

```
If raw_data[channel] > baseline[channel] then
baseline[channel] = baseline[channel] + base_increment
If raw_data[channel] < baseline[channel] = FTH) then
baseline[channel] = baseline[channel] - FTF * base_increment
Else If raw_data[channel] < baseline[channel] then
baseline[channel] = baseline[channel] - base_increment
```

The value of FTF is set by:

$FTF = 2^{FTH \times \text{Field Value}}$

Where FTH is a correction to the FTF based on the device gain.

Fast Tracking is a multiplier onto the Baseline tracking setting. It is applied in both Normal Power mode and Low Power mode with the same scaling factor.

6 Gain, Hysteresis, and Threshold

Each channel has a dedicated gain setting (GAINx). The Gain setting can be adjusted from an effective 1x to 232x, in 64 steps. Throughout this document, the value programmed into the register is referred to as GAINx, and the effective gain value that is applied by the algorithm is referred to as the Gain Factorx. The steps are logarithmically spaced, with each increment in the GAINx field increasing the Gain Factor by ~9%. Using a lower Gain Factorx reduces the sensitivity to environmental shifts. The gain is applied to net code, which is the raw measured value with the baseline tracking subtracted.

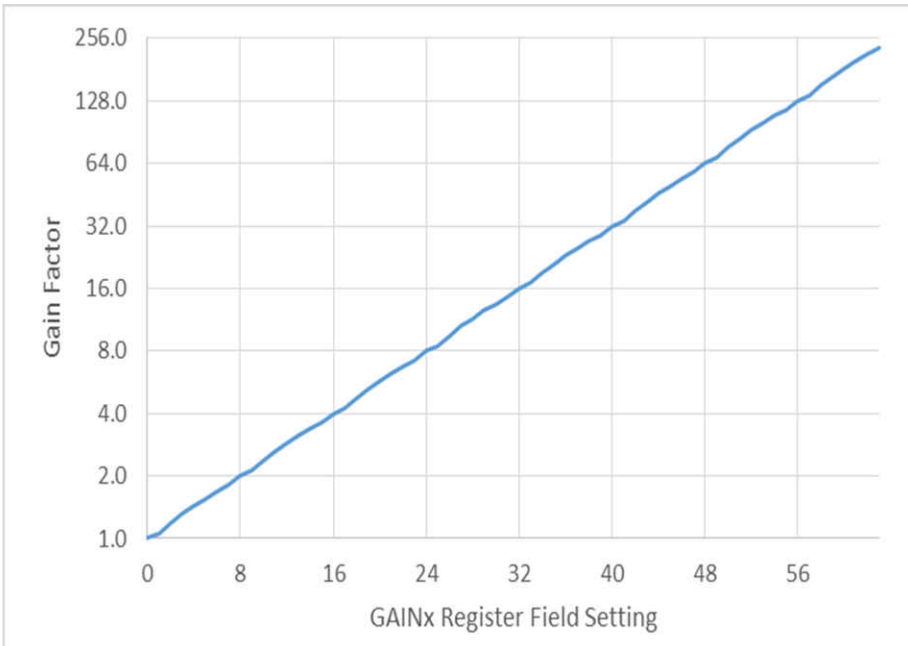

Figure 6-1. LDC211x and LDC3114 Gain Factor vs. Programmed Register Field Setting

The effective Net Code shift to a fast stimuli is:

$NET\_CODEx = 0.013 \times Gain\_Factorx \times \Delta f_{SENSOR}$

Where Δf_{SENSOR} is in PPM

6.1 Threshold and Hysteresis

After applying Gain Factorx, each channel Netx is compared to a fixed threshold of 128+*Hysteresis*. The *Hysteresis* value is set by the HYST register field. The *Hysteresis* value can be set from 0 to 60, in increments of 4. The HYST setting is common to all channels. A larger HYST setting improves the noise immunity and also increases the minimum button actuation time (when BTPAUSEx is not enabled). The GAINx and HYST settings effectively produce a variable Upper and Lower threshold, as seen in

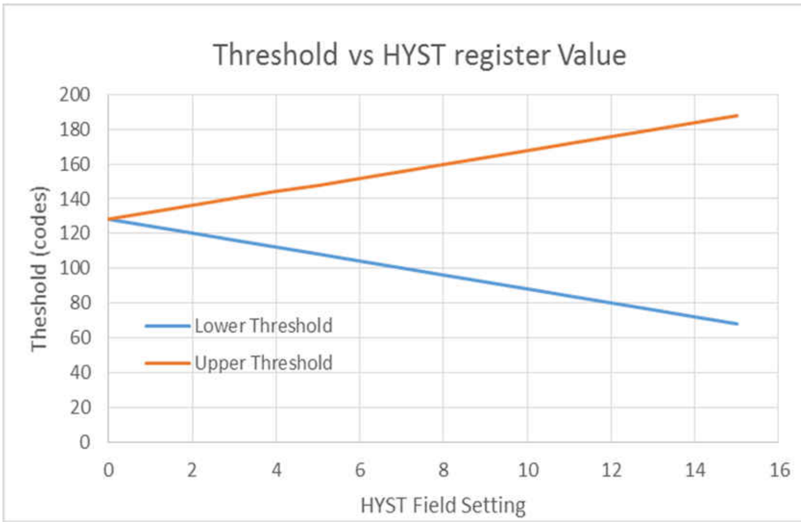

Figure 6-2. Compare Thresholds based on HYST setting

The hysteresis and threshold operation is consistent with standard practice, in that:

- If OUTx is asserted, OUTx deasserts if NETx < Lower Threshold
- If OUTx is deasserted, OUTx asserts if NETx > Upper Threshold

7 Multi-Button Algorithms

The LDC211x and LDC3114 include several algorithms which operate on multiple channels. These algorithm can be used to suppress false button actuation which is caused by mechanical cross-talk between multiple channels. Each of these algorithms can be applied to any subset of buttons, although a set of a single button will not have any effect. In general, it is not recommended to apply multiple algorithms to the same channel, although using one algorithm on one pair of channels and a different algorithm to the other pair of channels with these devices is acceptable.

7.1 Max Win

Max-Win is a useful algorithm which can be used when significant positively-correlated mechanical cross-talk is present in the system. This algorithm can be used on any subset of buttons, but is only effective if there are 2 or more enabled channels. This algorithm behaves identically in both Low Power Mode and Normal Power Mode. Note that this algorithm only modifies the assertion of the OUTx pins and the OUTx field; although the baseline tracking reset it performs will affect subsequent samples.

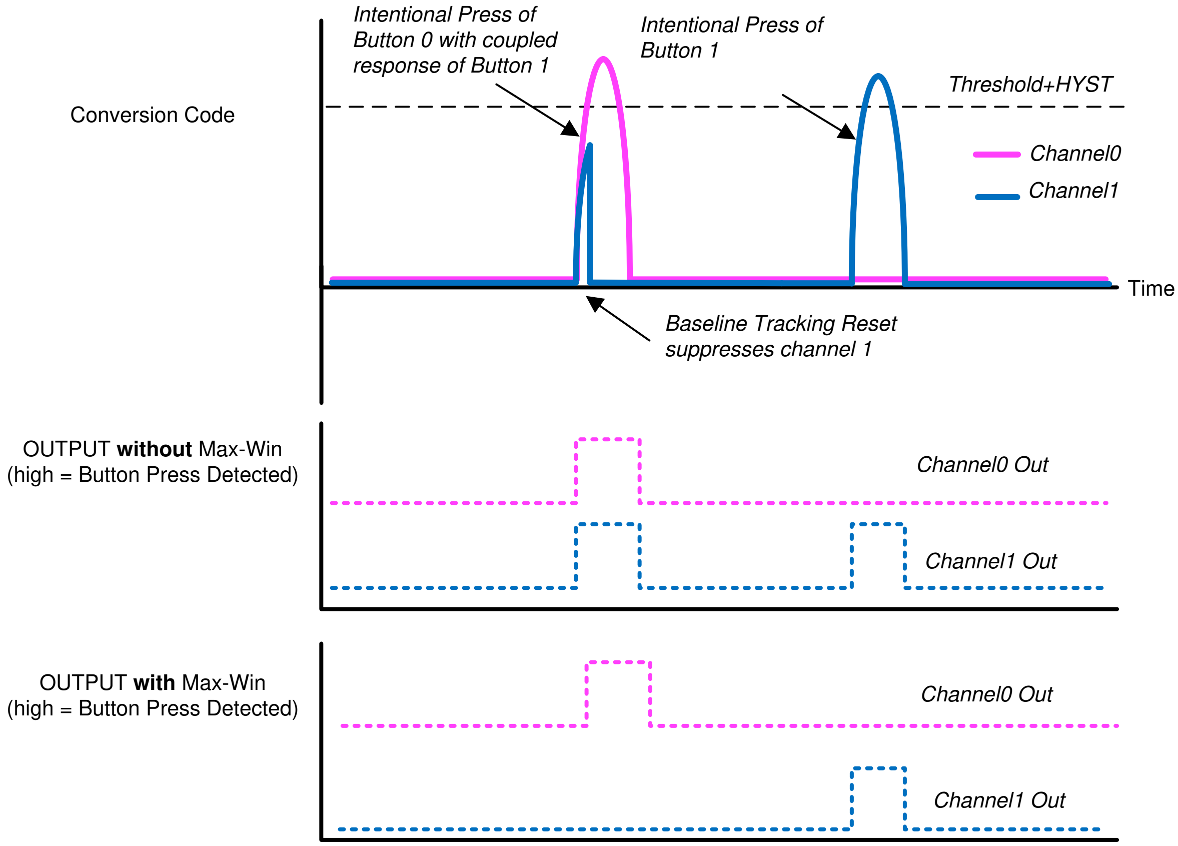

Figure 7-1. Max-Win functionality

To use Max-Win, set the desired MAXWINx fields to 1. As this algorithm uses the post-gain value, adjusting the Gainx value affects the performance of this operation. The algorithm compares all enabled buttons with MAXWINx enabled that have net code exceeding the threshold. In the case of an equal net value across 2 or more channels, the lower indexed channel dominates (e.g. channel 0 will suppress channel 1). Any suppressed channels have their baseline tracking value updated to the current setting.

This algorithm can also be used to inhibit up to 3 channels with a reference sensor which has a higher Gain Factorx. By constructing the reference sensor in a manner which it is sensitive to extreme torque or twist events, false button actuations can be suppressed.

7.2 Anti-Common Mode

Anti-Common Mode corrects for unintended mechanical deflection which is common to multiple buttons. Figure 7-2 provides a comparison between having Anti-Common enabled vs. disabled for an example stimulus.

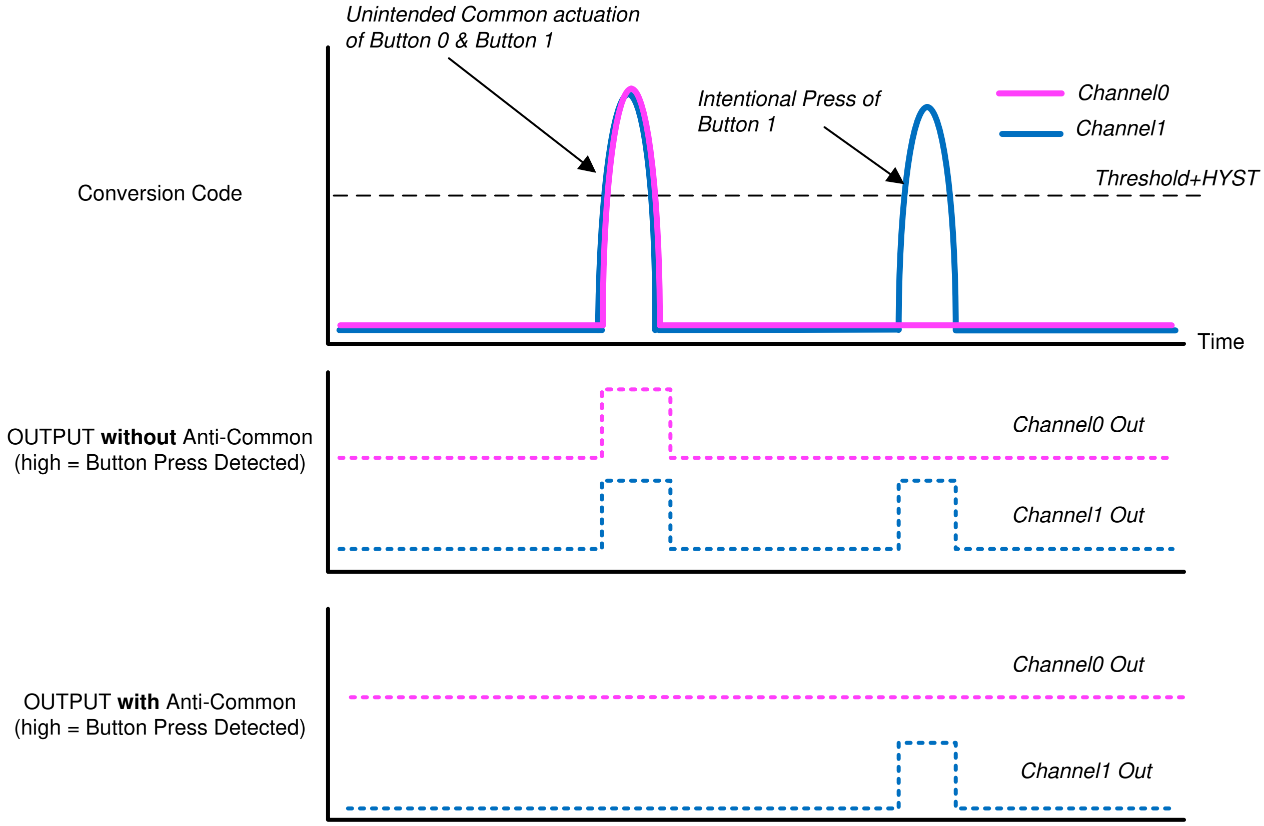

Figure 7-2. Anti-Common Functionality

Buttons can be included in the Anti-Common group by setting the corresponding ANTICOMx field to 1. The default setting for the ANTICOMx field is 0, which disables Anti-Common for corresponding channel. The pseudo-code for this algorithm is equivalent to:

```
ACM_offset = (∑net[channel] )/num_acm_chan

For each (channel with enabled ACM)

net[channel] = net[channel] – ACM_offset

Next channel
```

Note that the Net output value readback from the LDC211x is changed by the use of this algorithm.

7.3 Anti-Twist Factor

Anti-Twist inhibits false button events which are caused by torques and twists on the unit. These forces can produce opposite-phase responses on buttons.

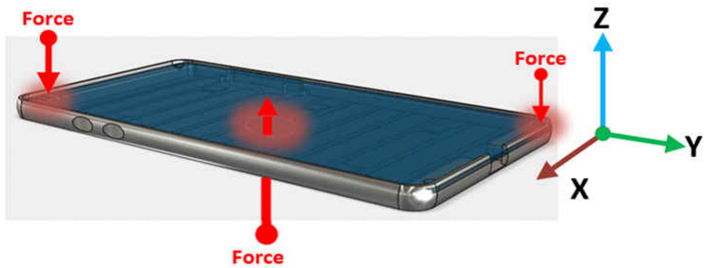

Figure 7-3. Example Twist Force Applied to Device

This functionality is enabled by setting the ANTITWIST register field to any value from 1 to 7; it is not enabled if ANTITWIST is set to 0, which is the default value. When ANTITWIST is enabled, all active buttons will be affected by the processing. The Anti-twist threshold is 4×ANTITWIST, and so it can be set from 4 to 28. Anti-Twist uses the following pseudo-code:

```
If Out[0] or Out[1] or Out[2] or Out[3] is true then
If at least for one channel: net[channel] > AntiTwist and
If at least for one channel: net[channel] < -AntiTwist then
For each active channel
baseline[channel] = raw_data[channel]
Next Channel
```

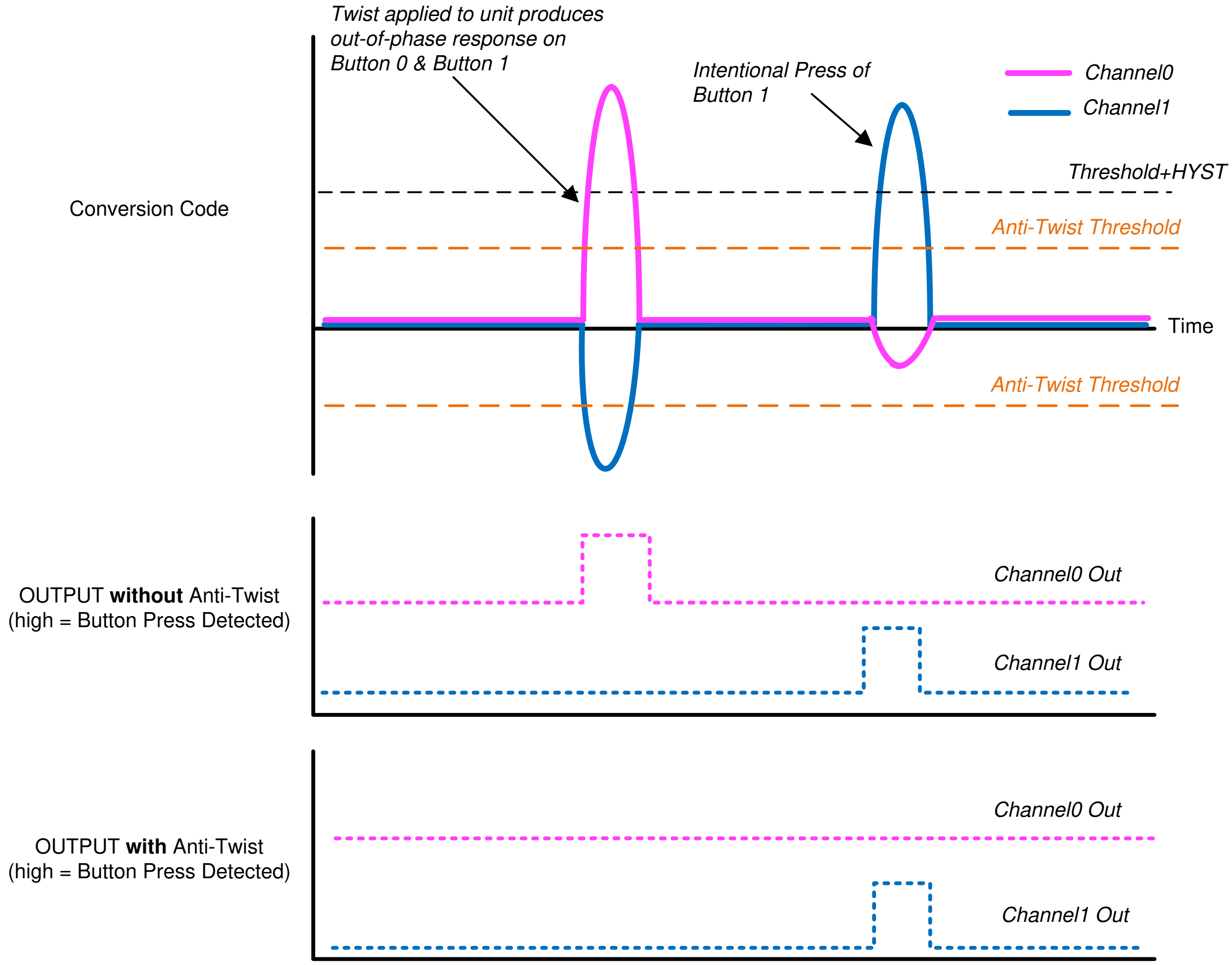

Figure 7-4. Anti-Twist Functionality

Basically, if any channel is found to exceed the Anti-Twist threshold in a positive direction simultaneous with another channel exceeding the Anti-Twist threshold in a negative direction, then any button press detection is disabled. This algorithm does not alter the Net Data in the output registers.

7.4 Anti-Deform Factor

Anti-Deform compensates for common-mode shifts on other buttons when a button is pressed. The common mode shifts are caused by mechanical offsets, and these offsets can either increase the likelihood of a false actuation or significantly increase the required mechanical actuation force.

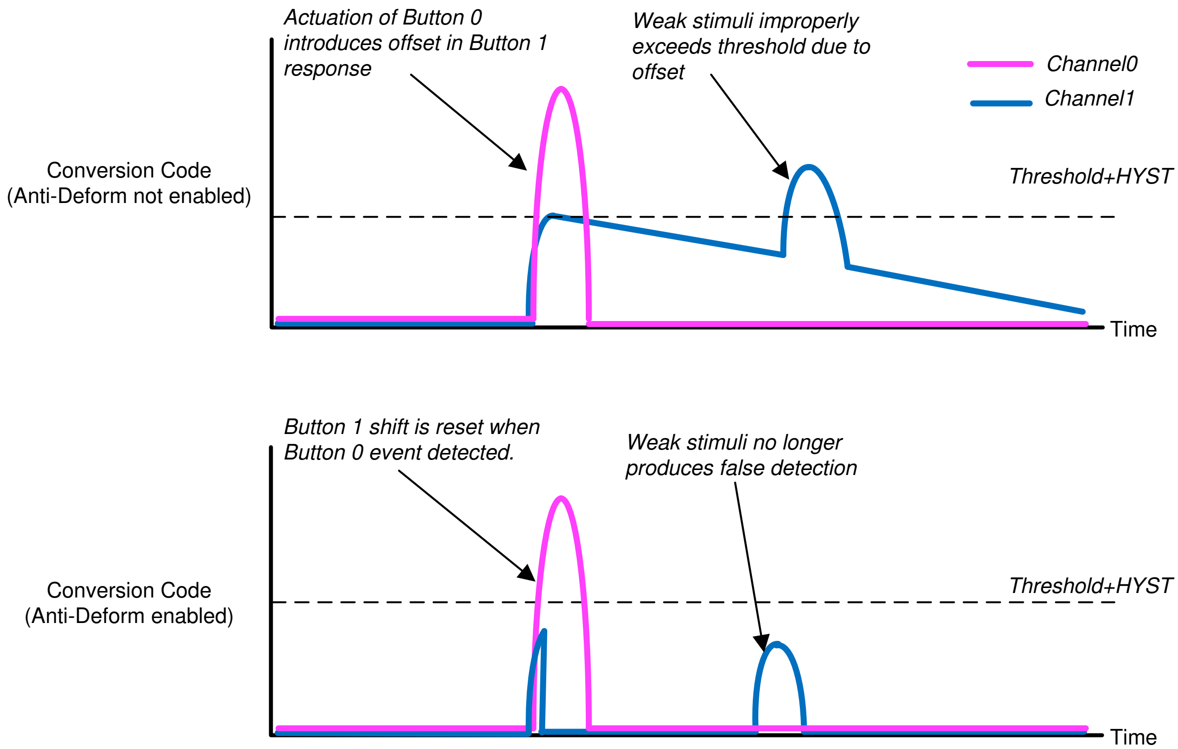

Figure 7-5. Deformation Can Cause Positive or Negative Offsets

Anti-Deform resets the baselines for channels that don't have a button press detection. This returns the sensitivity back to the system nominal level.

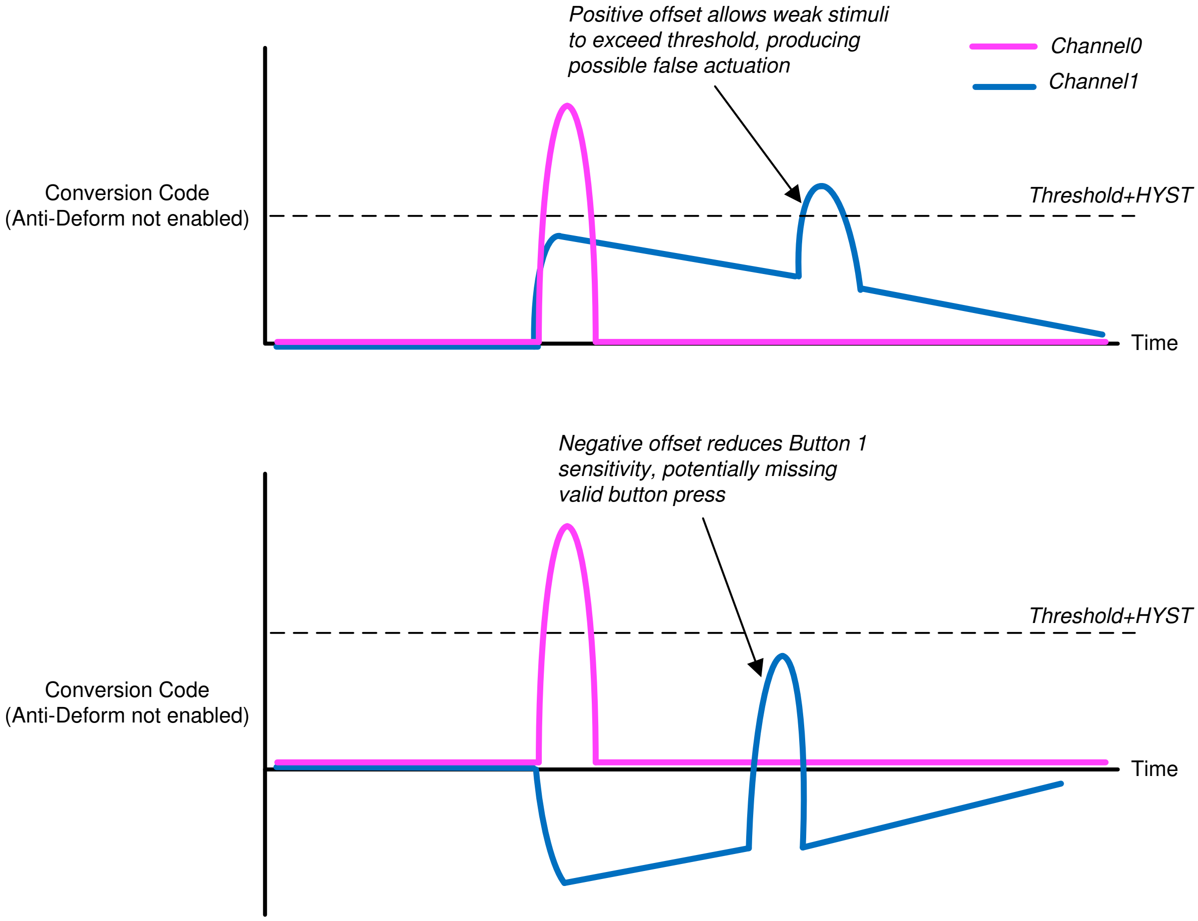

Figure 7-6. Anti-Deform Functionality

Channels can be included in the Anti-Deform group by setting the corresponding ANTIDFORMx field to 1. The default setting for the ANTIDFORMx field is 0, which disables Anti-Deform for corresponding channel. The pseudo-code for this algorithm is equivalent to:

```
If channel is active and channel has ANTIDFORM enabled and OUT[channel] is true then

For each AntiDform channel

baseline[channel] = raw_data[channel]

Next Channel
```

8 Summary

The internal algorithms of the LDC211x and LDC3114 devices can address a wide range of system variations. Appropriate use of the algorithms and details on the functionality provide clarity in which conditions to use which algorithm.

9 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision * (June 2018) to Revision A (July 2021) **Page**

- Updated document to include the *LDC3114* device....................1
- Updated the numbering format for tables, figures, and cross-references throughout the document....................1

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265