
maXTouch 144-node Touchscreen Controller

Protocol Guide

TABLE OF CONTENTS

Table of Contents	2
To Our Valued Customers	4
1.0 Overview	5
1.1 Introduction	5
1.2 Memory Map Structure	5
1.3 Information Block	5
1.4 ID Information	6
1.5 Object Table	6
1.6 Objects	8
1.7 Configuration Checks	11
1.8 Byte Order	11
1.9 Delta Values	11
1.10 Configuration and Tuning	11
1.11 Software Tools	12
2.0 Debug Objects	13
2.1 Introduction	13
2.2 Diagnostic Debug T37 Object	14
3.0 General Objects	19
3.1 Introduction	19
3.2 Message Processor T5 Object	20
3.3 Command Processor T6 Object	22
3.4 Power Configuration T7 Object	25
3.5 Acquisition Configuration T8 Object	29
4.0 Touch Objects	40
4.1 Introduction	40
4.2 Key Array T15	41
4.3 Multiple Touch Touchscreen T100 Object	46
5.0 Signal Processing Objects	79
5.1 Introduction	79
5.2 One-touch Gesture Processor T24 Object	80
5.3 Two-touch Gesture Processor T27 Object	89
5.4 Grip Suppression T40 Object	92
5.5 Touch Suppression T42 Object	97
5.6 Shieldless T56 Object	102
5.7 Lens Bending T65 Object	104
5.8 Noise Suppression T72 Object	110
5.9 Glove Detection T78 Object	128
5.10 Retransmission Compensation T80 Object	132
5.11 Touch Sequence Processor T93 Object	138
5.12 Self Capacitance Noise Suppression T108 Object	144
5.13 Symbol Gesture Processor T115 Object	155
5.14 Sensor Correction T121 Object	161
6.0 Support Objects	163
6.1 Introduction	163
6.2 Communications Configuration T18 Object	164
6.3 Self Test T25 Object	165
6.4 User Data T38 Object	171
6.5 Message Count T44 Object	172
6.6 CTE Configuration T46 Object	173
6.7 Timer T61 Object	176
6.8 Serial Data Command T68 Object	178
6.9 Dynamic Configuration Controller T70 Object	181
6.10 Dynamic Configuration Container T71 Object	189
6.11 Auxiliary Touch Configuration T104 Object	190
6.12 Self Capacitance Global Configuration T109 Object	193
6.13 Self Capacitance Tuning Parameters T110 Object	196
6.14 Self Capacitance Configuration T111 Object	197
6.15 Self Capacitance Measurement Configuration T113 Object	206
6.16 Symbol Gesture Configuration T116 Object	208
6.17 Low Power Idle Configuration T126 Object	212

Appendix A. Measurement Processing on mXT144U	215
A.1 Measurement Processing Objects	215
A.2 Touch Detection	216
A.3 Touch Classification	216
Appendix B. Checksum Calculation	218
B.1 24-bit CRC	218
B.2 8-bit CRC	220
Appendix C. Bootloading Procedure	224
Appendix D. Locating Configuration Errors	225
Appendix E. Future Information Block T254 Object	226
E.1 Introduction	226
E.2 Information Block T254	226
Appendix F. Associated Documents	228
Appendix G. Revision History	229
The Microchip Web Site	238
Customer Change Notification Service	238
Customer Support	238

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at docerrors@microchip.com. We welcome your feedback.

Most Current Document

To obtain the most up-to-date version of this document, please contact your Microchip representative.

You can determine the version of a document by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (for example, DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the protocol guide and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please contact your Microchip representative.

When contacting Microchip, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at <http://www.microchip.com> to receive the most current information on all of our products.

1.0 OVERVIEW

1.1 Introduction

This document describes in detail the Object Protocol for the Microchip mXT144U 1.0.

The Object Protocol provides a single common interface across the Microchip maXTouch controllers. This allows the different features in each controller to be configured in a consistent manner. This makes the future expansion of features and simple product upgrades possible, whilst allowing backwards compatibility for the host driver and application code.

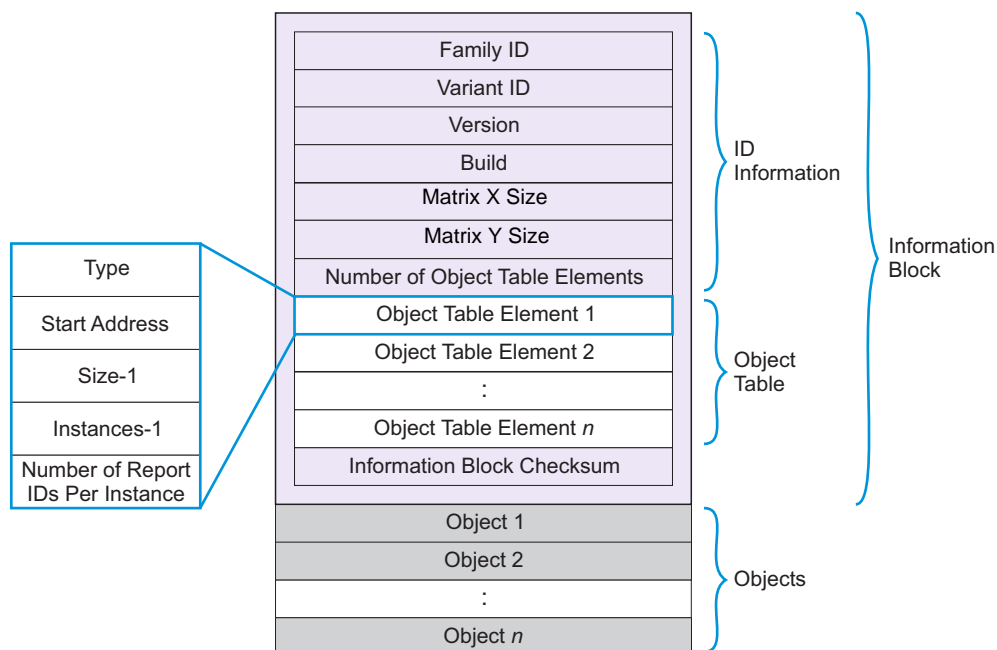
1.2 Memory Map Structure

The protocol is designed to control the processing chain in a modular manner. This is achieved by breaking the features of the device into objects that can be controlled individually. Each object represents a certain feature or function of the device, such as a touchscreen. Where appropriate, objects can be disabled or enabled as needed.

Each object has its own configuration memory. The objects are stacked together to produce an object-based memory map. A generalized structure of this memory map is shown in [Figure 1-1](#).

There are some special objects that can use their memory space for a unique purpose. An example is the Command Processor T6 object, which executes a command when a certain value is written into its memory space.

FIGURE 1-1: GENERIC MEMORY MAP STRUCTURE



From [Figure 1-1](#) it can be seen that the memory map contains two main sections:

- An Information Block. This documents which objects are contained in the memory map for the device (see [Section 1.3 “Information Block”](#)). It is further sub-divided into the ID Information Block (see [Section 1.4 “ID Information”](#)) and the Object Table (see [Section 1.5 “Object Table”](#)).
- The objects themselves (see [Section 1.6 “Objects”](#)).

1.3 Information Block

The Information Block allows the host to read information about the layout of objects in the memory map. It contains a list of all the objects in the memory map. This is used by the host driver to determine which objects exist, where they are located in the memory map and their sizes. The host driver can therefore read the device Information Block and gather enough information to be able to communicate with the device.

The Information Block is positioned at the start of the device memory map at address zero. This allows it to be read easily by the host as the first operation.

The Information Block contains the following three sections:

- **ID Information Fields** – These include:
 - Standard ID fields that make up the unique identifier for the device
 - The size of the touchscreen matrix the device supports
 - The number of objects in the Object Table
- **Object Table** – This acts as an “index” to the objects in the memory map. Note that one of the objects may be an Information Block object that holds additional Object Table entries (see [Appendix E “Future Information Block T254 Object”](#)).
- **Information Block Checksum** – This allows the host to check that the Information Block has been read correctly over the communications interface.

[Table 1-1](#) shows the contents of the Information Block.

TABLE 1-1: INFORMATION BLOCK LAYOUT

Byte	Description of Field	Section
0	Family ID	ID Information
1	Variant ID	
2	Version	
3	Build	
4	Matrix X Size	
5	Matrix Y Size	
6	Number of elements in the Object Table	
7 – 12	Object Table element 1 (6 bytes)	Object Table
13 – 18	Object Table element 2 (6 bytes)	
...	...	
...	Last Object Table element (6 bytes)	
(end-2) – end	24-bit checksum (3 bytes)	Checksum Field

1.4 ID Information

The first four bytes of the Information Block are used to identify the device and its version, as in [Table 1-2](#).

TABLE 1-2: DEVICE IDENTIFIER FIELDS

Field	Description
Family ID	A unique identifier that indicates the device family. The family ID for the mXT144U is 0xA6.
Variant ID	A unique identifier that indicates the device variant. The variant ID for the mXT144U is 0x08.
Version	The current major and minor firmware version of the device. The upper nibble contains the major version and the lower nibble contains the minor version. For example, firmware version 1.0 is stored as 0x10.
Build	The firmware build number.

1.5 Object Table

1.5.1 INTRODUCTION

The Object Table is held within the Information Block. The Object Table contains information on all the objects held within the memory map. It indicates which objects are present and their addresses.

Each element in the Object Table has the fields listed in [Table 1-3](#).

TABLE 1-3: FORMAT OF AN OBJECT TABLE ELEMENT

Byte	Description of Field
0	Type
1	Start position LSByte
2	Start position MSByte
3	Size – 1
4	Instances – 1
5	Number of Report IDs per instance

IMPORTANT! Future versions of the device may include the Information Block T254 object. This object will contain additional Object Table entries. For future compatibility, any current driver code should be written to allow for the presence of the Information Block T254 object when it is eventually used. The host driver code must parse the Object Table, as at present, and also process the Information Block T254 object, if this is found to be present on the device. See [Appendix E “Future Information Block T254 Object”](#) for more information.

1.5.2 TYPES

Each type of object has a unique type code to identify it. This is the number after the “T” suffix at the end of the object’s name, as given in the object descriptions. For example, the type code for the Command Processor T6 (GEN_COMMANDPROCESSOR_T6) is **6**.

1.5.3 START POSITION

Bytes 1 and 2 of the Object Table element holds the start location of the object in the memory map (LSByte and MSByte respectively).

The driver code should ALWAYS read these bytes to find out where in the memory map the object is located and use this address to communicate with the object. The driver code should never use hard-coded addresses for the objects, as these may change with firmware updates.

This means that driver code can be written without making assumptions about the addresses of the objects. This ensures that the code is “future-proof” and will work correctly following firmware updates to the device. It also makes it possible to write common driver code for communication with any Microchip device that uses this object-based protocol approach.

1.5.4 SIZE

Byte 3 of the Object Table element holds the size in bytes (minus 1) of the object in the memory map. This is stored as Size–1, so it is effectively the offset to the end of the object.

1.5.5 NUMBER OF INSTANCES

Byte 4 of the Object Table element holds the number of instances of the object in the memory map, minus 1. The number of instances can be calculated by adding 1 to this number (see [Table 1-4](#)). The different instances of an object are arranged sequentially in the memory map.

1.5.6 REPORT IDS

If an object sends messages, it is necessary to identify the messages from the object so that they can be correctly interpreted. A report ID is therefore used to identify the source object of a message returned in the Message Processor T5 object (see [Section 3.2 “Message Processor T5 Object”](#)).

Report IDs are numbered uniquely and sequentially in the order in which the objects are listed in the Object Table, allowing for the appropriate number of instances for each object. Note the following:

- A report ID of zero is a reserved value for use by Microchip. Report IDs from a user’s perspective therefore effectively start from 1.
- A report ID value of 255 is reserved to indicate an “invalid message” response.
- If an object has report IDs allocated, each instance of the object will have its own block of report IDs.

- In the case of a Multiple Touch Touchscreen T100 object, there is 1 report ID allocated for the screen status message, 1 reserved report ID, and 5 report IDs for the touch status messages, making a total of 7 report IDs.

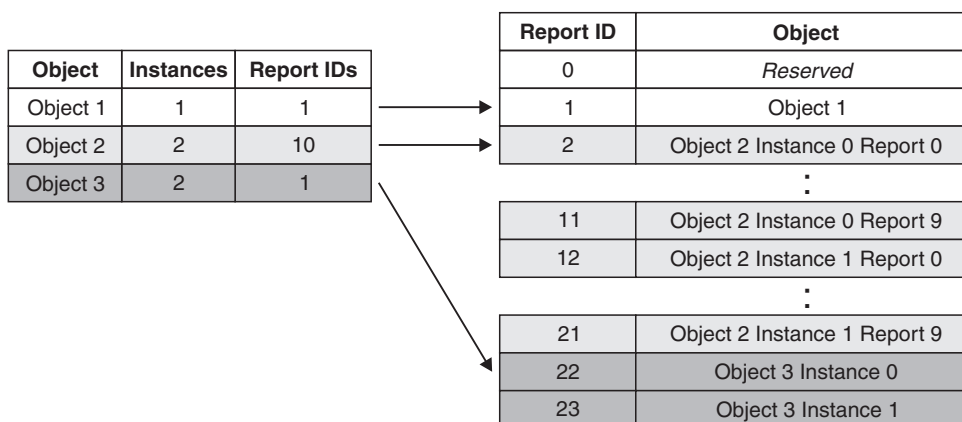
Figure 1-2 shows an example of how the number of instances and the Report IDs per instance determine the report IDs for a set of objects.

The driver code should build up its own in-memory table of object types and associated report IDs during its initialization. It can do this by parsing the object structure given in the Object Table. This in-memory table can then be used to interpret the messages returned by the device.

A typical algorithm to process the report IDs is as follows:

1. For each element in the Object Table:
 - a) Read byte 4 to retrieve the number of instances (remember to add 1 to the value retrieved).
 - b) Read byte 5 to retrieve the number of report IDs per instance.
 - c) Multiply the figures retrieved in steps a. and b. together, and then add this number of “object type/report ID” pairings to the table being built. The report IDs should have sequential values starting with 1 (the zero value is reserved for use by Microchip).
2. Check for the presence of an Information Block T254 object (See [Appendix E “Future Information Block T254 Object”](#)). If the object is present on the device, repeat Step 1 for each element held within the Information Block T254 object, continuing to allocate the new report IDs to those already allocated.

FIGURE 1-2: EXAMPLE ASSIGNMENT OF REPORT IDS



The number of report IDs for each object is shown in [Table 1-4](#).

1.6 Objects

1.6.1 CLASSES OF OBJECTS

The mXT144U contains the following classes of objects:

- **Debug objects** – provide a raw data output method for development and testing. See [Section 2.0 “Debug Objects”](#).
- **General objects** – required for global configuration, transmitting messages and receiving commands. See [Section 3.0 “General Objects”](#).
- **Touch objects** – operate on measured signals from the touch sensor and report touch data. See [Section 4.0 “Touch Objects”](#).
- **Signal processing objects** – process data from other objects (typically signal filtering operations). See [Section 5.0 “Signal Processing Objects”](#).
- **Support objects** – provide additional functionality on the device. See [Section 6.0 “Support Objects”](#).

1.6.2 OBJECT INSTANCES

TABLE 1-4: OBJECTS ON THE MXT144U

Object	Number of Instances	Report IDs per Instance	Reference
Debug Objects			
Diagnostic Debug T37	1	0	Section 2.2 “Diagnostic Debug T37 Object”
General Objects			
Message Processor T5	1	0	Section 3.2 “Message Processor T5 Object”
Command Processor T6	1	1	Section 3.3 “Command Processor T6 Object”
Power Configuration T7	1	0	Section 3.4 “Power Configuration T7 Object”
Acquisition Configuration T8	1	0	Section 3.5 “Acquisition Configuration T8 Object”
Touch Objects			
Key Array T15	1	1	Section 4.2 “Key Array T15”
Multiple Touch Touchscreen T100	1	7	Section 4.3 “Multiple Touch Touchscreen T100 Object”
Signal Processing Objects			
One-touch Gesture Processor T24	1	4	Section 5.2 “One-touch Gesture Processor T24 Object”
Two-touch Gesture Processor T27	1	1	Section 5.3 “Two-touch Gesture Processor T27 Object”
Grip Suppression T40	1	0	Section 5.4 “Grip Suppression T40 Object”
Touch Suppression T42	1	0	Section 5.5 “Touch Suppression T42 Object”
Shieldless T56	1	1	Section 5.6 “Shieldless T56 Object”
Lens Bending T65	3	1	Section 5.7 “Lens Bending T65 Object”
Noise Suppression T72	1	1	Section 5.8 “Noise Suppression T72 Object”
Glove Detection T78	1	0	Section 5.9 “Glove Detection T78 Object”
Retransmission Compensation T80	1	1	Section 5.10 “Retransmission Compensation T80 Object”
Touch Sequence Processor T93	1	1	Section 5.11 “Touch Sequence Processor T93 Object”
Self Capacitance Noise Suppression T108	1	1	Section 5.12 “Self Capacitance Noise Suppression T108 Object”
Symbol Gesture Processor T115	1	1	Section 5.13 “Symbol Gesture Processor T115 Object”
Sensor Correction T121	1	0	Section 5.14 “Sensor Correction T121 Object”

TABLE 1-4: OBJECTS ON THE MXT144U (CONTINUED)

Object	Number of Instances	Report IDs per Instance	Reference
Support Objects			
Communications Configuration T18	1	0	Section 6.2 “Communications Configuration T18 Object”
Self Test T25	1	1	Section 6.3 “Self Test T25 Object”
User Data T38	1	0	Section 6.4 “User Data T38 Object”
Message Count T44	1	0	Section 6.5 “Message Count T44 Object”
CTE Configuration T46	1	1	Section 6.6 “CTE Configuration T46 Object”
Timer T61	6	1	Section 6.7 “Timer T61 Object”
Serial Data Command T68	1	1	Section 6.8 “Serial Data Command T68 Object”
Dynamic Configuration Controller T70	20	1	Section 6.9 “Dynamic Configuration Controller T70 Object”
Dynamic Configuration Container T71	1	0	Section 6.10 “Dynamic Configuration Container T71 Object”
Auxiliary Touch Configuration T104	1	0	Section 6.11 “Auxiliary Touch Configuration T104 Object”
Self Capacitance Global Configuration T109	1	1	Section 6.12 “Self Capacitance Global Configuration T109 Object”
Self Capacitance Tuning Parameters T110	4	0	Section 6.13 “Self Capacitance Tuning Parameters T110 Object”
Self Capacitance Configuration T111	2	0	Section 6.14 “Self Capacitance Configuration T111 Object”
Self Capacitance Measurement Configuration T113	1	0	Section 6.15 “Self Capacitance Measurement Configuration T113 Object”
Symbol Gesture Configuration T116	1	0	Section 6.16 “Symbol Gesture Configuration T116 Object”
Low Power Idle Configuration T126	1	1	Section 6.17 “Low Power Idle Configuration T126 Object”

1.6.3 CONFIGURATION VALUES

The objects are designed such that a value of zero in their fields is a “safe” value. Typically a value of zero means that a default value is used, and this will be indicated within the object’s field descriptions in this protocol guide.

An object must be configured as required with non-zero values before use. Any unused settings can be left at their default zero values. The settings should also be written to the non-volatile memory using the Command Processor T6 object (see [Section 3.3 “Command Processor T6 Object”](#)).

The field descriptions in this protocol guide specify the recommended ranges for values. Any values outside the specified range for a field should be treated as reserved values and should not be used. Note that in many cases the device will not allow values outside the specified ranges.

1.6.4 COMPATIBILITY OF OBJECT VERSIONS

The Object Protocol described in this document may contain fields that are not present in the memory map as it is implemented on a particular firmware version of the device. Over time newer versions of the objects in the Object Protocol may gain additional fields to implement new features.

New fields are added to the end of an object to allow for this situation. This preserves the order of the fields between old and new object versions. A driver designed to work with an older version of the device can safely set any unknown fields located at the end of the object to zero. The device will then behave in the same manner as the older version of the device without the field.

The host driver must always use the Object Table to locate the address of each object and the object's current size. It must also zero any fields that it does not intend to use. This ensures that the host driver code is compatible with this object expansion scheme.

1.7 Configuration Checks

The device prevents invalid configurations by performing a sequence of checks. These checks are performed on power-up. They are also performed whenever certain configuration settings are updated. These are typically settings that force a recalculation, a reinitialization or a recalibration when they are changed.

If an error is found during the configuration check, the device pauses until the configuration error is resolved. The device also flags the error to the host by setting the CFGERR bit of the Command Processor T6 message data (see [Section 3.3.3 “Messages”](#)). This message will be sent to the host every 200 ms until the error is corrected.

Backup requests are allowed when an error has been found during a configuration check. This allows a setting to be corrected and backed up to the non-volatile memory (NVM). The device can then be reset for the setting to take effect, if the setting requires this. The device will not operate until all errors have been corrected.

Possible causes of configuration errors are:

- Touch objects with overlapping channels or channels outside the maximum matrix dimensions
- Touch objects under the minimum X and Y size, or a Key Array T15 object with a channel size greater than 8
- Field settings outside the permitted range
- A bad checksum for the configuration settings held in the non-volatile memory

If a configuration error is encountered during product development, it is sufficient for the designer to check the settings used and correct them. In a working product, the device driver should resend the configuration settings and request a backup. The device driver should be written to handle this.

To find the source of the configuration error, first disable all touch, signal processing and support objects. Then re-enable each object in turn until the configuration error is found. Once the error has been corrected, the other objects can be re-enabled and processing can continue as normal. See [Appendix D “Locating Configuration Errors”](#) for a flow chart showing this method for finding configuration errors.

Notice should be taken of any recommendations in this document concerning configuration checks for the offending object.

1.8 Byte Order

The memory map uses a “little-endian” configuration for its bytes, meaning that all multibyte fields lead with the least significant byte (LSByte).

1.9 Delta Values

During measurement, the device uses raw 16-bit values for the deltas. These are converted by the Object-based Protocol to 8-bit deltas wherever possible in many of the controls and in the diagnostic debug data. This saves space and, in the case of debug data, allows for significant time savings in data transfer.

For Mutual Capacitance and Self Capacitance Touch deltas, the conversion between 16-bit deltas and 8-bit deltas is as follows:

$$16\text{-bit delta} = 8\text{-bit delta} \times 8$$

$$8\text{-bit delta} = 16\text{-bit delta} / 8$$

1.10 Configuration and Tuning

This Protocol Guide provides reference information on the various configuration parameters for the mXT144U. For further guidance on configuring and tuning the mXT144U, refer to the *maXTouch U Series Tuning Guide*.

1.11 Software Tools

The primary interface with the mXT144U device is through the host driver and application code within the user's product.

An alternative interface is also available by using maXTouch Studio. This provides a graphical IDE within which the configuration and tuning of a maXTouch device can take place so that the performance of the device can be assessed. Note that maXTouch Studio Lite is supplied with the maXTouch evaluation kits.

Various plug-in tools are also available to help with configuring and tuning the various objects within the device and can be integrated into the maXTouch Studio development environment. Contact your Microchip representative for more information.

2.0 DEBUG OBJECTS

2.1 Introduction

Debug objects contain raw data for development and testing purposes. [Table 2-1](#) lists the debug objects on the mXT144U.

TABLE 2-1: DEBUG OBJECTS

Object	Description
Diagnostic Debug T37 (DEBUG_DIAGNOSTIC_T37)	Allows access to diagnostic debug data to aid development. See Section 2.2 "Diagnostic Debug T37 Object" .

2.2 Diagnostic Debug T37 Object

2.2.1 INTRODUCTION

The Diagnostic Debug T37 object holds the diagnostic debug data. It reports both system data and object-specific data from individual objects. See [Section 2.2.1.1 "System Diagnostic Debug Modes"](#) for more information on the system diagnostic debug modes. The object-specific diagnostic debug modes are described elsewhere in this document in the sections for the specific objects.

2.2.1.1 System Diagnostic Debug Modes

The Diagnostic Debug T37 object works by organizing the data in pages. The value of the PAGE field determines which page of data is currently available, and the object's DATA field holds the data for that page. The pages can be navigated by writing Page Up/Page Down commands to the DIAGNOSTIC field in the Command Processor T6 object (see ["DIAGNOSTIC Field"](#)).

The mode in which the data is displayed is determined by the command written to the DIAGNOSTIC field of the Command Processor T6 object (see ["DIAGNOSTIC Field"](#)).

A mode or page change command is processed only once per measurement cycle. Note that no command processing is done if the device is in deep sleep mode.

The system diagnostic debug modes are shown in [Table 2-2](#). The object-specific diagnostic debug modes are described within the individual object sections.

TABLE 2-2: SYSTEM DIAGNOSTIC DEBUG DATA MODES

T6 Command	Debug Mode	Description
0x01	Page Up	Command Processor T6 debug data page navigation
0x02	Page Down	Command Processor T6 debug data page navigation
0x10	Mutual Capacitance Deltas Mode	The Diagnostic Debug T37 object holds signal deltas. These are signed (two's complement) 16-bit mutual capacitance delta values for all nodes. Note that these are the raw deltas. The firmware divides these by 8 before use elsewhere (such as for threshold comparisons).
0x11	Mutual Capacitance References Mode	The Diagnostic Debug T37 object holds reference values. These are signed (two's complement) 16-bit mutual capacitance reference values for all nodes.
0x38	DC Data Mode	The Diagnostic Debug T37 object holds the DC level estimates for Self Capacitance Touch data as signed 16-bit values. Note that data values will be set to zero if Self Capacitance Touch is not enabled.
0x80	Device Information Mode	The Diagnostic Debug T37 object holds information on the device; specifically, the revision identifier.
0x81	Product Data Store Mode	The Diagnostic Debug T37 object holds the contents of the Product Data Store (PDS) and its CRC. See Section 2.2.1.8 "Product Data Store Mode" .
0xF7	Self Capacitance Delta Mode	The Diagnostic Debug T37 object holds the self capacitance deltas. These are signed (two's complement) 16-bit self capacitance delta values for all nodes.
0xF8	Self Capacitance Reference Mode	The Diagnostic Debug T37 object holds the self capacitance reference values.
Other value	Object-specific debug mode	See individual objects for details of the debug data available.

Note: Changing the mode resets the page number to zero.

2.2.1.2 Delta Mode (Mutual Capacitance)

Figure 2-1 shows the organization of the data in the pages for the mutual capacitance delta mode. Note that the nodes are numbered along the X lines, that is in the order: element 0 = X0Y0, 1 = X0Y1, 2 = X0Y2 and so on.

The page number and data index for a given mutual capacitance node's data can be calculated as follows:

Page number = FLOOR (node_number / nodes_per_page)
Data index = bytes_per_node x (node_number MOD nodes_per_page)

where:

- bytes_per_node is 2.
- nodes_per_page is the page size divided by bytes_per_node. In Figure 2-1 this is (128 / 2).

FIGURE 2-1: DIAGNOSTIC DEBUG T37 FIELDS – DELTA MODE (MUTUAL CAPACITANCE)

Page	Data Index	Data[]
Page 0	0 – 127	Deltas for nodes 0 to 63
		:
		:
Page 143	256 – 287	Deltas for nodes 128 to 143
	288 – 383	Reserved

Assumes page size = 128

2.2.1.3 Reference Mode (Mutual Capacitance)

Figure 2-2 shows the organization of the data in the pages for the mutual capacitance reference mode. Note that the nodes are numbered along the X lines, that is in the order: element 0 = X0Y0, 1 = X0Y1, 2 = X0Y2 and so on.

The page number and data index for a given mutual capacitance node's data can be calculated as follows:

Page number = FLOOR (node_number / nodes_per_page)
Data index = bytes_per_node x (node_number MOD nodes_per_page)

where:

- bytes_per_node is 2.
- nodes_per_page is the page size divided by bytes_per_node. In Figure 2-2 this is (128 / 2).

FIGURE 2-2: DIAGNOSTIC DEBUG T37 FIELDS – REFERENCE MODE (MUTUAL CAPACITANCE)

Page	Data Index	Data[]
Page 0	0 – 127	References for nodes 0 to 63
		:
		:
Page 2	256 – 287	References for nodes 128 to 143
	288 – 383	Reserved

Assumes page size = 128

2.2.1.4 Delta Mode (Self Capacitance)

Figure 2-3 shows the organization of the data in the pages for the self capacitance delta mode.

FIGURE 2-3: DIAGNOSTIC DEBUG T37 FIELDS – DELTA MODE (SELF CAPACITANCE)

Page	Data Index	Data[]	
Page 0	0 – 23	Touch deltas for Y0 to Y11	
	24 – 47	Touch deltas for X0 to X11	
	48 – 127	Reserved	
Page 2	0 – 23	Single-ended deltas for Y0 to Y11	When Self Capacitance Measurement Configuration T113 CTRL ALTAXISEN = 1
	24 – 47	Single-ended deltas for X0 to X11	
	48 – 127	Reserved	When Self Capacitance Measurement Configuration T113 CTRL ALTAXISEN = 0

Assumes page size = 128

2.2.1.5 Reference Mode (Self Capacitance)

Figure 2-4 shows the organization of the data in the pages for the self capacitance reference mode.

FIGURE 2-4: DIAGNOSTIC DEBUG T37 FIELDS – REFERENCE MODE (SELF CAPACITANCE)

Page	Data Index	Data[]	
Page 0	0 – 23	Touch references for Y0 to Y11	
	24 – 35	Touch references for even X lines	
	36 – 47	Reserved	
	48 – 59	Touch references for odd X lines	
	60 – 127	Reserved	
Page 1	Reserved; page unused		
Page 2	0 – 23	Single-ended references for Y0 to Y11	When Self Capacitance Measurement Configuration T113 CTRL ALTAXISEN = 1
	24 – 35	Single-ended references for even X lines	
	36 – 47	Reserved	When Self Capacitance Measurement Configuration T113 CTRL ALTAXISEN = 0
	48 – 59	Single-ended references for odd X lines	
	60 – 127	Reserved	

Assumes page size = 128

2.2.1.6 Device Information Mode

Figure 2-5 shows the organization of the data in the pages for the device information mode.

NOTE The format of the device revision mode varies from device to device.

The Revision ID for the device silicon can be read from the upper 4 bits of Byte 19 on Page 0.

FIGURE 2-5: DIAGNOSTIC DEBUG T37 FIELDS – DEVICE REVISION IDENTIFIER MODE

Page	Data Index	Data[]
Page 0	0 – 18	Reserved
	19	Revision ID (upper 4 bits)
	20 – 127	Reserved

Assumes page size = 128

2.2.1.7 DC Data Mode

Figure 2-6 shows the organization of the data in the pages for the DC Data Mode.

FIGURE 2-6: DIAGNOSTIC DEBUG T37 FIELDS – DC DATA MODE

Page	Data Index	Data[]
Page 0	0 – 1	DC estimate for self capacitance touch seeded by single-ended self capacitance touch
	2 – 3	DC estimate for single-ended self capacitance touch after filtering, drifting and removal of the reference
	4 – 5	Single-ended self capacitance reference from which single-ended self capacitance DC estimate is derived
	6 – 7	Single-ended self capacitance raw DC estimate
	8 – 9	The larger of the DC estimates for X and Y axes for self capacitance touch
	10 – 11	DC estimate of the axis being processed by Self Capacitance Measurement Configuration T113 for self capacitance touch
	12 – 127	Reserved

Assumes page size = 128

2.2.1.8 Product Data Store Mode

Figure 2-7 shows the organization of the data in the pages for the Product Data Store mode.

Note that if the Serial Data Command T68 object has not yet written any data, then the data and checksum will be set to 0. The checksum status will also report 0 (that is, the data is valid).

FIGURE 2-7: DIAGNOSTIC DEBUG T37 FIELDS – PRODUCT DATA STORE MODE

Page	Data Index	Data[]
Page 0	0	Number of bytes in the data, including checksum (CRC)
	1	Checksum status ⁽¹⁾
	2 – 61	Data written to PDS by Serial Data Command T68
	62 – 65	24-bit checksum (stored in 32-bit field)
	66 – 127	Reserved

Assumes page size = 128

Note 1: Checksum status has one of the following values:
0 = data is valid
255 = data is invalid

2.2.2 CONFIGURATION

**TABLE 2-3: CONFIGURATION FOR DIAGNOSTIC DEBUG T37
(DEBUG_DIAGNOSTIC_T37)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	MODE	Current mode							
1	PAGE	Current page							
2 – (2+n-1)	DATA[n]	Current page of data							

 n = page size**MODE Field**

This field contains an indication of the current mode for the data in the DATA[] field. It has the same value as the mode change commands (but not the page change commands) in the DIAGNOSTIC field of the Command Processor T6 object (see “[DIAGNOSTIC Field](#)”).

PAGE Field

This field contains the current page number for the data held in the DATA[] field, as controlled by the page change commands in the DIAGNOSTIC field of the Command Processor T6 object (see “[DIAGNOSTIC Field](#)”).

DATA[] Fields

This field contains the current page of data.

3.0 GENERAL OBJECTS

3.1 Introduction

General objects provide global configuration, such as receiving commands and transmitting messages. [Table 3-1](#) lists the general objects on the mXT144U.

TABLE 3-1: GENERAL OBJECTS

Object	Description
Message Processor T5 (GEN_MESSAGEPROCESSOR_T5)	Handles the transmission of messages. This object holds a message in its memory space for the host to read. See Section 3.2 “Message Processor T5 Object” .
Command Processor T6 (GEN_COMMANDPROCESSOR_T6)	Performs a command when written to. Commands include reset, calibrate and backup settings. See Section 3.3 “Command Processor T6 Object” .
Power Configuration T7 (GEN_POWERCONFIG_T7)	Controls the sleep mode of the device. Power consumption can be lowered by controlling the acquisition frequency and the sleep time between acquisitions. See Section 3.4 “Power Configuration T7 Object” .
Acquisition Configuration T8 (GEN_ACQUISITIONCONFIG_T8)	Controls how the device takes each capacitive measurement. See Section 3.5 “Acquisition Configuration T8 Object” .

3.2 Message Processor T5 Object

3.2.1 INTRODUCTION

The purpose of the Message Processor T5 object is to relay the latest status information to the host. This object contains the message data from those objects in the memory map that generate messages (for example, the touch objects and the Command Processor T6 object). A message is generated whenever status of an object has changed. For this to happen, the object report enable bit must be set.

The system provides an interrupt style mechanism using the $\overline{\text{CHG}}$ line to indicate to the host that there is at least one message to read. See [Section 3.2.1.1 “Reading Messages with the \$\overline{\text{CHG}}\$ Line”](#) for more details.

3.2.1.1 Reading Messages with the $\overline{\text{CHG}}$ Line

When using the I²C interface, if a device has data to send, it asserts the $\overline{\text{CHG}}$ line (active low) to indicate to the host that there is a message. The host should then read the message and use the REPORTID field to determine from which object the message originated. This provides the host with an interrupt-style interface which has the potential for fast response times and reduces the need for wasteful communications.

The host should ALWAYS use the $\overline{\text{CHG}}$ line as an indication that a message is available to read in the Message Processor T5 object. The host should not read the Message Processor T5 object at any other time, such as to poll the device for messages. If the Message Processor T5 object is read when the $\overline{\text{CHG}}$ line is not asserted, an “invalid message” report ID is returned in the Message Processor T5 object.

Multiple messages can easily be read in a continuous read operation using direct memory access (DMA) and message pointer wrapping is implemented to allow this. The Message Count T44 object (see [Section 6.5 “Message Count T44 Object”](#)) provides a count of pending messages so that the host driver code knows how many messages to read.

Message pointer wrapping occurs if the address pointer is pointing at either the Message Count T44 object or the start of the Message Processor T5 object (that is, the report ID of the next available message).

NOTE

- The Message Count T44 object occurs immediately before the Message Processor T5 object in the memory map so that the message wrapping mechanism will work.
- If checksum mode is enabled, the address pointer wrapping between messages occurs after the checksum byte has been sent. Otherwise the wrapping occurs before the checksum byte to save unnecessary reads of the unused checksum byte.

3.2.2 CONFIGURATION

**TABLE 3-2: CONFIGURATION FOR MESSAGE PROCESSOR T5
(GEN_MESSAGEPROCESSOR_T5)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	REPORTID	Report ID for source object							
1 – n	MESSAGE	Message data from source object							
n+1	CHECKSUM	Checksum							

Note: n = size of largest message possible

NOTE

The size of the MESSAGE field is set to the length of largest message possible. It is dependent on the objects present in the device at a particular revision. The MESSAGE field size should be calculated by subtracting 2 from the size of the Message Processor T5 object retrieved from the Object Table entry (see [Section 1.5 “Object Table”](#)). Any unused bytes in a particular message should be treated as reserved bytes.

REPORTID Field

This field contains the report ID for the message. Messages contain report IDs to allow the host to identify the type of message and its originator. Report IDs are assigned to any object that can send messages. See [Section 1.5.6 “Report IDs”](#) for more information on the assignment of report IDs.

MESSAGE Field

This field contains the message data for the object generating the message.

The size of the MESSAGE field is fixed to the size of the message data for the largest object. For compatibility with future firmware updates, this should *always* be calculated by subtracting 2 from the size of the object recorded in the Object Table entry for the Message Processor T5 object (see [Section 1.5 “Object Table”](#)).

For information on the contents of the MESSAGE field, see the descriptions for each object elsewhere in this protocol guide.

CHECKSUM Field

This field contains the 8-bit checksum for the Message Processor T5 object (that is, for the REPORTID and MESSAGE fields) if a communications checksum is requested.

To request that a checksum is generated, the MSBit of the address of the Message Processor T5 object is set to 1 during a read. For example, if the address of the Message Processor T5 object is 0x0477, specifying the address as 0x8477 will generate a checksum for that read.

If the communications checksum feature is not enabled, this byte should not be read.

See [Appendix B “Checksum Calculation”](#) for details on how to calculate the checksum.

3.3 Command Processor T6 Object

3.3.1 INTRODUCTION

The Command Processor T6 object allows commands to be sent to the device. This is done by writing an appropriate value to one of its fields.

3.3.2 CONFIGURATION

**TABLE 3-3: CONFIGURATION FOR COMMAND PROCESSOR T6
(GEN_COMMANDPROCESSOR_T6)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	RESET	Reset							
1	BACKUPNV	Backup settings							
2	CALIBRATE	Calibrate							
3	REPORTALL	Report current status							
4	DEBUGCTRL	Reserved		SIGNAL	REFS	DELTAS	DELTAS8	TSCRN	CHIP
5	DIAGNOSTIC	Diagnostic debug command							
6	DEBUGCTRL2	DBGOBJMODEEN	Reserved						

RESET Field

This field forces a reset of the device if a non-zero value is written. If 0xA5 is written to this field, the device resets into bootloader mode.

Write value: Non-zero (normal) or 0xA5 (bootloader)

BACKUPNV Field

This field supports the commands in [Table 3-4](#):

TABLE 3-4: BACKUPNV COMMANDS

Command	Description
0x55	Backs up the settings to the Non-volatile Memory (NVM). When the Dynamic Configuration Controller T70 object is enabled, this command clears the lock on events and resumes the Dynamic Configuration Controller T70 object.
0x33	Stops the Dynamic Configuration Controller T70 object and restores the settings from the NVM.
0x44	Restores the settings from the NVM without stopping the Dynamic Configuration Controller T70 object.

Once the device has processed this command it generates a status message containing the new NVM checksum.

CALIBRATE Field

This field performs a global recalibration on all mutual-capacitance channels. If all the channels are disabled, no message is generated. If the device is in Deep Sleep mode (see [Section 3.4.1 "Introduction"](#)), a message is generated when the device wakes from sleep.

Write value: Non-zero

REPORTALL Field

This field forces all objects that generate messages to report their current status:

- For optional objects, this applies only if they have their report enable bit set and are currently enabled.
- For objects that are always present and generate messages setting this bit will always cause a status message to be reported.

This field is cleared once the command has been processed. If the device is in Deep Sleep mode (see [Section 3.4.1 "Introduction"](#)), the REPORTALL command will be processed once the device wakes from sleep.

Write value: Non-zero

DEBUGCTRL Field

This field generates device debugging data using the hardware debug interface. Refer to QTAN0050, *Using the maXTouch Debug Port*, for more details.

Note that to use this feature, the circuit has to be designed accordingly. Refer to the *mXT144U 1.0 Data Sheet* for more information.

CHIP: Enables the sending of the Command Processor general status byte.

TSCRN: Enables the sending of the touchscreen data (for example, XY positions, states, and so on) for each supported touch.

DELTAS8: Enables the sending of 8-bit delta values for all nodes.

DELTAS: Enables the sending of 16-bit delta values for all nodes.

REFS: Enables the sending of each channel reference value.

SIGNAL: Enables the sending of each channel signal value.

DIAGNOSTIC Field

This field allows commands to be written to control the use of the Diagnostic Debug T37 object (see [Section 2.2.1 "Introduction"](#)). Specifically, it allows the pages of diagnostic debug data to be navigated and sets the data mode. This field is cleared once the command has been processed. The host can poll this field, for example, to determine that a page change has been actioned and that the requested data is now valid.

The valid commands are listed in [Table 3-5](#).

TABLE 3-5: DIAGNOSTIC DEBUG COMMANDS

Command	Name	Description
0x01	Page Up	Increment page number.
0x02	Page Down	Decrement page number.
Other Value	Debug data mode	The Diagnostic Debug T37 object holds the debug data for a specific debug mode. See Section 2.2 "Diagnostic Debug T37 Object" for details of the system debug modes and the commands to access them. See individual objects for details of the object-specific debug modes and the commands to access them.

Note: Changing the mode resets the page number to zero.

DEBUGCTRL2 Field

DBGOBJMODEEN: Enables object specific debugging data using the hardware debug interface. Refer to QTAN0050, *Using the maXTouch Debug Port*, for more details. If this bit is set to 1, other objects can output their specific debug information as required.

Note that to use this feature, the circuit has to be designed accordingly. Refer to the *mXT144U 1.0 Data Sheet* for more information.

3.3.3 MESSAGES

The message data for the Command Processor T6 object is shown in [Table 3-6](#).

**TABLE 3-6: MESSAGE DATA FOR COMMAND PROCESSOR T6
(GEN_COMMANDPROCESSOR_T6)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	RESET	OFL	SIGERR	CAL	CFGERR	COMSERR	Reserved	
2 – 4	CHECKSUM	Configuration settings checksum							

STATUS Field

Reports the current status and flags errors. A bit is set to indicate the corresponding status/error. Note that there may be more than one status/error reported.

CFGERR, CAL, SIGERR and OFL report ongoing status and error conditions, so once these status/error conditions have terminated, a further message is sent with the appropriate bit cleared.

COMSERR and RESET are one-off reports indicating already terminated conditions. These error conditions do not generate a further message with a cleared bit.

COMSERR: There is an error with the communications checksum. This error bit is set when the device is being used in communications checksum mode and there has been a checksum error on the bytes that have been written to the device. Note that if there is a checksum error after a write, then the data will still have been written to the device. It is the host's responsibility to take corrective action.

CFGERR: There is a configuration error in one or more of the enabled objects. The device pauses its processing and generates a status message every 200 ms. Note that the device will stop scanning for touches while the error persists.

See [Section 1.7 “Configuration Checks”](#) for more information on configuration checks.

NOTE	It is possible to execute a backup command while the device is in this error state.
-------------	---

CAL: The device is calibrating. Note that CAL messages may not be generated if other objects (for example, Acquisition Configuration T8) disable these.

SIGERR: There was an error in the acquisition. This error should not normally be seen.

OFL: The acquisition and processing cycle length (see [“IDLEACQINT and ACTVACQINT Fields”](#) and [“IDLEACQINT and ACTVACQINT Fields”](#)) has overflowed the desired power mode interval. The OFL flag is not updated in Free-run or Deep Sleep modes. This message can be suppressed by setting the OVFRPTSUP bit in the Power Configuration T7 CFG field (see [Section 3.4.2 “Configuration”](#)).

RESET: The device has reset.

CHECKSUM Field

Reports the checksum of the configuration settings held in the non-volatile memory. See [Appendix B “Checksum Calculation”](#) for details on how to calculate the checksum.

3.4 Power Configuration T7 Object

3.4.1 INTRODUCTION

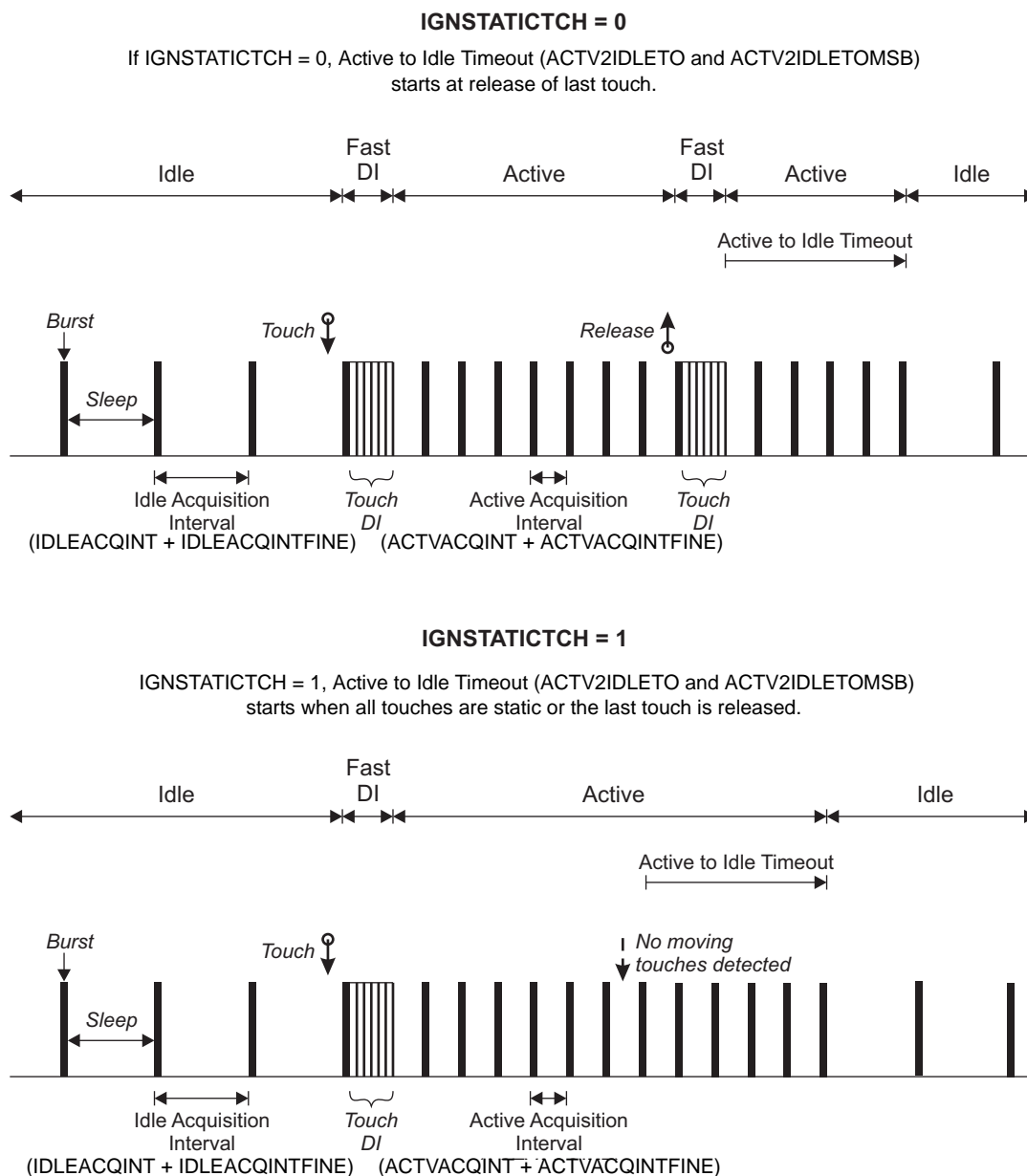
The Power Configuration T7 object controls the active and idle (sleep) times of the device. Power consumption can be lowered by controlling the acquisition frequency and sleep times between measurements.

The device operates in two modes: active (touch detected) and idle (no touches detected).

The normal state for the device is idle mode. In this mode the device operates in a series of long burst cycles. Each cycle consists of a short burst (during which measurements are taken to detect a possible touch) followed by an inactive sleep period.

Figure 3-1 shows the power modes for a Multiple Touch Touchscreen T100 object.

FIGURE 3-1: POWER MODE FIELDS – MULTIPLE TOUCH TOUCHSCREEN T100 OBJECT



When the user touches a touchscreen, the device enters a Fast Detect Integration (Fast DI) mode (see the TCHDI field in the touch objects for an explanation of detect integration). This consists of a series of quick, short bursts to confirm that a change in the touch state has indeed occurred. The number of bursts is determined by the TCHDI field in the touch objects. If it is a genuine touch, the device enters active mode. In this mode the device operates in a series of burst cycles that intersperse measurement bursts with very short sleep periods. These sleep periods are typically shorter than those in idle mode.

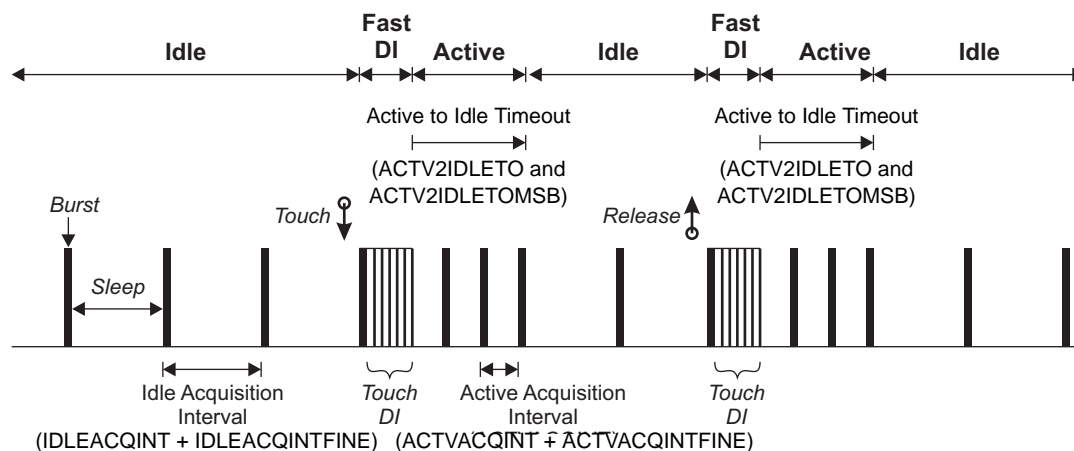
If $IGNSTATICTCH = 0$, when the user releases the touch, the device again enters a short Fast DI mode to confirm that a change in the touch state has occurred. If there are other touches still on the screen, the device will stay in active mode. If this was the last touch, the device will stay in active mode for a short timeout period (Active to Idle Timeout), and drop to idle mode after this timeout has expired.

If $IGNSTATICTCH = 1$, when all touches have been static for a short timeout period (Active to Idle Timeout), the device will switch over to idle mode. If the timeout is not reset by a moving or a new touch then, when the user releases the static touch, the device again enters a short Fast DI mode to confirm that a change in the touch state has occurred, immediately followed by the idle mode.

For a Key Array T15 object, the active to idle timeout (and the subsequent Idle mode) applies on a touchdown as well as a release, as in [Figure 3-2](#).

NOTE The changes to the cycle time happen regardless of whether the touch object is reporting or not.

FIGURE 3-2: POWER MODE FIELDS – KEY ARRAY T15 OBJECT



The Power Configuration T7 object also controls whether the acquisition and subsequent processing are pipelined or not. The normal sequence of events during an acquisition cycle is that data is acquired, the data is processed and then the device sleeps for the remainder of the cycle. When data acquiring is pipelined with the data processing, the processing is postponed until such time as the next data acquisition is performed. This means that the processing of one acquisition interval is done in parallel with the acquisition of data for the next acquisition interval. This can significantly increase the throughput of data when in Free-run mode.

Pipelining of the active and idle modes is controlled separately by the ACTVPIPEEN and IDLEPIPEEN bits in the CFG field.

3.4.2 CONFIGURATION

**TABLE 3-7: CONFIGURATION FOR POWER CONFIGURATION T7
(GEN_POWERCONFIG_T7)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	IDLEACQINT	Idle acquisition interval							
1	ACTVACQINT	Active acquisition interval							
2	ACTV2IDLETO	Active to idle time out							
3	CFG	INITACTV	OVFRPTSUP	ACTV2IDLETOMSB				ACTVPIPEEN	IDLEPIPEEN
4	CFG2	Reserved							IGNSTATICTCH
5	IDLEACQINTFINE	Fine idle acquisition interval							
6	ACTVACQINTFINE	Fine active acquisition interval							

IDLEACQINT and ACTVACQINT Fields

The length of the idle and active burst cycles is determined by the Idle Acquisition Interval (IDLEACQINT) and the Active Acquisition Interval (ACTVACQINT) fields respectively in combination with the IDLEACQINTFINE and ACTVACQINTFINE fields respectively (see “IDLEACQINTFINE and ACTVACQINTFINE Fields”). The IDLEACQINT/ACTVACQINT field specifies the millisecond part of the interval and the IDLEACQINTFINE/ACTVACQINTFINE field specifies the microsecond part. See [Figure 3-1](#) and [Figure 3-2](#).

A setting of 255 in IDLEACQINT/ACTVACQINT forces the device to enter Free-run mode the next time that the appropriate mode (idle or active) is entered. In Free-run mode the device does not sleep between acquisitions. This gives the fastest response time at the expense of increased power consumption.

A setting of zero forces the device to enter Deep Sleep mode the next time that the appropriate mode (idle or active) is entered. The device remains in Deep Sleep mode until the IDLEACQINT or ACTVACQINT setting is restored. Deep Sleep mode is used to minimize power consumption if the device does not need to be sensing. If Deep Sleep mode is requested, it is advisable to set both IDLEACQINT and ACTVACQINT to zero to avoid indeterminate behavior if one mode is still active (note that, in this case, IDLEACQINTFINE and ACTVACQINTFINE will be ignored). The status flags in the Command Processor T6 (see [Section 3.3.2 “Configuration”](#)) are not updated when the device is in Deep Sleep mode.

Other values for IDLEACQINT or ACTVACQINT specify the millisecond part of the Idle or Active Acquisition Interval, which is added to IDLEACQINTFINE or ACTVACQINTFINE, as appropriate, to form the actual Idle or Active Acquisition Interval that is used. A high value causes more sleep time between acquisitions. This results in lower power consumption but a slower response time.

The Idle and Active Acquisition intervals must not be set to less than the actual burst time. The device is also designed to sleep as much as possible in order to conserve power. The Idle Acquisition Interval should therefore be set longer than Active Acquisition Interval. Under some circumstances it is desirable to set Idle Acquisition Interval lower than Active Acquisition Interval. For example, this might be necessary to minimize the difference between the best-case and the worst-case touchdown latency.

Range: 0 (Deep Sleep), 1 to 254, 255 (Free-run)

IDLEACQINT Typical: 32

ACTVACQINT Typical: 16

ACTV2IDLETO Field and CFG ACTV2IDLETOMSB Bits

The device automatically goes into idle mode whenever possible after each matrix scan to conserve power, unless a touch object is being touched (see [Figure 3-1](#)).

The device does not go into idle mode immediately. Instead there is a timeout period. The device runs in active mode during this timeout period to allow further touches to keep the device active. The timeout period is determined by the Active to Idle Timeout setting. This is a 12-bit value with the 4 MSBs in the ACTV2IDLETOMSB bits of the CFG field and the 8 LSBs in the ACTV2IDLETO field.

Under normal operation, the device enters idle mode after the expiry of the Active to Idle Timeout and then remains in idle mode until the next touch is detected. If there is more than one touch present, the Active to Idle Timeout applies only after the last touch has been released. This means that once the device has been awakened by a change, the touch response time is fast for as long as the sensor remains in use. Once channel activity lapses for a period longer than the Active to Idle Timeout, the device returns to idle mode.

The Active to Idle Timeout is specified in 200 ms increments, where 0 means 1 cycle.

Range: 0 (1 cycle), 1 to 4095

Typical: 50

CFG Field

IDLEPIPEEN: Enables pipelining in idle mode. Pipelining is enabled if set to 1 and disabled if set to 0.

ACTVPIPEEN: Enables pipelining in active mode. Pipelining is enabled if set to 1 and disabled if set to 0.

ACTV2IDLETOMSB: Specifies the four msbits of the Active to Idle Timeout. See [“ACTV2IDLETO Field and CFG ACTV2IDLETOMSB Bits”](#) for details].

OVRFRPTSUP: If this bit is set, overflow messages are suppressed (and a pending overflow message will be cleared).

INITACTV: Selects whether the chip starts in active mode or idle mode after a power-up or reset. If set to 1 then the chip starts in active mode and remains in active mode until the Active to Idle Timeout expires (as set by the ACTV2IDLETO Field and the CFG ACTV2IDLETOMSB bits). If INITACTV is set to 0, then the device starts in idle mode.

CFG2 Field

IGNSTATICTCH: Under normal operation (that is, when this bit is set to 0), the Active to Idle Timeout is refreshed while any touch is active (either moving or static). This means that a static touch will keep the object in active mode and the Active to Idle Timeout expires only after the last touch has been released (see [Figure 3-1](#)).

If this bit is set to 1, then static touches do not cause the Active to Idle Timeout to reset and the timeout expires once all tracked touches are either static or released (see [Figure 3-1](#)).

This feature trades off subsequent touch up/down latency (but not first touch down latency) against power saving.

IDLEACQINTFINE and ACTVACQINTFINE Fields

These fields fine tune the Idle and Active Acquisition Interval. They specify an additional interval (in units of 10 μ s) that is added to the values specified by the IDLEACQINT and ACTVACQINT fields to achieve a more exact refresh rate.

Range: 0 to 255 (effective range: 0 to 99)

3.5 Acquisition Configuration T8 Object

3.5.1 INTRODUCTION

The Acquisition Configuration T8 object controls how the device takes capacitive measurements.

3.5.1.1 Calibration Recovery Process

[Figure 3-3](#) shows an overview of the recalibration processes on the mXT144U controlled by the Acquisition Configuration T8 object.

FIGURE 3-3: RECALIBRATION PROCESSES

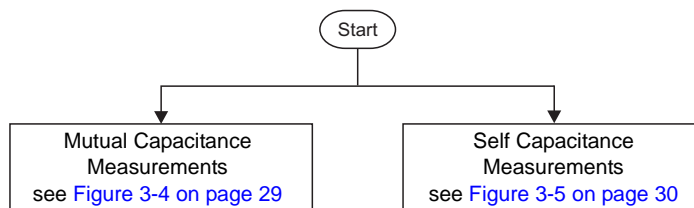
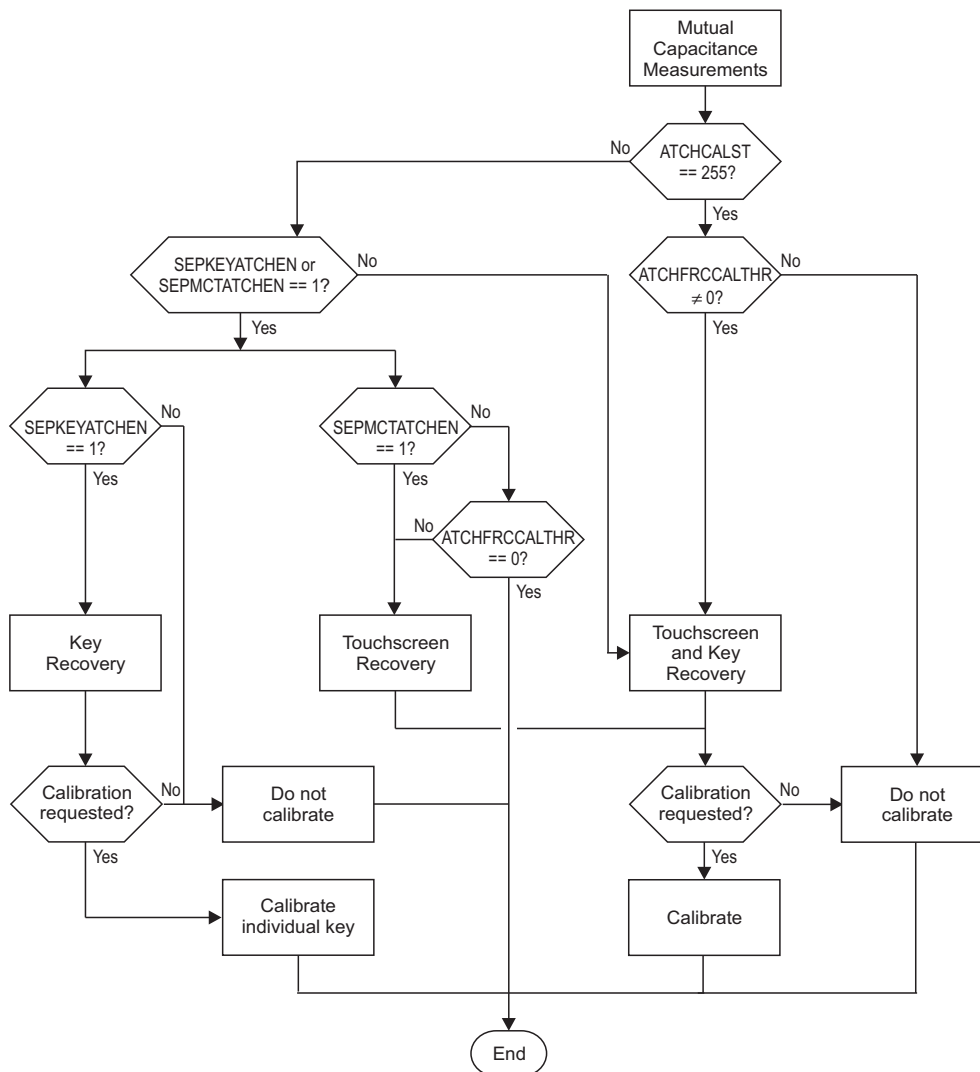
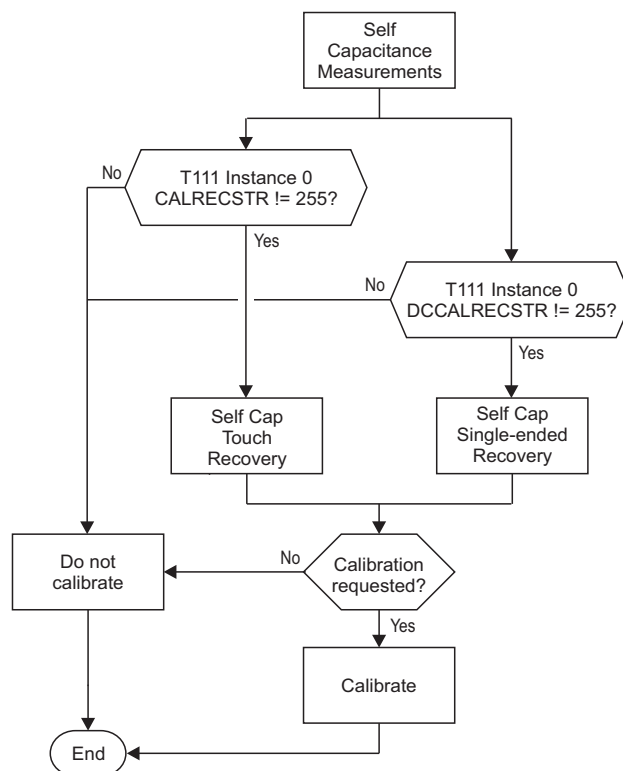


FIGURE 3-4: RECALIBRATION PROCESS – MUTUAL CAPACITANCE



The mutual capacitance recovery process is described in more detail in [Section 3.5.1.2 “Mutual Capacitance Recovery Process”](#).

FIGURE 3-5: RECALIBRATION PROCESS – SELF CAPACITANCE

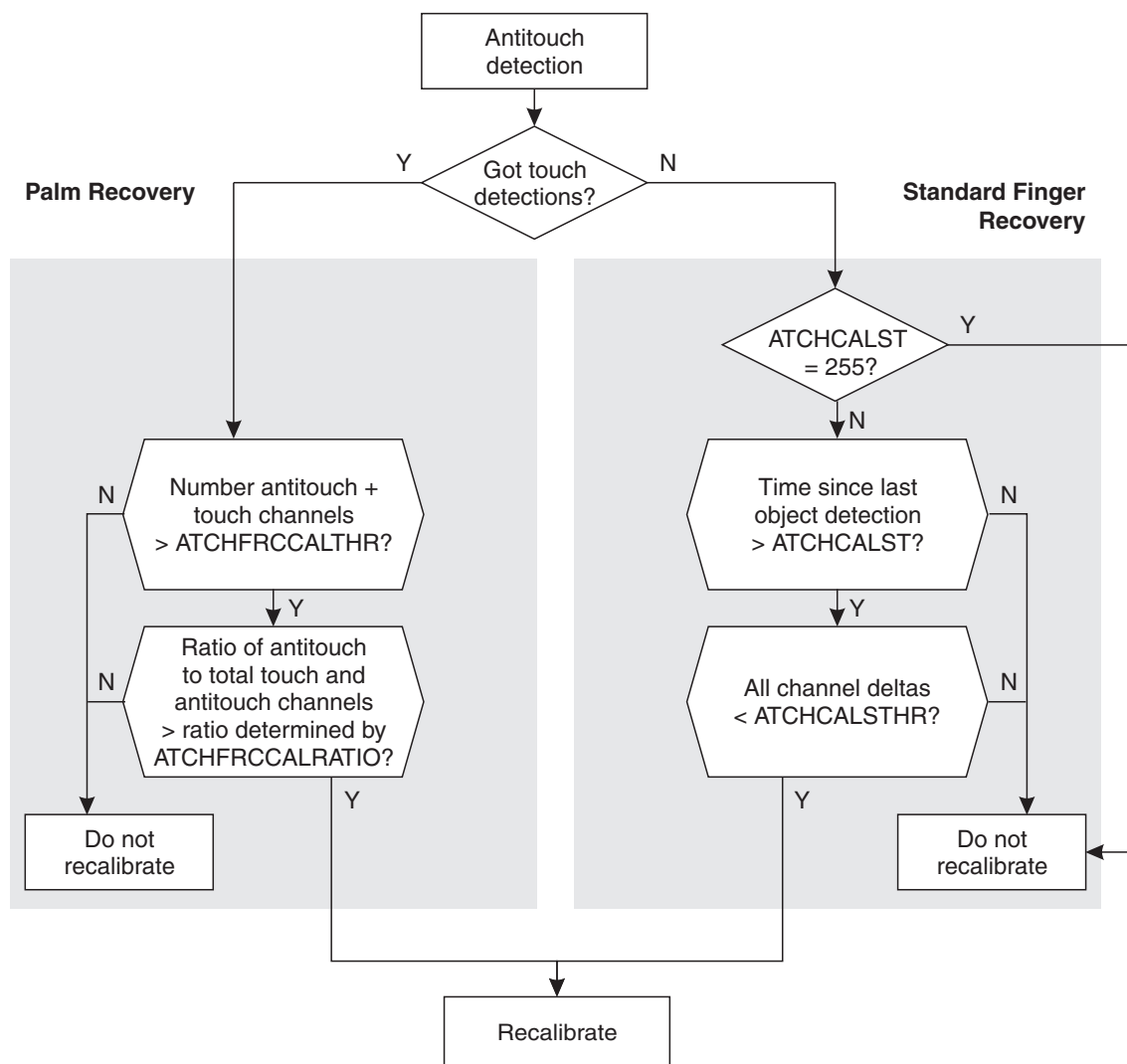


The self capacitance recovery process is controlled by the Self Capacitance Configuration T111 object (see [Section 6.14 “Self Capacitance Configuration T111 Object”](#)).

3.5.1.2 Mutual Capacitance Recovery Process

The mutual capacitance touch recovery process is shown in [Figure 3-6](#).

FIGURE 3-6: TOUCH RECOVERY PROCESSES



The standard finger mutual capacitance recovery process is intended to allow the sensor to recover when a finger is present on the sensor during calibration and is then subsequently removed. This process attempts to calibrate the sensor when only antitouches are detected on the sensor (see [Figure 3-6](#)). The ATCHCALST and ATCHCALSTHR fields allow this process to be blocked under certain circumstances (see [“ATCHCALST Field”](#) and [“ATCHCALSTHR Field”](#)).

In addition, the Acquisition Configuration T8 object provides further control over how individual keys are calibrated. See [“CFG Field”](#) for more information on the CFG field SEPKEYATCHEN and SEPMCTATCHEN bits.

A palm touch during calibration results in a complex pattern of both touch and antitouch detections. The standard finger recovery recalibration process, however, is blocked by any channels in touch, so a recalibration would never occur. The mutual capacitance palm recovery process therefore allows the sensor to recover when a palm is present on the sensor during calibration and is then subsequently removed. Palm recovery process is controlled by the ATCHFRCCALTHR and ATCHFRCCALRATIO fields (see [“ATCHFRCCALTHR and ATCHFRCCALRATIO Fields”](#) and [“ATCHFRCCALTHR and ATCHFRCCALRATIO Fields”](#)).

3.5.1.3 Self Capacitance Calibration and Recovery Processes

The self capacitance recovery process is controlled by the CALRECSTR field in the Self Capacitance Configuration T111 object. See [“CALRECSTR Field”](#) for more details.

3.5.2 CONFIGURATION

**TABLE 3-8: CONFIGURATION FOR ACQUISITION CONFIGURATION T8
(GEN_ACQUISITIONCONFIG_T8)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CHRGTIME	Charge-transfer dwell time							
1	Reserved	Reserved							
2	TCHDRIFT	Touch drift							
3	DRIFTST	Drift suspend time							
4	TCHAUTOCAL	Touch automatic calibration							
5	Reserved	Reserved							
6	ATCHCALST	Antitouch calibration suspend time							
7	ATCHCALSTHR	Antitouch calibration suspend threshold							
8	ATCHFRCCALTHR	Antitouch force calibration threshold							
9	ATCHFRCCALRATIO	Antitouch force calibration ratio							
10	MEASALLOW	Reserved				SELFPROX	Reserved	SELFTHCH	MUTUALTHCH
11	MEASIDLEDEF	Reserved				SELFPROX	Reserved	SELFTHCH	MUTUALTHCH
12	MEASACTVDEF	Reserved						SELFTHCH	MUTUALTHCH
13	REFMODE	Reference mode							
14	CFG	CALRECTUN	Reserved				SEPMCTATCHEN	SEPKEYATCHEN	DISCALRPT

CHRGTIME Field

This setting controls the charge-transfer dwell time. It is specified in units of 41.67 ns (24 MHz clock cycles). Higher charge times result in a slower scan time. The maximum for this field is 255 (10.625 μ s). The default value of 0 means 48 (2 μ s).

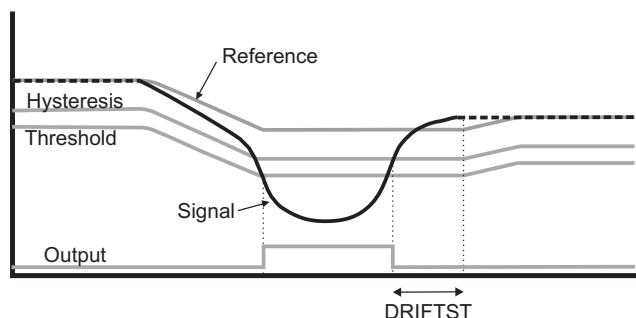
Range: 0 (48 = 2 μ s), 1 to 255 (in 41.67 ns increments)

TCHDRIFT Field

The Touch Drift (TCHDRIFT) setting controls the drift interval for mutual capacitance measurements on the touchscreen sensor array. The drift intervals for any other measurement types (if present) are configured in other objects.

Signals can drift because of changes in the nature of the components and materials over time and temperature. It is crucial that such drift is compensated for, otherwise false detections and sensitivity shifts can occur.

Drift compensation (see [Figure 3-7](#)) is performed by making the reference level track the raw signal at a slow rate, but only while there is no detection in effect. The rate of adjustment must be slow, otherwise legitimate detections could be ignored. The TCHDRIFT field can be configured in increments of 200 ms, where a value of 0 means 10 (2 seconds).

FIGURE 3-7: THRESHOLDS AND DRIFT COMPENSATION

The device compensates for drift by using a slew-rate limited change to the reference level. The threshold and hysteresis values are slaved to this reference. Any changes to TCHDRIFT affect the rate. The maximum steps per update is fixed at 2; that is, the drift compensation can drift at a rate of up to 2 steps once every (TCHDRIFT × 200) ms.

Once the touch signal has crossed the threshold for the current touch classification (see [Appendix A.3 "Touch Classification"](#)) on the appropriate touch object, the drift compensation mechanism is suspended.

Range: 0 (10 = 2 seconds), 1 to 254 (in 200 ms increments)

Typical: 20

DRIFTST Field

The Drift Suspend Time (DRIFTST) setting for mutual capacitance measurements controls the time from a touch release on a touch object until the drift process is re-enabled. DRIFTST is used to restrict drift on all channels while one or more mutual-capacitance channels are activated. It defines the length of time the drift is halted after a touch detection.

This feature is particularly useful in preventing an actual touch – or simply a hovering finger – from causing untouched channels to drift. Without this feature, a sensitivity shift could be created that would ultimately inhibit any further touch detection.

DRIFTST can be configured to a value of between 0 and 255 in increments of 200 ms, where a value of zero means 20 (4 seconds). This gives a range of 200 ms to 51s.

Range: 0 (20 = 4 seconds), 1 to 255 (in 200 ms increments)

TCHAUTOCAL Field

A prolonged (usually unintentional) contact from a foreign object may result in a mutual capacitance touch detection for a prolonged interval. It is desirable to perform a recalibration in order to restore a touch object's function. This is usually done after a time delay of some seconds.

The Touch Automatic Calibration (TCHAUTOCAL) setting controls the length of time a touch is held until it is considered false and an automatic recalibration is performed to compensate.

The TCHAUTOCAL timer monitors touch detections. If a detection event exceeds the timer setting, an automatic recalibration occurs. After the recalibration has taken place, normal functionality resumes, even if the touch object is still being contacted by the foreign object. This feature is enabled globally, but the exact mechanism depends on the object being touched:

- For a Multiple Touch Touchscreen T100 object (see [Section 4.3 "Multiple Touch Touchscreen T100 Object"](#)) there is a counter per touch, incremented while the touches remain stationary. If a touch is stationary at the end of the TCHAUTOCAL period an automatic recalibration occurs. A touch is considered stationary if it moves less than 1 node (in either axis) within the TCHAUTOCAL period. Note that the entire sensor matrix is calibrated, and not just the area occupied by the Multiple Touch Touchscreen T100 object that triggered the recalibration.
- For a Key Array T15 object (see [Section 4.2 "Key Array T15"](#)) there is a counter per key, incremented while a touch remains on the key. An automatic recalibration occurs if the touch is still present at the end of the TCHAUTOCAL period. Note that a touch detection within a key does not clear the reset counters for any other keys, or the reset counters for any other touch objects.

NOTE	Recalibration of a Key Array T15 does not calibrate the entire sensor matrix, only that part occupied by touched keys. Recalibration of the Multiple Touch Touchscreen T100, however, does calibrate the entire sensor matrix.
-------------	--

TCHAUTOCAL can be disabled by setting it to zero (infinite timeout). In this case the object never automatically recalibrates during a continuous detection (but the host could still command it). TCHAUTOCAL above 0 is configured in 200 ms increments.

NOTE	TCHAUTOCAL should not be used if Retransmission Compensation T80 is enabled.
-------------	--

Range: 0 (infinite), 1 to 255 (in 200 ms increments)

ATCHCALST Field

The ATCHCALST field sets the antitouch calibration suspend time for use in the touch recovery process (see [Section 3.5.1.2 “Mutual Capacitance Recovery Process”](#)). This is the time from the last touch release to when a standard finger recovery recalibration can occur. ATCHCALST can be configured to a value of between 0 and 254 (0 and 51s), in increments of 200 ms. A setting of zero allows the recalibration logic to work immediately after the last finger is removed from the sensor. A value of 255 disables the mutual capacitance finger recovery process.

If the device is configured to use mutual capacitance measurements only, it is recommended to set ATCHCALST to zero, as other values can affect the ability of the device to recover from a bad calibration.

Note that the antitouch calibration suspend time should not be set to a period longer than the drift suspend time (DRIFTST).

NOTE If the device is configured to use mutual capacitance measurements only (that is, MEASALLOW MUTUALTCH = 1), the Antitouch Calibration Suspend Time feature should not be used if the Retransmission Compensation T80 object is enabled. ATCHCALST should therefore be set to 255 if the Retransmission Compensation T80 object is enabled.

Range: 0 to 254 (in 200 ms increments), 255 (mutual capacitance finger recovery disabled)

Typical: 0 (immediate)

ATCHCALSTHR Field

The ATCHCALSTHR field sets the antitouch calibration suspend threshold. Any channel with a touch delta above this threshold will suspend all standard finger recovery recalibrations. If this field is set to zero, the standard finger recovery calibration is never blocked except by the ATCHCALST period.

Note that if the ATCHCALST field is set to a value of 255, the mutual capacitance finger recovery process is disabled and the ATCHCALSTHR field has no effect.

See also [Section 3.5.1.1 “Calibration Recovery Process”](#).

Range: 0 to 255

Typical: 0 (never suspend)

ATCHFRCCALTHR and ATCHFRCCALRATIO Fields

These two fields control the mutual capacitance palm recovery process that allows the sensor to recover when a palm is present on the sensor during calibration and is then subsequently removed.

Note that this field controls the palm recovery process for mutual capacitance measurements; for the calibration recovery process for self capacitance measurements, see [“CALRECSTR Field”](#).

A palm touch during calibration results in a complex pattern of both touch and antitouch detections. The standard finger recovery recalibration process (see the ATCHCALSTHR and ATCHCALST fields), however, is blocked by any channels in touch, so a recalibration would never occur. These two fields ensure that this block is overridden and that a palm recovery calibration occurs (see [Figure 3-6](#)).

The ATCHFRCCALTHR field sets the threshold (in channels) to allow a palm recovery calibration. If the number of channels in touch or antitouch¹ is greater than or equal to ATCHFRCCALTHR, then the ATCHFRCCALRATIO field is used to determine if a recalibration should occur. Setting the ATCHFRCCALTHR field to zero disables both that field and the ATCHFRCCALRATIO field and a recalibration will not occur.

The ATCHFRCCALRATIO field determines the ratio of antitouch channels to total touch and antitouch channels that must be met for a recalibration to occur. This field takes a signed value that represents the desired ratio (see [Table 3-9](#)). Note that negative values should be avoided as they risk rogue calibrations when a palm is placed over the sensor.

1. A touch channel is one in that is above +TCHTHR (touch threshold). An antitouch channel is one that is below –TCHTHR. See [“TCHTHR Field”](#) for more information on the TCHTHR field.

TABLE 3-9: TYPICAL VALUES FOR ATCHFRCCALRATIO

Value	Meaning
-128 to 0	Reserved; only use a value of zero or less if advised to do so by Microchip.
+1 to 127	Antitouch channels must be greater than 50 percent total (touch + antitouch) channels (where 127 = 100 percent total channels)

ATCHFRCCALTHR Range: 0 (mutual capacitance palm recovery disabled), 1 to 255

ATCHFRCCALTHR Typical: 50

ATCHFRCCALRATIO Range: -128 to +127

ATCHFRCCALRATIO Typical: 25 (60 percent)

MEASALLOW Field

This field defines which measurement types are allowed in the user's product. One or more types of touches can be set using the bits in this field. A value of 0 means 1; that is, mutual capacitance touches are always allowed.

MUTUALTCH: If this bit is set to 1, mutual capacitance touches are allowed.

NOTE	Mutual capacitance touches are always allowed, so setting this bit to 0 has the same effect as setting it to 1.
-------------	---

SELFTCH: If this bit is set to 1, self capacitance touches are allowed.

SELFPROX: If this bit is set to 1, self capacitance single-ended measurements are allowed for use by the Self Capacitance Measurement Configuration T113 object. Note that if SELFPROX is enabled, then the SELFTCH bit must also be enabled.

MEASIDLEDEF Field

This field defines the default measurement type that is to be used in idle mode; that is, the default mode prescribes which data the Multiple Touch Touchscreen T100 processes during normal operation.

MUTUALTCH: If this bit is set to 1, the default measurement type is mutual capacitance in idle mode.

SELFTCH: If this bit is set to 1, the default measurement type is self capacitance in idle mode. Note that if the default measurement type is set to SELFTCH, then the Auxiliary Touch Configuration T104 object must be enabled for use. Otherwise a configuration error results.

SELFPROX: If this bit is set to 1, the default measurement type is self capacitance single-ended in idle mode (that is, low power idle mode). Note that if the default measurement type is set to SELFPROX, then the Self Capacitance Measurement Configuration T113 object must be enabled for use. Otherwise a configuration error results.

Range: See [Table 3-10](#)

TABLE 3-10: MEASIDLEDEF VALID VALUES

MEASIDLEDEF			Measurements Run Every Cycle in Idle			Comments
SELFPROX	SELFCTCH	MUTUALCTCH	SELFPROX	SELFCTCH	MUTUALCTCH	
0	0	0	Yes	Yes	Yes	Default Self capacitance touch scans (SCT) occur every cycle if an object that requires SCT deltas is enabled (for example, Retransmission Compensation T80). Otherwise SCT scans occur occasionally for drift purposes.
0	0	1	Yes	Yes	Yes	SCT scans occur every cycle if an object that requires SCT deltas is enabled (for example, Retransmission Compensation T80). Otherwise SCT scans occur occasionally for drift purposes.
0	1	0	Yes	Yes	–	
1	0	0	Yes	–	–	Low power idle
Other combinations reserved			Not defined			

MEASACTVDEF Fields

This field defines the default measurement type that is to be used in active mode; that is, the default mode prescribes which data the Multiple Touch Touchscreen T100 processes during normal operation.

Note the following:

- Only one bit must be set at any one time.
- Note that if MEASACTVDEF has SELFCTCH set then both axes of self capacitance measurements need to be enabled (see “CTRL Field” and [Appendix A “Measurement Processing on mXT144U”](#)) for the Multiple Touch Touchscreen T100 to be able to track touches.

MUTUALCTCH: If this bit is set to 1, the default measurement type is mutual capacitance in active mode.

SELFCTCH: If this bit is set to 1, the default measurement type is self capacitance in active mode. Note that if the default measurement type is set to SELFCTCH, then the Auxiliary Touch Configuration T104 object must be enabled for use. Otherwise a configuration error results.

Range: 0 (1 = mutual capacitance), 1 and 2

REFMODE Field

This field defines the reference mode that the system will use (see [Table 3-11](#)).

TABLE 3-11: REFERENCE MODES

REFMODE Value	Description
0	The system will be running with one set of references. Do not set to 0 unless advised by Microchip.
1	The system will be running with two sets of references (required if the device is using Dual X mode). This is the recommended value.
2 – 255	Reserved

Range: 0 to 1 (1 is the recommended value)

CFG Field

DISCALRPT: Disables reporting of Acquisition Configuration T8 auto calibration messages. If this bit is set to 1, generation of a Command Processor T6 message on a recalibration event that is generated by the touch or antitouch mechanisms of the Acquisition Configuration T8 object will be suppressed. The recalibration itself will still happen, but it will not be flagged.

SEPKEYATCHEN: Enables Separate Key Antitouch Calibration. The effect of this field depends on ATCHCALST (see “[ATCHCALST Field](#)”). If ATCHCALST is set to a value less than 255, the effect of SEPKEYATCHEN is as shown in [Table 3-12](#). If ATCHCALST is set to 255, mutual capacitance finger recovery is disabled and SEPKEYATCHEN has no effect (although palm recovery still applies).

SEPMCTATCHEN: Enables Separate Mutual Capacitance Antitouch Calibration. The effect of this field depends on ATCHCALST (see “[ATCHCALST Field](#)”). If ATCHCALST is set to a value less than 255, the effect of SEPMCTATCHEN is as shown in [Table 3-12](#). If ATCHCALST is set to 255, mutual capacitance finger recovery is disabled and SEPMCTATCHEN has no effect (although palm recovery still applies).

TABLE 3-12: SEPMCTATCHEN AND SEPKEYATCHEN (ATCHCALST < 255)

SEPMCTATCHEN	SEPKEYATCHEN	Effect on Touchscreen	Effect on Keys [†]
1	1	Mutual capacitance touchscreen calibration calibrates the entire sensor matrix, but touched keys are ignored even though keys are calibrated with the touchscreen.	Keys calibrate individually without causing a touchscreen calibration.
0	1	Mutual capacitance touchscreen finger recovery is disabled.	Keys calibrate individually without causing a touchscreen calibration.
1	0	Mutual touchscreen calibration calibrates the entire sensor matrix, but touched keys are ignored even though keys are calibrated with the touchscreen.	Mutual capacitance key recovery is disabled.
0	0	Mutual calibration takes touchscreen and keys into account equally.	Keys are calibrated with the touchscreen only. *

[†] Applies to the Key Array T15 object only.

* This configuration option equates to the behavior on previous maXTouch devices.

CALRECTUN: Enables Calibration Recovery Tuning Data. If this bit is set to 1, tuning data for all calibration recovery mechanisms is output via the Diagnostic Debug T37 and Command Processor T6 objects (see [Section 2.2 “Diagnostic Debug T37 Object”](#) and “[DIAGNOSTIC Field](#)”). The tuning and debug data is described in [Section 3.5.4 “Diagnostic Debug Data”](#).

3.5.3 CONFIGURATION CHECKS

The Acquisition Configuration T8 object causes a configuration check to be performed in the following circumstances:

- When certain fields are changed

In addition, some fields may cause an automatic recalibration to be performed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

TABLE 3-13: CONFIGURATION CHECKS FOR ACQUISITION CONFIGURATION T8 (GEN_ACQUISITIONCONFIG_T8)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CHRGTIME	No	Yes	None
TCHDRIFT	No	No	None
DRIFTST	No	No	None
TCHAUTOCAL	No	No	None

**TABLE 3-13: CONFIGURATION CHECKS FOR ACQUISITION CONFIGURATION T8
(GEN_ACQUISITIONCONFIG_T8)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
ATCHCALST	No	No	None
ATCHCALSTHR	No	No	None
ATCHFRCCALTHR	No	No	None
ATCHFRCCALRATIO	No	No	None
MEASALLOW	Yes	Yes	None
MEASIDLEDEF	Yes	No	Error if one of the following is true: <ul style="list-style-type: none"> • Illegal value • SELFTCH = 1 and Auxiliary Touch Configuration T104 not enabled • SELFPROX = 1 and Self Capacitance Measurement Configuration T113 not enabled
MEASACTVDEF	Yes	No	Error if one of the following is true: <ul style="list-style-type: none"> • Illegal value • SELFTCH = 1 and Auxiliary Touch Configuration T104 not enabled
REFMODE	Yes	Yes	Error if REFMODE = 0 and Noise Suppression T72 uses both dual x and non dual X within noise states.
CFG	No	No	None

3.5.4 DIAGNOSTIC DEBUG DATA

The diagnostic debug data modes for the Acquisition Configuration T8 object is shown in [Table 3-14](#). The diagnostic debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in [Table 3-14](#).

The diagnostic debug modes are described in the following sections.

TABLE 3-14: DIAGNOSTIC DEBUG COMMANDS

T6 Command	Name	Description
0x33	Calibration Recovery Tuning Data Mode	The Diagnostic Debug T37 object holds the Calibration Recovery Tuning Data. This is the tuning data for any enabled calibration recovery mechanisms. Note that the CFG CALRECTUN bit must be set to 1 for this to happen (see "CFG Field"). See Section 3.5.4.1 "Calibration Recovery Tuning Data Mode" .

3.5.4.1 Calibration Recovery Tuning Data Mode

[Figure 3-8](#) shows the format of the data in Calibration Recovery Tuning Data Mode. This reports the tuning data for any enabled calibration recovery mechanisms.

FIGURE 3-8: FORMAT OF THE CALIBRATION RECOVERY TUNING DATA

Data Index		Data[]		
Page 0	0	Acquisition Configuration T8 ATCHCALST counter	Mutual Capacitance Data	
	1	Peak touch delta		
	2	Number of nodes in touch		
	3			
	4	Number of nodes in antitouch		
	5			
	6	Calibration recovery counter		
	7	Calibration recovery state ⁽¹⁾		
	8	Calibration recovery metric	Self Capacitance Touch Data	
	9			
	10	Calibration recovery counter		
	11	Calibration recovery state ⁽¹⁾		
	12	Acquisition Configuration T8 ATCHCALST counter	Separate Key Antitouch Calibration Data	
	13	Calibration recovery counter		
	14	Calibration recovery state ⁽¹⁾		
	15	Calibration recovery metric	Self Capacitance Touch DC Data	
	16			
	17	Calibration recovery counter		
	18	Calibration recovery state ⁽¹⁾		
	19 – 127	Reserved		

Assumes page size = 128

Note 1: Calibration recovery state has one of the following values:
0 = Idle
1 = Counting
2 = Triggered

4.0 TOUCH OBJECTS

4.1 Introduction

Touch objects operate on measured signals from the touch sensor and report touch data. For example, a Multiple Touch Touchscreen T100 object reports XY touch positions. [Table 4-1](#) lists the touch objects on the mXT144U.

TABLE 4-1: TOUCH OBJECTS

Object	Description
Key Array T15 (TOUCH_KEYARRAY_T15)	Creates a rectangular array of keys. A Key Array T15 object reports simple on/off touch information. See Section 4.2 “Key Array T15” .
Multiple Touch Touchscreen T100 (TOUCH_MULTITOUCHSCREEN_T100)	Creates a Touchscreen that supports the tracking of more than one touch. See Section 4.3 “Multiple Touch Touchscreen T100 Object” .

4.2 Key Array T15

4.2.1 INTRODUCTION

A Key Array T15 object is used to configure a rectangular array of XY nodes on the mutual capacitance sensor for use as keys. The Key Array T15 object can be used to configure an array of up to 8 keys.

The key numbers are assigned to X lines in Y–X order, as defined by the following equation:

$$\text{Key number} = (\text{Xline} - \text{XORIGIN}) + ((\text{Yline} - \text{YORIGIN}) \times \text{XSIZE})$$

For example, for a Key Array T15 of 3 X by 2 Y lines, the key numbers are allocated as in [Table 4-2](#).

TABLE 4-2: EXAMPLE KEY NUMBERS

	X_n	X_{n+1}	X_{n+2}
Y_{m+1}	3	4	5
Y_m	0	1	2

4.2.1.1 Touch Threshold

The Key Array T15 object has a Touch Threshold (TCHTHR) that defines how large a node's touch delta (that is, Reference minus the signal.) must be to qualify as a potential touch detection. The reference level is determined during calibration and adjusted using drift compensation. The final detection confirmation uses the Touch Detect Integration as described in the TCHDI field. Larger values for the threshold desensitize nodes since the signal must change more in order to exceed the threshold level. Conversely, lower thresholds make nodes more sensitive.

The setting for TCHTHR for each node depends on the amount of signal swing that occurs when a node is touched. Thicker panels or smaller electrode geometries reduce node gain, that is signal swing from touch. In this case smaller TCHTHR values are required to detect touch.

A Touch Hysteresis (TCHHYST) can also be set for all keys in the key array. Note, however, that this hysteresis is capped at run time to 25% of the current touch threshold for all keys.

Other objects can adjust the minimum threshold in specific circumstances. These objects are:

- Noise Suppression T72 – Supplies a minimum touch threshold that desensitizes the Key Array T15 to avoid reporting any detected noise that would be reported as touches (see [Section 5.8 “Noise Suppression T72 Object”](#)).
- Glove Detection T78 – Can adjust the threshold to allow for glove touches (see [Section 5.9 “Glove Detection T78 Object”](#)).

4.2.2 CONFIGURATION

TABLE 4-3: CONFIGURATION FOR KEY ARRAY T15 (TOUCH_KEYARRAY_T15)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	INTAKSEN	Reserved			TSCONTSUP	Reserved	RPTEN	ENABLE
1	XORIGIN	X line start position of object							
2	YORIGIN	Y line start position of object							
3	XSIZE	Number of X lines the object occupies							
4	YSIZE	Number of Y lines the object occupies							
5	AKSCFG	Group8	Group7	Group6	Group5	Group4	Group3	Group2	Group1
6	BLEN	Gain							
7	TCHTHR	Touch threshold							
8	TCHDI	Touch detect integration							
9	TCHHYST	Touch hysteresis							
10	CFG	Reserved						DBGREFSEN	DBGDELTA

CTRL Field

ENABLE: Enables the use of this Key Array T15 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

TSCONTSUP: If this bit is set to 1, the Touchscreen Contact Suppression algorithm enabled. When this feature is enabled, key reporting for this Key Array T15 instance will be suppressed if there is a contacting touch on any touchscreen when the keys are going into detect. Key reporting will remain suppressed until all keys for this instance of Key Array T15 are released.

In designs where keys are situated near to the touchscreen without a bezel separation, there is a possibility that the user's touch might unintentionally slide onto a key. Even with AKS enabled, the key can still report, once the touch is no longer detected on the touchscreen. In this case, enabling the touchscreen contact suppression can help suppress the key.

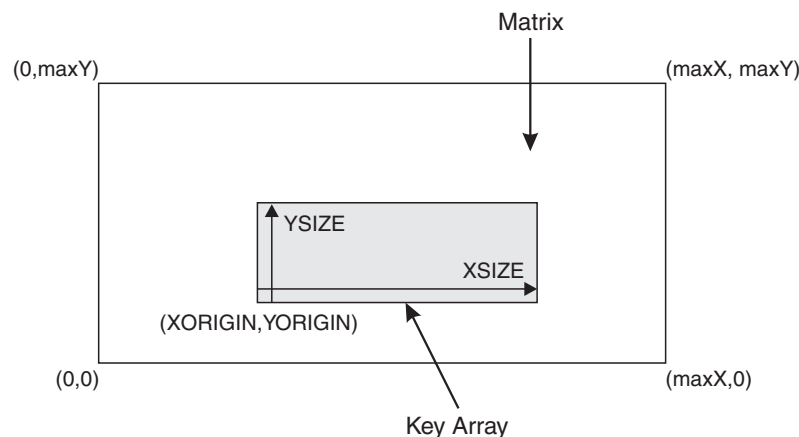
Note that for touchscreen contact suppression to function, the external AKS (see “AKSCFG Field”) between the key array and touchscreen has to be disabled (that is, they cannot be in the same AKS group).

INTAKSEN: Enables the internal Adjacent Key Suppression (AKS) between keys in the array. If internal AKS is enabled, then when one key is touched, touches on all the other keys within the Key Array are suppressed. Set this bit to 1 to enable internal AKS, and to 0 to disable internal AKS.

XORIGIN, YORIGIN, XSIZE and YSIZE Fields

These fields specify the size and position of the key Array on the actual matrix (see Figure 4-1). The XORIGIN and YORIGIN fields specify the origin in X and Y lines. The XSIZE and YSIZE fields specify the size in X/Y nodes (keys).

FIGURE 4-1: KEY ARRAY T15 FIELDS



The minimum size for an enabled key array is 1 X by 1 Y line. The maximum size for a key array is such that when XSIZE and YSIZE are multiplied together they obey the rule: $XSIZE \times YSIZE \leq 8$ nodes. Setting XSIZE and YSIZE to values that create more than 8 keys causes a configuration error.

NOTE The key array must not share X and Y lines with other touch objects (specifically Multiple Touch Touchscreen T100). In addition, the key array must use X and Y lines numbered above those used by the enabled Multiple Touch Touchscreen T100 object.
Refer to the *mXT144U 1.0 Data Sheet* for more information on the layout of the touchscreen sensor.

XORIGIN Range: (T100.XORIGIN + T100.XSIZE) to (maxX – T15.XSIZE + 1)

YORIGIN Range: (T100.YORIGIN + T100.YSIZE) to (maxY – T15.YSIZE + 1)

If Multiple Touch Touchscreen T100 is not enabled, the minimum is considered to be 0.

XSIZE Range: 1 to (8 / YSIZE)

YSIZE Range: 1 to (8 / XSIZE)

AKSCFG Field

This field configures Adjacent Key Suppression (AKS) between this Key Array T15 object and any other Touchscreen or Key Array T15 objects.

AKS technology is a patented method used to detect which touch object is touched when objects are located close together. A touch in a group of AKS objects is indicated only on the object with the largest signal. This is assumed to be the intended object. Once an object in an AKS group is in detect, there can be no further detections within that group until the object is released.

Group1 to 8: These bits form a bit field that specifies which AKS groups this Key Array is within. The default value of 0 means that the object is in no AKS groups and the AKS feature is disabled for the Key Array.

Range: 0 to 255

BLEN Field

This field sets the gain of the analog circuits in front of the analog to digital converter (ADC). Note that despite its name, the BLEN field does not control the burst length.

Range: 0 to 22

TCHTHR Field

This field specifies the Touch Threshold in 8-bit deltas (see [Section 4.2.1 "Introduction"](#) for details).

Note that the Touch Hysteresis specified in the TCHHYST field also has an effect on the TCHTHR setting (See ["TCHHYST Field"](#)) and should therefore be allowed for in the setting.

Range: 2 to 255

TCHDI Field

To suppress false detections caused by spurious events like electrical noise, the device incorporates a detection integrator (TCHDI) counter mechanism to provide signal filtering. A per-key counter is incremented each cycle that a touch is detected. When this counter reaches a preset limit the touch is finally declared to be present. If on any acquisition a delta is not seen to exceed the threshold level, the counter is cleared and the process has to start from the beginning.

It takes TCHDI + 1 cycles from touchdown to when the first touch is actually reported via the $\overline{\text{CHG}}$ pin, with a minimum time of 2 cycles. Once a touch is detected, the limit is changed to TCHDI cycles.

An opposite process is applied when a key leaves detection. The counter is decremented each cycle that the delta does not exceed the threshold level, and incremented again if it does exceed the threshold. When the counter reaches zero, the touch is finally declared to be out of detect. It takes TCHDI + 1 cycles for a touch to go out of detect.

The range for this field is 0 to 255, where 0 is the same as 1.

Range: 0 (1), 1 to 255

TCHHYST Field

This field sets the Touch Hysteresis (in 8-bit deltas). This is the level of hysteresis applied to touch detections for all keys in the key array. For a touch to enter detection the touch delta must be greater than the touch threshold (TCHTHR). For a touch to leave detection the touch delta must be less than (TCHTHR – TCHHYST). This field allows the Key Array T15 to be tuned so that a hovering finger does not cause detection to flicker on and off.

Note that if the Noise Suppression T72 object supplies a minimum touch threshold that is greater than TCHTHR – TCHHYST, then a touch's delta must be greater than the minimum touch threshold plus TCHHYST to enter detection, and less than the minimum touch threshold to leave detection.

TCHHYST is capped internally to (TCHTHR / 4). A hysteresis greater than 25% of the Touch Threshold is therefore not possible.

Note that a value of 0 means 2 and this is uncapped.

Range: 0 (2), 1 to 63

CFG Field

This field allows different debug data types to be enabled or disabled for use with the hardware debug interface

DBGDELTAEN: If this bit is set to 1, the key deltas are output by the hardware debug interface. Note: that the Command Processor T6 DEBUGCTRL2 DBGOBJMODEEN bit must be set to 1 for this to work (see [“DEBUGCTRL2 Field”](#)).

DBGREFSEN: If this bit is set to 1, the key references are output by the hardware debug interface. Note: that the Command Processor T6 DEBUGCTRL2 DBGOBJMODEEN bit must be set to 1 for this to work (see [“DEBUGCTRL2 Field”](#)).

4.2.3 CONFIGURATION CHECKS

A Key Array T15 object causes a configuration check to be performed in the following circumstances:

- When the object is enabled (that is, the ENABLE bit is set in the CTRL field)
- When certain fields are changed (as listed in [Table 4-4](#))

In addition, some fields will cause an automatic recalibration to be performed (see [Table 4-4](#)).

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)), and the device halts until the error has been corrected.

To fix the error, the object settings should be checked to verify that they are all within their allowed limits, as described in the field descriptions. In particular, the following should be checked:

- There are no more than 8 keys defined, that is: $XSIZE \times YSIZE \leq 8$
- If self capacitance measurements are enabled, the key array does not share X and Y lines with other touch objects (specifically Multiple Touch Touchscreen T100).
- TCHTHR is greater than or equal to 2

TABLE 4-4: CONFIGURATION CHECKS FOR KEY ARRAY T15 (TOUCH_KEYARRAY_T15)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes ^{(1) (2)}	None
XORIGIN	Yes	Yes	Error if out of range
YORIGIN	Yes	Yes	Error if out of range
XSIZE	Yes	Yes	Error if out of range
YSIZE	Yes	Yes	Error if out of range
AKSCFG	No	No	None
BLN	No	Yes	None
TCHTHR	Yes	No	None
TCHDI	No	No	None
TCHHYST	Yes	No	None
CFG	No	No	None

Note 1: If the ENABLE bit is toggled on or off.

Note 2: If the ENABLE bit is toggled on or off. Note that if the ENABLE bit is cleared and *all* of the touch objects are now disabled, no calibration takes place.

4.2.4 MESSAGES

A Key Array T15 object reports on/off touch information in its message data. The message data for a Key Array T15 object is shown in [Table 4-5](#).

Note that Key Array T15 messages will be suppressed if Touchscreen Contact Suppression is enabled, and the conditions are met (see TSCONTSUP bit in the [“CTRL Field”](#)).

**TABLE 4-5: MESSAGE DATA FOR KEY ARRAY T15
(TOUCH_KEYARRAY_T15)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	DETECT	Reserved						
2	KEYSTATE	KEY7	KEY6	KEY5	KEY4	KEY3	KEY2	KEY1	KEY0

STATUS Field

Reports the current status of the object.

DETECT: Set if any key is in a touched state.

KEYSTATE Field

Reports the state of each key, one bit per key; 0 = key is untouched, 1 = key is touched.

4.2.5 DIAGNOSTIC DEBUG DATA

The diagnostic debug data modes for the Key Array T15 object is shown in [Table 4-6](#). The diagnostic debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in [Table 4-6](#).

The diagnostic debug modes are described in the following sections.

TABLE 4-6: DIAGNOSTIC DEBUG COMMANDS

T6 Command	Name	Description
0x17	Key Array Delta mode	The Diagnostic Debug T37 object holds the 16-bit deltas for each of the enabled keys on the device. See Section 4.2.5.1 "Key Deltas Mode" .
0x18	Key Array References mode	The Diagnostic Debug T37 object holds the 16-bit references for each of the enabled keys on the device. See Section 4.2.5.2 "Key References Mode" .

4.2.5.1 Key Deltas Mode

[Figure 4-2](#) shows the organization of the data in the pages for the key delta mode. The bytes give the 16-bit deltas for the Key Array keys (8 in total). The keys are numbered in node order, as described in [Section 4.2.1 "Introduction"](#).

FIGURE 4-2: DIAGNOSTIC DEBUG T37 DATA – KEY DELTA MODE

Page	Data Index	Data[]
	0 – 15	Key deltas for Key Array
	16 – 127	Reserved

Assumes page size = 128

4.2.5.2 Key References Mode

[Figure 4-3](#) shows the organization of the data in the pages for the key references mode. The bytes give the 16-bit references for the Key Array keys (8 in total). The keys are numbered in node order, as described in [Section 4.2.1 "Introduction"](#).

FIGURE 4-3: DIAGNOSTIC DEBUG T37 DATA – KEY DELTA MODE

Page	Data Index	Data[]
	0 – 15	Key references for Key Array
	16 – 127	Reserved

Assumes page size = 128

4.3 Multiple Touch Touchscreen T100 Object

4.3.1 INTRODUCTION

A Multiple Touch Touchscreen T100 object is used to configure a Multiple Touch Touchscreen on the sensor matrix. It reports on all measurements made, including those detected by other objects (see [Section 4.3.4 “Messages”](#)). Specifically, it reports the following screen and touch status information:

- Finger touches detected by the Multiple Touch Touchscreen T100 object
- Grip suppression by the Grip Suppression T40 object
- Screen suppression by the Touch Suppression T42 object
- Glove touches detected by the Glove Detection T78 object

NOTE

- To be able to track and report touches in self capacitance measurements, the Multiple Touch Touchscreen T100 object needs to have data for both axes (see [“CTRL Field”](#) and [Appendix A “Measurement Processing on mXT144U”](#)).

4.3.1.1 Touch Threshold

Multiple Touch Touchscreen T100 object has a Touch Threshold (TCHTHR) that defines how large a node's touch delta (Reference minus the signal) must be to qualify as a potential touch detection (in 8-bit deltas). The reference level is determined during calibration and adjusted using drift compensation. The final detection confirmation uses the Touch Detect Integration as described in [“TCHDIDOWN Field”](#). Larger values for the threshold desensitize nodes, since the signal must change more in order to exceed the threshold level. Conversely, lower thresholds make nodes more sensitive.

The setting for TCHTHR for each node depends on the amount of signal swing that occurs when a node is touched. Thicker panels or smaller electrode geometries reduce node gain, that is signal swing from touch. In this case smaller TCHTHR values are required to detect touch.

A Touch Hysteresis is provided by the TCHHYST field. For a touch to enter detection the touch delta must be greater or equal to the touch threshold (TCHTHR). For a touch to leave detection the touch delta must be less than (TCHTHR – TCHHYST).

In addition to the Touch Threshold there is also an Internal Touch Threshold that operates in a similar manner to control internally tracked touch detections. By default, it is set to (TCHTHR – TCHHYST)/2. The Internal Touch Threshold also has a hysteresis (INTTHRHYST). For a touch, therefore, to enter detection the touch delta must be greater or equal to INTTHR and for a touch to leave detection the touch delta must be less than (INTTHR – INTTHRHYST).

Other objects can adjust the minimum touch threshold in specific circumstances. These objects are:

- Noise Suppression T72 – Supplies a minimum touch threshold that desensitizes the Multiple Touch Touchscreen T100 to avoid reporting any detected noise that would be reported as touches (see [Section 5.8 “Noise Suppression T72 Object”](#)). This affects both the Touch Threshold and the Internal Touch Threshold.

Note that if the Noise Suppression T72 object supplies a minimum touch threshold that is greater than TCHTHR – TCHHYST, then a touch delta must be greater than the minimum touch threshold plus TCHHYST to enter detection, and less than the minimum touch threshold to leave detection.

Similarly, if the Noise Suppression T72 object supplies a minimum touch threshold that is greater than INTTHR – INTTHRHYST, then a touch's delta must be greater than the minimum touch threshold plus INTTHRHYST to enter detection, and less than the minimum touch threshold to leave detection.

- Glove Detection T78 – Can adjust the threshold to allow for glove touches (see [Section 5.9 “Glove Detection T78 Object”](#)).

4.3.1.2 Edge Correction

The Multiple Touch Touchscreen T100 object allows edge correction to be applied for improved linearity at the edge of the screen on certain touchscreen designs. The following edge correction functionality is provided (see [Table 4-7](#)):

- **Uniform edge correction** – The edge correction settings are the same for both the high and low sense lines on the X or Y axis.
- **Separate edge correction** – The edge correction settings are different for both the high and low sense lines on the X or Y axis. This allows separate control of the edge correction that is applied to the high and low sense line edges. Sensors with different non-linear responses at the high and low sense lines of the X and Y axes can therefore be corrected for.

- **Dual X** – In addition to the uniform and separate edge correction modes, the X axis can have additional settings that apply when Dual X measurements are being made.

Note that the types of edge correction can be set independently on the X and Y axes. For example, it is possible to have uniform edge correction on the Y axis and separate edge correction on the X axis.

See the following sections for more information on the fields that control edge correction:

“XEDGECFG, DXEDGECFG and YEDGECFG Fields”

“XEDGEDIST, DXEDGEDIST and YEDGEDIST Fields”

“XEDGECFGHI, DXEDGECFGHI and YEDGECFGHI Fields”

“XEDGEDISTHI, DXEDGEDISTHI and YEDGEDISTHI Fields”

TABLE 4-7: EDGE CORRECTION FIELDS

Setting	Fields	Effect	
Uniform X Edge Correction XEDGECFGHI ENABLE = 0	XEDGECFG XEDGEDIST	Set the edge correction for both high and low X lines.	Low and high sense lines have same edge correction
Uniform Dual X Edge Correction DXEDGECFGHI ENABLE = 0	DXEDGECFG DXEDGEDIST	Set the edge correction for both high and low X lines for Dual X measurements.	
Uniform Y Edge Correction YEDGECFGHI ENABLE = 0	YEDGECFG YEDGEDIST	Set the edge correction for both high and low Y lines.	
Separate X Edge Correction XEDGECFGHI ENABLE = 1	XEDGECFGHI XEDGEDISTHI	Set the edge correction for high X lines.	Low and high sense lines have different edge correction
	XEDGECFG XEDGEDIST	Set the edge correction for low X lines.	
Separate Dual X Edge Correction DXEDGECFGHI ENABLE = 1	DXEDGECFGHI DXEDGEDISTHI	Set the edge correction for high X lines for Dual X measurements.	
	DXEDGECFG DXEDGEDIST	Set the edge correction for low X lines for Dual X measurements.	
Separate Y Edge Correction YEDGECFGHI ENABLE = 1	YEDGECFGHI YEDGEDISTHI	Set the edge correction for high Y lines.	
	YEDGECFG YEDGEDIST	Set the edge correction for low Y lines.	

Note: Other objects may supply edge correction for specific types of touches, in which case they will take precedence over Multiple Touch Touchscreen T100 edge correction settings.

4.3.2 CONFIGURATION

**TABLE 4-8: CONFIGURATION FOR MULTIPLE TOUCH TOUCHSCREEN T100
(TOUCH_MULTITOUCHSCREEN_T100)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	SCANEN	Reserved				DISSCRMMSG0	RPTEN	ENABLE
1	CFG1	INVERTX	INVERTY	SWITCHXY	DISLOCK	Reserved			RPTEACHCYCLE
2	SCRAUX	Reserved				INTTHRAREA	ATCHAREA	TCHAREA	NUMRPTTCH
3	TCHAUX	Reserved		AREAHW	PEAK	HW	AREA	AMPL	VECT
4	TCHEVENTCFG	Reserved		DISMOVE	DISSUP	DISUNSUP	DISUP	DISDOWN	Reserved
5	AKSCFG	GROUP8	GROUP7	GROUP6	GROUP5	GROUP4	GROUP3	GROUP2	GROUP1
6	NUMTCH	Reserved			Number of reported touches				
7	XYCFG	YSPAN	Reserved			XSPAN	Reserved		
8	XORIGIN	X line start position of object							
9	XSIZE	Number of X lines the object occupies							
10	XPITCH	X line pitch							
11	XLOCLIP	X low clipping boundary width							
12	XHICLIP	X high clipping boundary width							
13	XRANGE	X resolution LSByte							
14		X resolution MSByte							
15	XEDGECFG	Reserved		CORRECTIONGRADIENT					
16	XEDGEDIST	X edge correction distance							
17	DXXEDGECFG	Reserved		CORRECTIONGRADIENT					
18	DXXEDGEDIST	Dual X X edge correction distance							
19	YORIGIN	Y line start position of object							
20	YSIZE	Number of Y lines the object occupies							
21	YPITCH	Y line pitch							
22	YLOCLIP	Y low clipping boundary width							
23	YHICLIP	Y high clipping boundary width							
24	YRANGE	Y resolution LSByte							
25		Y resolution MSByte							
26	YEDGECFG	Reserved		CORRECTIONGRADIENT					
27	YEDGEDIST	Y edge correction distance							
28	GAIN	Gain							
29	DXGAIN	Dual X Gain							
30	TCHTHR	Touch threshold							
31	TCHHYST	Touch threshold hysteresis							
32	INTTHR	Internal threshold							
33	NOISESF	Noise Scaling Factor							
34	CUTOFFTHR	Cut-off threshold							
35	MRGTHR	Merge threshold							
36	MRGTHRADJSTR	Merge Threshold Adjustment Strength							
37	MRGHYST	Merge hysteresis							
38	DXTHRSF	Dual X Threshold scaling factor							
39	TCHDIDOWN	Reserved		TCHDI for touch-down					
40	TCHDIUP	Reserved		TCHDI for touch release					
41	NEXTTCHDI	Reserved		NEXTTCHDI					
42	Reserved	Reserved							
43	JUMPLIMIT	Jump limit							
44	MOVFILTER	DISABLE	MEDOFF	Reserved			SPEEDRESP		
45	MOVSMOOTH	Movement smoothing							

**TABLE 4-8: CONFIGURATION FOR MULTIPLE TOUCH TOUCHSCREEN T100
(TOUCH_MULTITOUCHSCREEN_T100)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
46	MOVPRED	Movement prediction							
47	MOVHYSTI	Movement hysteresis, initial LSByte							
48		Reserved				Movement hysteresis, initial msbits			
49	MOVHYSTN	Movement hysteresis, next LSByte							
50		Reserved				Movement hysteresis, next msbits			
51	AMPLHYST	Touch amplitude reporting hysteresis							
52	SCRAREAHYST	Screen area reporting hysteresis							
53	INTTHRHYST	Internal tracking threshold hysteresis							
54	XEDGECFGHI	ENABLE	Reserved	CORRECTIONGRADIENT					
55	XEDGEDISTHI	X high edge correction distance							
56	DXXEDGECFGHI	ENABLE	Reserved	CORRECTIONGRADIENT					
57	DXXEDGEDISTHI	Dual X X high edge correction distance							
58	YEDGECFGHI	ENABLE	Reserved	CORRECTIONGRADIENT					
59	YEDGEDISTHI	Y high edge correction distance							
60	CFG2	Reserved					CONFTHR		
61	MOVHYSTCFG	Reserved	MINSPEED				SPEEDRSP		
62	AMPLCOEFF	Coefficient for scaled touch amplitude							
63	AMPLOFFSET	Offset for scaled touch amplitude							
64	JUMPLIMITMOV	JumpLimit for Moving Touches							
65	JLMMOVTHR	JLMMOVTHR LSByte							
66		Reserved				JLMMOVTHR MSbits			
67	JLMMOVINTTHR	Jump Limit Movement Integration Threshold							

CTRL Field

ENABLE: Enables the use of this Multiple Touch Touchscreen T100 object. The object is enabled if set to 1, and disabled if set to 0. The object does not scan for touches if it is disabled, in order to conserve power.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0. Events must be enabled for this bit to have an effect.

DISSCRMMSG0: Disables the generation of screen messages in the object's first report ID.

SCANEN: Enables close scanning for touch detections. The device normally has a very efficient mechanism for determining touch separation (that is, distinguishing between multiple fingers). This is sufficient for most purposes. Under some (very unusual) circumstances, however, it is possible for it to fail to distinguish between multiple touches. Setting this bit to 1 causes the device to perform a close scan for touch detections once every 200 ms. Multiple touches are correctly detected, but the checks still occur infrequently enough to prevent excessive power consumption. This scan is done only when the touchscreen is touched.

CFG1 Field

RPTEACHCYCLE: Setting this bit to 1 causes the Multiple Touch Touchscreen T100 object to generate a touch status message for each touch that is reporting, irrespective of whether any events have happened. The message will have the same touch co-ordinates as the preceding touch status message. Setting this bit to 0 gives the default message generation behavior described in [Section 4.3.4.3 "Subsequent Report IDs – Touch Status Messages"](#).

DISLOCK: Disables the coordinate position locking that is applied to touches inside a clipping boundary (see ["X RANGE/Y RANGE, X LOCLIP/Y LOCLIP and XHICLIP/YHICLIP Fields"](#) for more information). Set this field to 1 to disable locking, and to 0 to enable locking.

SWITCHXY: Switches the X and Y positions; that is, the screen is flipped about the diagonal from (X0, Y0) to (Xmax, Ymax).

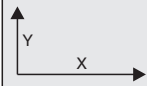
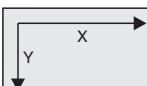




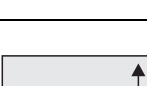
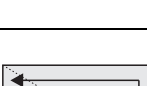
INVERTY: Inverts Y coordinates; that is: $Y_{newval} = (Y_{max} - Y)$.

INVERTX: Inverts X coordinates; that is: $X_{newval} = (X_{max} - X)$.

Note that an INVERTX and/or INVERTY operation takes place before a SWITCHXY.

The effect of these three bits is shown in Table 4-9.

TABLE 4-9: CFG1 TOUCHSCREEN ORIENTATION SETTINGS

Bits			Touchscreen Coordinates	Touchscreen Orientation
7 (INVERTX)	6 (INVERTY)	5 (SWITCHXY)		
0	0	0		Normal orientation
0	1	0		Vertical flip
1	0	0		Horizontal flip
1	1	0		Rotated 180°
0	0	1		Diagonal mirror along axis from (X0, Y0) to (Xmax, Ymax)
0	1	1		Rotated 90° clockwise
1	0	1		Rotated 90° counterclockwise
1	1	1		Diagonal mirror along axis from (X0, Ymax) to (Xmax, Y0)

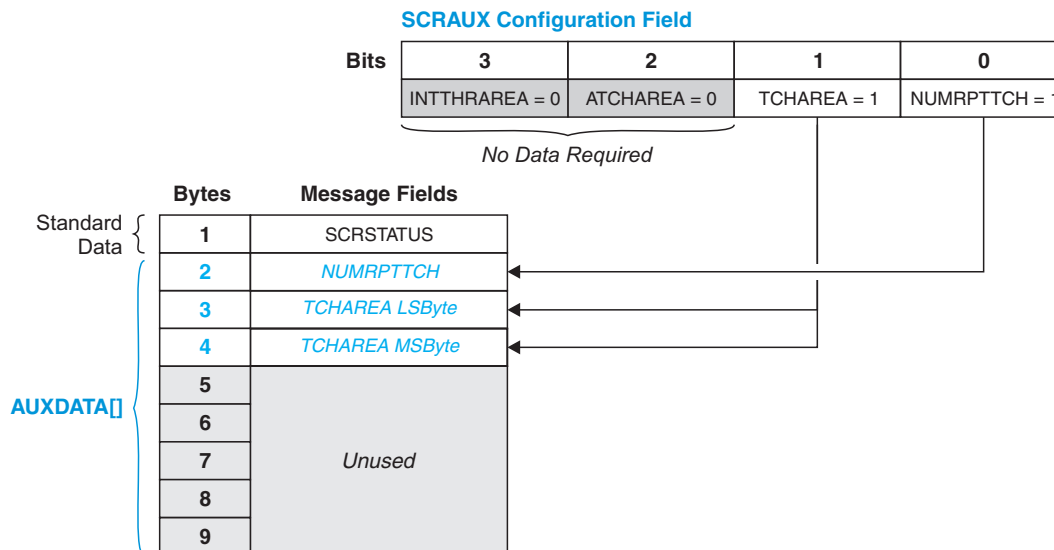
Note: Xmax and Ymax refer to the maximum X and Y screen positions; that is, they relate to the touchscreen resolution and not to the X and Y lines.

SCRAUX Field

This field configures the report output in the AUXDATA[] fields for the screen status messages (see Section 4.3.4.1 “First Report ID – Screen Status Messages”). See also “SCRAREAHYST Field”.

The AUXDATA[] message bytes are filled in the same order as the SCRAUX bits, depending on which bits are enabled. For example, if only the NUMRPTTCH and TCHAREA bits are set in the SCRAUX configuration field, then only AUXDATA[0] to AUXDATA[2] are used (that is, byte for NUMRPTTCH followed by two bytes for TCHAREA). This is summarized in [Figure 4-4](#).

FIGURE 4-4: EXAMPLE SCREEN STATUS MESSAGE



NUMRPTTCH: If this bit is set to 1, the number of reportable touches that are being tracked on the touchscreen will be reported in the AUXDATA[] fields. The AUXDATA[] Fields for NUMRPTTCH are shown in [Table 4-10](#).

TABLE 4-10: AUXDATA[] FIELDS FOR NUMRPTTCH

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	NUMRPTTCH	Number of Reportable Touches							

The NUMRPTTCH message field indicates the number of reportable touches currently on the screen. A screen status message is generated when this value changes.

TCHAREA: If this bit is set to 1, the number of nodes \geq (TCHTHR – TCHHYST) on the touchscreen will be reported in the AUXDATA[] fields. The AUXDATA[] Fields for TCHAREA are shown in [Table 4-11](#).

TABLE 4-11: AUXDATA[] FIELDS FOR TCHAREA

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	TCHAREA	Touch area LSByte							
n + 1		Touch area MSByte							

The TCHAREA message field indicates the total number of nodes \geq (TCHTHR – TCHHYST) on the screen. A screen status message is generated when this value changes by more than SCRAREAHYST.

Note that the data reported in TCHAREA is based on data after processing by the Retransmission Compensation T80 object if the Retransmission Compensation T80 object is enabled.

ATCHAREA: If this bit is set to 1, the number of nodes below $-(TCHTHR - TCHHYST)$ on the touchscreen will be reported in the AUXDATA[] fields. The AUXDATA[] Fields for ATCHAREA are shown in [Table 4-12](#).

TABLE 4-12: AUXDATA[] FIELDS FOR ATCHAREA

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	ATCHAREA	Anti-touch area LSByte							
n + 1		Anti-touch Area MSByte							

The ATCHAREA message field indicates the total number of nodes below $-(TCHTHR - TCHHYST)$ on the screen. A screen status message is generated when this value changes by more than SCRAREAHYST.

INTTHRAREA: If this bit is set to 1, the number of nodes above (INTTHR) on the touchscreen will be reported in the AUXDATA[] fields. The AUXDATA[] Fields for INTTHRAREA are shown in [Table 4-13](#).

TABLE 4-13: AUXDATA[] FIELDS FOR INTTHRAREA

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	INTTHRAREA	Internal tracking threshold area LSByte							
n + 1		Internal tracking threshold area MSByte							

The INTTHRAREA message field indicates the total number of nodes above INTTHR on the screen. A screen status message is generated when this value changes by more than SCRAREAHYST.

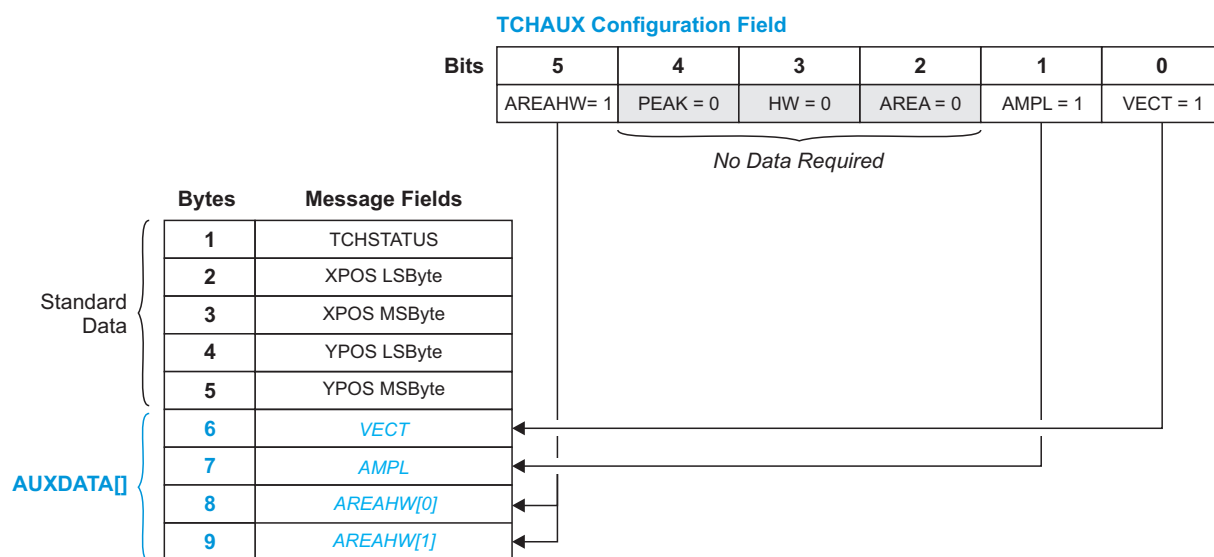
Note that the data reported in INTTCHAREA is based on data after processing by the Retransmission Compensation T80 object if the process if the Retransmission Compensation T80 object is enabled.

TCH AUX Field

This field configures the report output in the AUXDATA[] fields for the touch status messages (see [Section 4.3.4.3 “Subsequent Report IDs – Touch Status Messages”](#)).

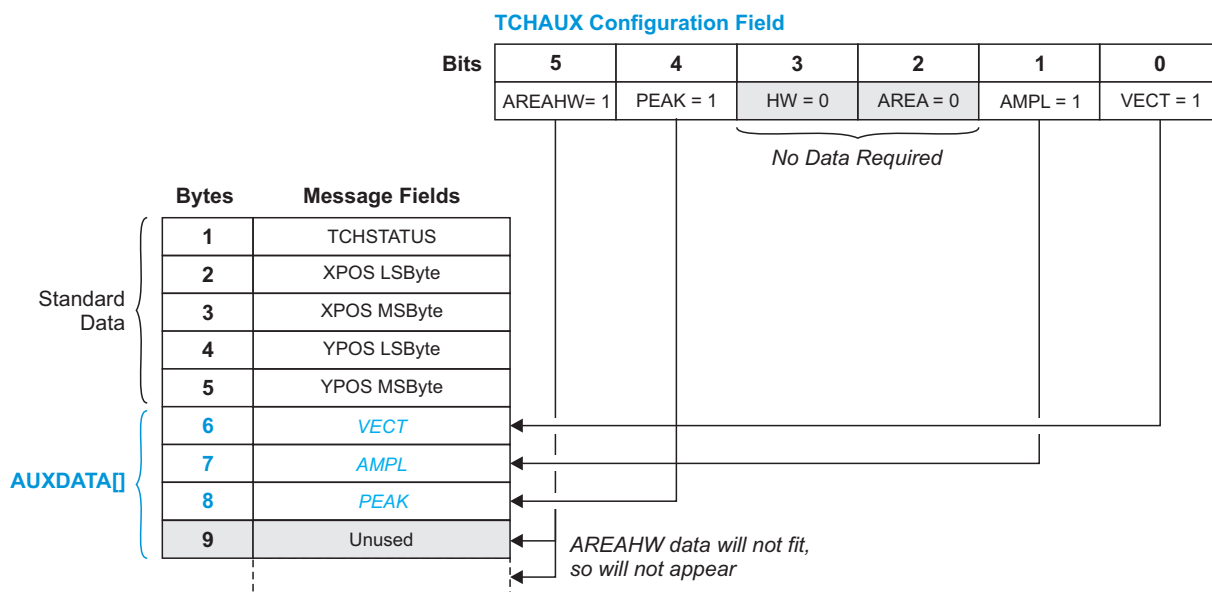
The AUXDATA[] bytes are filled in the same order as the TCH AUX bits (from bit 0 to bit 7), depending on which bits are enabled. For example if the VECT, AREA and AREAHW bits are set to 1 in the TCH AUX field, then the AUXDATA[] bytes will contain one byte for the VECT data, followed by one byte for AREA data and finally two bytes for the AREAHW data. This is summarized in [Figure 4-5](#). Note that the intermediate bits (AREA, HW and PEAK) are set to 0 in this example. The AREA, HW and PEAK data is therefore skipped in the AUXDATA[] bytes.

FIGURE 4-5: EXAMPLE TOUCH STATUS MESSAGE



The AUXDATA[] bytes can hold a maximum of four bytes of data, so if more than four bytes of data is selected in TCH AUX, the data for the most significant TCH AUX bits will not appear. For example, if PEAK is also selected, in addition to the bits selected in [Figure 4-5](#), the resulting data is as shown in [Figure 4-6](#). In this case, there is not enough room in the AUXDATA[] bytes for the AREAHW data so it does not appear.

FIGURE 4-6: EXAMPLE TOUCH STATUS MESSAGE – DATA DOES NOT FIT



VECT: If this bit is set to 1, the vector calculation will be performed on each touch and reported in the AUXDATA[] fields. If this bit set to 0, the calculation will not be performed allowing processing time to be minimized. The AUXDATA[] Fields for VECT are shown in Table 4-14.

TABLE 4-14: AUXDATA[] FIELDS FOR VECT

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	VECT	VCOMP1				VCOMP0			

The VECT message field gives an indication of the orientation of the touch and a confidence level as a vector made up of two signed (two's complement) 4-bit components:

- VCOMP1: ratio of (+X,+Y) to (−X,−Y), representing a diagonal (value: −7 to +7)
- VCOMP0: ratio of X to Y (value: −7 to +7)

The magnitude of the vector represents the confidence. The vector uses the following calculations:

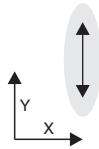
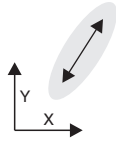
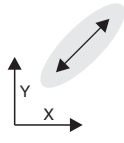
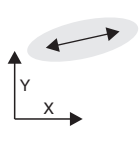
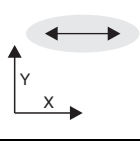
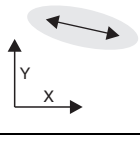
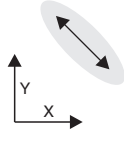
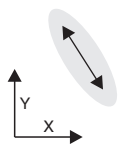
$$Angle = \tan^{-1} \left(\frac{vcomp_1}{vcomp_0} \right) \S 2$$

$$Magnitude = \sqrt{vcomp_1^2 + vcomp_0^2}$$

NOTE The touch vector is affected by the orientation of the screen (see “CFG1 Field”). In the example touch directions shown in Table 4-15 the normal orientation is assumed.

Changes to this field do not generate a touch status message; the vector is reported only when another message-triggering event occurs.

TABLE 4-15: TOUCH DIRECTION (NORMAL ORIENTATION ASSUMED)

TCHVECTOR Field		Touch Direction
VCOMP1	VCOMP0	
0	Positive	
Negative	Positive	
Negative	0	
Negative	Negative	
0	Negative	
Positive	Negative	
Positive	0	
Positive	Positive	

AMPL: If this bit is set to 1, the amplitude calculation will be performed on each touch and reported in the AUXDATA[] fields. If this bit set to 0, the calculation will not be performed allowing processing time to be minimized. See also “[AMPLHYST Field](#)”. The AUXDATA[] Fields for AMPL are shown in [Table 4-16](#).

TABLE 4-16: AUXDATA[] FIELDS FOR AMPL

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	AMPL	Touch amplitude							

The AMPL message field reports the touch amplitude (saturated to 8 bits). This is a value that is proportional to the signal delta of the nodes within the touch. This can be used, for example, to extrapolate the size or pressure of the user's finger touch. Touch status messages are generated when the delta change in the value is greater than the Amplitude Hysteresis (see "[AMPLHYST Field](#)"). The value of the reported touch amplitude can be scaled by the AMPLCOEFF and AMPLOFFSET fields, which are applied before AMPLHYST (see "[AMPLCOEFF Field](#)" and "[AMPLOFFSET Field](#)").

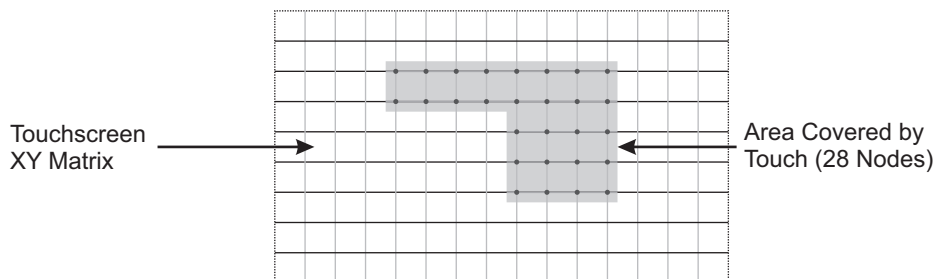
AREA: If this bit is set to 1, the area calculation will be performed on each touch and reported in the AUXDATA[] fields. If this bit set to 0, the calculation will not be performed allowing processing time to be minimized. The AUXDATA[] Fields for AREA are shown in [Table 4-17](#).

TABLE 4-17: AUXDATA[] FIELDS FOR AREA

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	AREA	Area							

The AREA message field reports the size of the touch area in terms of the number of nodes that are covered by the touch and are above (TCHTHR–TCHYST). For example, the area covered by the touch in [Figure 4-7](#) is 28 nodes.

FIGURE 4-7: TOUCH AREA



Changes to this field do not generate a touch status message; the touch area is reported only when another message-triggering event occurs.

HW: If this bit is set to 1, the height/width calculation will be performed on each touch and reported using the AUXDATA[] fields. Note that the calculation of the height and width uses the XPITCH and YPITCH values and so will be accurate only if these are set correctly for the sensor. If this bit set to 0, the calculation may not be performed, allowing processing time to be minimized.

The AUXDATA[] Fields for HW are shown in [Table 4-18](#).

TABLE 4-18: AUXDATA[] FIELDS FOR HW

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	HEIGHT	Touch height							
n + 1	WIDTH	Touch width							

The HEIGHT field reports the height of the touch in mm.

The WIDTH field reports the width of the touch in mm.

PEAK: If this bit is set to 1, the peak calculation will be performed on each touch and reported in the AUXDATA[] fields. If this bit set to 0, the calculation will not be performed allowing processing time to be minimized. The AUXDATA[] Fields for PEAK are shown in [Table 4-19](#).

TABLE 4-19: AUXDATA[] FIELDS FOR PEAK

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	PEAK	Peak delta							

The PEAK message field reports the peak delta of the touch in 8-bit delta units (saturated to 8 bits). Changes to this field do not generate a touch status message; the peak delta is reported only when another message-triggering event occurs.

AREAHW: If this bit is set to 1, the area and height/width calculations will be performed on each touch and the data reported in the AUXDATA[] fields. If this bit set to 0, the calculations may not be performed (allowing processing time to be minimized) and the data will not be reported.

The AREAHW bit can be used to allow more pieces of auxiliary data to be reported at once (for example, area, height and width can be reported at the same time as VECT and AMPL). There is no point in setting it at the same time as AREA or HW as the same information will be reported in two different ways.

The AUXDATA[] Fields for AREAHW are shown in [Table 4-20](#).

TABLE 4-20: AUXDATA[] FIELDS FOR AREAHW

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n	AREAHW[0]	Reserved	EXP		AREA				
n + 1	AREAHW[1]	WIDTH				HEIGHT			

The AREAHW[0] EXP is an exponent (shift) which applies to AREAHW[0] AREA, AREAHW[1] HEIGHT and AREAHW[1] WIDTH. For example, EXP can be used to extract the touch area from AREAHW[0], as follows:

Touch Area = AREAHW[0] AREA << AREAHW[0] EXP

After the exponent has been applied, the units of area are nodes and the units of height and width are mm.

If the exponent reaches its maximum value then the reported values in AREAHW[0] AREA, AREAHW[1] HEIGHT and AREAHW[1] WIDTH will be saturated to their respective bit widths.

See AREA on [page 55](#) and HW on [page 55](#) for further information about the reported data.

Changes to the AREAHW bit do not generate a touch status message; the AREAHW data is reported only when another message-triggering event occurs.

TCHEVENTCFG Field

This field provides secondary configuration options to configure the touch events that are able to cause touch status messages to be generated (see [Section 4.3.4.3 "Subsequent Report IDs – Touch Status Messages"](#)).

DISDOWN: If this bit is set to 1, the touch DOWN event will not generate a touch message.

DISUP: If this bit is set to 1, the touch UP event will not generate a touch message.

DISUNSUP: If this bit is set to 1, the touch UNSUP event will not generate a touch message.

DISSUP: If this bit is set to 1, the touch SUP event will not generate a touch message.

DISMOVE: If this bit is set to 1, the touch MOVE event will not generate a touch message.

AKSCFG Field

This field configures Adjacent Key Suppression (AKS) between this Multiple Touch Touchscreen T100 object and any other touch present on the device.

AKS technology is a patented method used to detect which touch object is touched when objects are located close together. A touch in a group of AKS objects is indicated only on the object with the largest signal. This is assumed to be the intended object. Once an object in an AKS group is in detect, there can be no further detections within that group until the object is released.

GROUP1 to GROUP8: These bits form a bit field that specifies which AKS groups this Multiple Touch Touchscreen T100 object is within. The default value of 0 means that the object is in no AKS groups and the AKS feature is disabled for the Multiple Touch Touchscreen T100.

NUMTCH Field

The Number of Touches (NUMTCH) setting indicates the number of touches to be reported. Report IDs are not removed (see [Section 1.5.6 "Report IDs"](#)). The upper limit for this field is the maximum number of touches (5 on the mXT144U).

Touch IDs are allocated sequentially as touches are detected and any Touch IDs below NUMTCH are reported. This means that Touch IDs \geq NUMTCH are not reported, even if the number of touches currently on the touchscreen does not exceed NUMTCH.

Range: 0 to 5 (maximum number of touches)

XYCFG Field

This field configures settings related to the physical layout of a touchscreen sensor in both axes.

XSPAN and YSPAN: These bits enable the use of an “extra electrode” in the creation of the touch coordinates. XSPAN/YSPAN should be set to 1 if the first and last pixels of the LCD are outside the center point of the first and last sense electrodes.

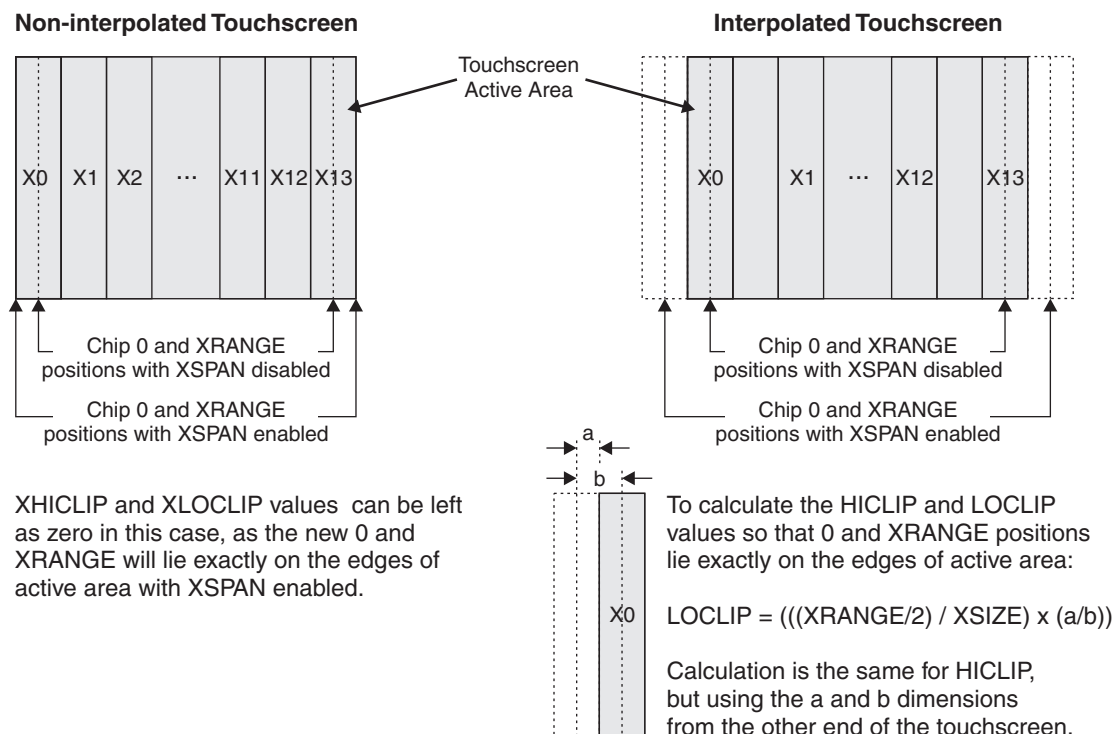
At the edges of a touchscreen sensor pattern there should ideally be a half-width electrode. Depending on how the screen has been designed, there may actually be an electrode that is somewhere between 50 percent and 100 percent width at the edge. This can cause discrepancies between the physical touch position and the reported touch position. XSPAN/YSPAN helps correct these discrepancies.

The XSPAN/YSPAN field is used in combination with XLOCLIP/YLOCLIP and XHICLIP/YHICLIP settings. This allows the device to correct for accuracy or linearity problems resulting from the practicalities of the ITO design.

XSPAN/YSPAN moves the theoretical 0 and X RANGE/Y RANGE position outwards from the screen center by one half electrode width each. The XLOCLIP/YLOCLIP and XHICLIP/YHICLIP controls can then be used to move the theoretical 0 and X RANGE/Y RANGE position back in towards the center in finer steps.

Figure 4-8 shows an example of how to correct for two screen designs with an X size of 15. Note that although Figure 4-8 shows the XSPAN control for the X lines, the same principles apply when the YSPAN bit is used with the Y lines.

FIGURE 4-8: CORRECTING FOR TOUCHSCREEN DESIGNS (X COORDINATES)



XORIGIN, YORIGIN, XSIZE and YSIZE Fields

These fields specify the size and position of the touchscreen on the actual matrix, in terms of X and Y lines (see Figure 4-9). The XORIGIN and YORIGIN fields specify the origin and the XSIZE and YSIZE fields specify the size.

Note that in the definitions that follow, maxX and maxY are the indices of the highest available X and Y lines, depending on the particular device.

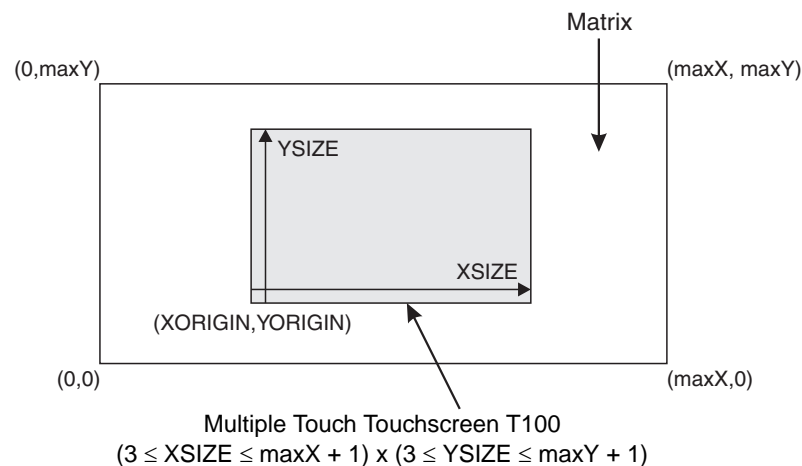
- The range for XORIGIN and YORIGIN is zero to ($\text{maxX} - \text{XSIZE} + 1$) or ($\text{maxY} - \text{YSIZE} + 1$). If self capacitance measurements are enabled, however (see “MEASALLOW Field”), the touchscreen origin must be at X0,Y0. For this reason it is recommended that the touchscreen origin is always set to X0, Y0. Other touch objects (if any) should be placed on high X and Y lines.

The XSIZE and YSIZE fields should be set as follows:

- The minimum value for XSIZE is 3. If Dual X Drive is enabled for use in the Noise Suppression T72 object, the minimum XSIZE is increased to 4.
- The minimum value for YSIZE is 3.
- The maximum size for a Multiple Touch Touchscreen T100 depends on the number of X and Y lines on a particular device.

NOTE A Multiple Touch Touchscreen T100 object cannot share an X or Y line with another touch object (for example, a Key Array T15).
Refer to the *mXT144U 1.0 Data Sheet* for more information on the layout of the touchscreen sensor.

FIGURE 4-9: ORIGIN AND SIZE FIELDS



XORIGIN Range: 0 always

YORIGIN Range: 0 always

XSIZE Range: 3 to $\text{maxX} + 1$ or
4 to $\text{maxX} + 1$ if Dual X Drive enabled
(4, 6, 8, 10 or 12 if self capacitance measurements enabled)

YSIZE Range: 3 to $\text{maxY} + 1$ (4, 6, 8, 10 or 12 if self capacitance measurements enabled)

XPITCH and YPITCH Fields

These fields specify the physical pitch of the X and Y lines on the touchscreen sensor; that is, the spacing between the centers of the X or Y lines. These settings are specified in units of 0.1 mm, where a setting of zero means 5 mm. These fields allow the touch suppression algorithms to be tuned to non-standard pitch sizes.

Range: 0 (5 mm), 1 to 255 (in 0.1 mm units)

XRANGE/YRANGE, XLOCLIP/YLOCLIP and XHICLIP/YHICLIP Fields

These six fields control the resolution, and thus the reported position, of the touchscreen.

NOTE The XRANGE/YRANGE, XLOCLIP/YLOCLIP and XHICLIP/YHICLIP fields operate on the physical ITO matrix of the touchscreen, and are not affected by the logical orientation set by the orientation controls in the CFG1 field.

The XRANGE and YRANGE fields set the output resolution for the reported position.

The minimum resolution is:

$$(R \times N) / 256$$

The maximum resolution is:

$$(R \times N) - 2 \quad (\text{capped to } 65534)$$

where:

$$R = 1024 - XLOCLIP - XHICLIP$$

$$N = XSIZE - 1 + XSPAN_ENABLED$$

or:

$$R = 1024 - YLOCLIP - YHICLIP$$

$$N = YSIZE - 1 + YSPAN_ENABLED$$

as appropriate.

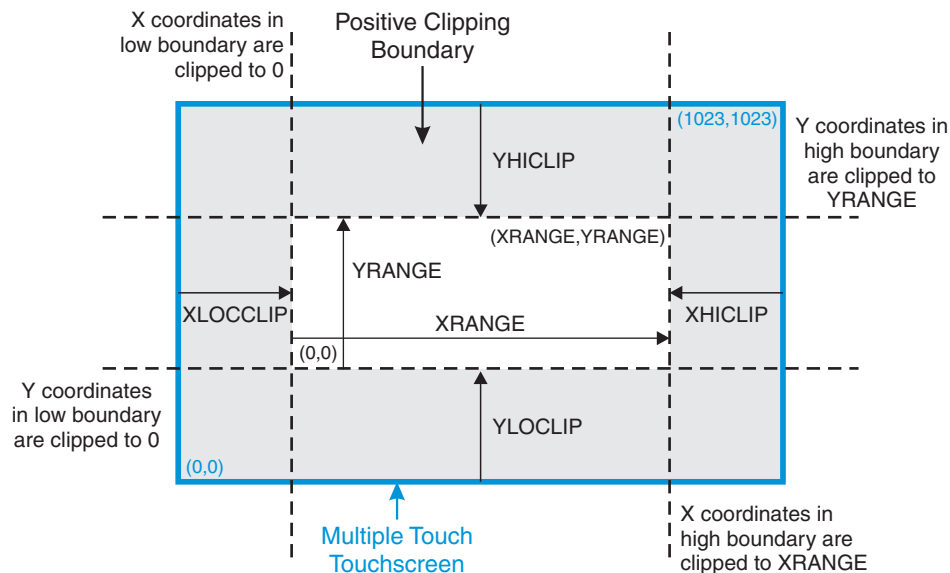
A setting of 0 (default) sets the resolution to 1023. These fields allow a touchscreen's position to match an LCD's resolution. For example, to set up a touchscreen to match an LCD with a resolution of 800 x 600, X RANGE and Y RANGE would be set to 799 and 599 respectively.

NOTE The maximum allowed value for X RANGE/Y RANGE is 65534.

The XLOCLIP/YLOCLIP and XHICLIP/YHICLIP fields set up a clipping boundary (illustrated by the dark grey regions in Figure 4-10 and Figure 4-11). Any touch position within this boundary has its X or Y coordinate clipped to the minimum or maximum value. These fields take a signed (two's complement) 8-bit value, allowing settings in the range -128 to +127. These fields work at 10-bit resolution (that is, a touchscreen range of 1023). This means that the maximum clipping values allow a clipping boundary one eighth of the screen width or height.

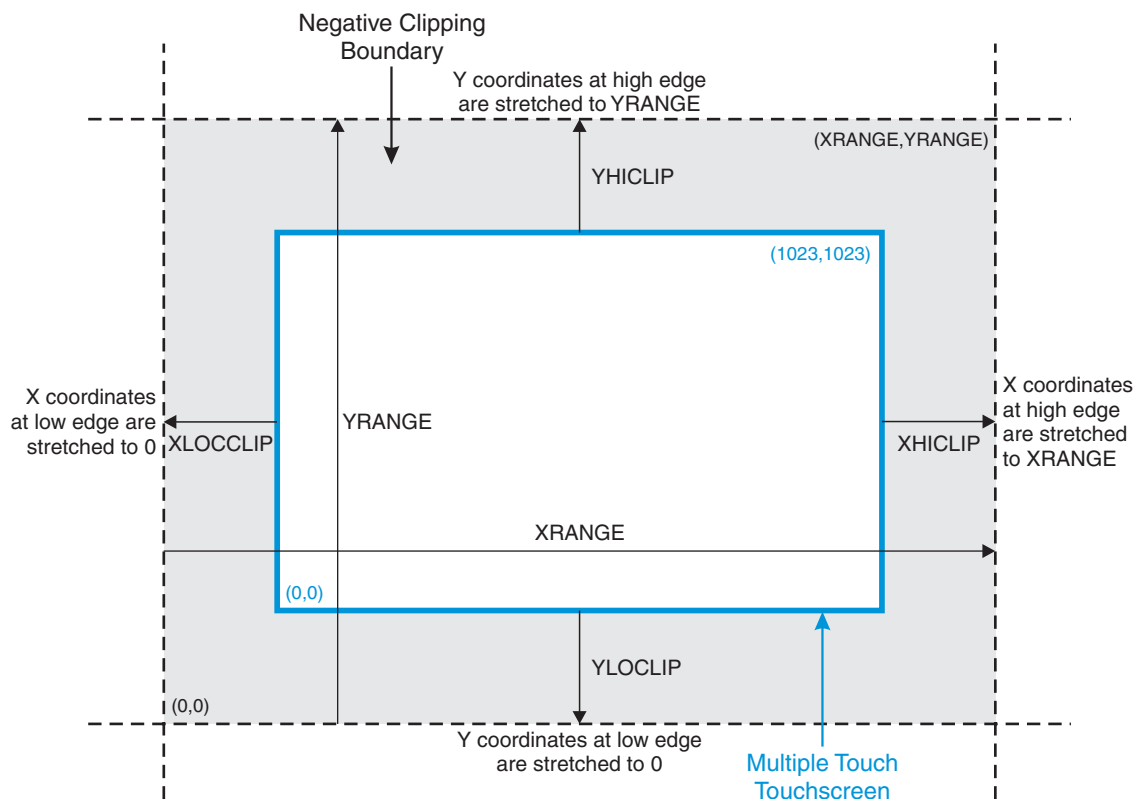
Positive values increase the size of the clipping boundary, moving the 0 and X RANGE/Y RANGE reported positions further inside the screen (see Figure 4-11). If the touch enters the clipping boundary, the coordinates are locked at the point at which the touch entered the region. The coordinates are updated only when the touch leaves the clipping boundary. This coordinate locking can be disabled using the DISLOCK field.

FIGURE 4-10: RESOLUTION FIELDS – POSITIVE CLIPPING BOUNDARY



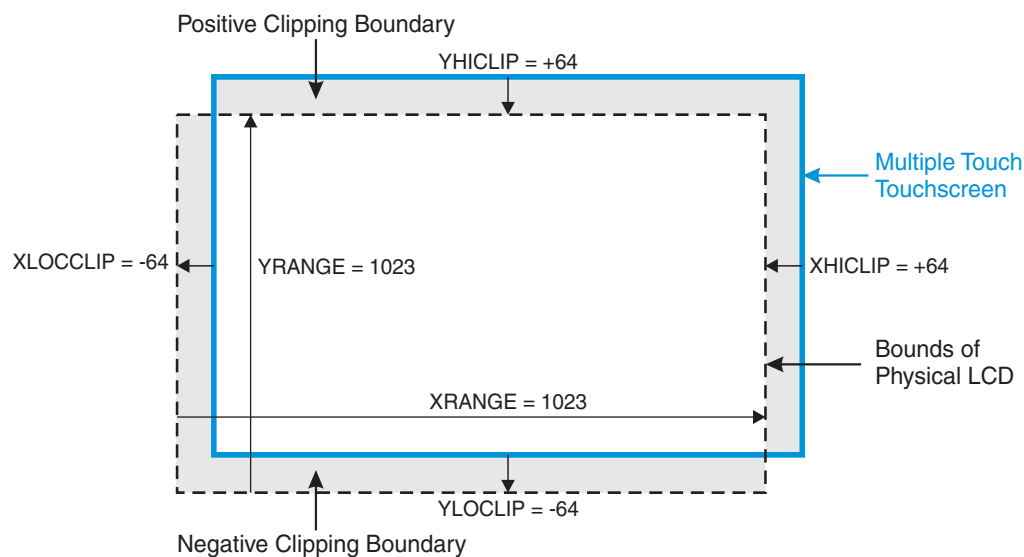
Negative values stretch the 0 and X RANGE/Y RANGE positions outside of the screen (see Figure 4-11). This limits the reported minimum and maximum value.

FIGURE 4-11: RESOLUTION FIELDS – NEGATIVE CLIPPING BOUNDARY



Adjusting the clipping boundary modifies the accuracy of the reported positions. For example, if the touchscreen is smaller than the physical LCD or is not aligned with the LCD, a combination of positive and negative clipping boundaries help align the reported zero point with the zero point of the LCD. An example of this is shown in Figure 4-12.

FIGURE 4-12: EXAMPLE RESOLUTION FIELDS



This example results in the coordinates shown in Figure 4-13 with the coordinate mapping shown in Figure 4-14.

FIGURE 4-13: EXAMPLE RESOLUTION FIELDS – RESULTING COORDINATES

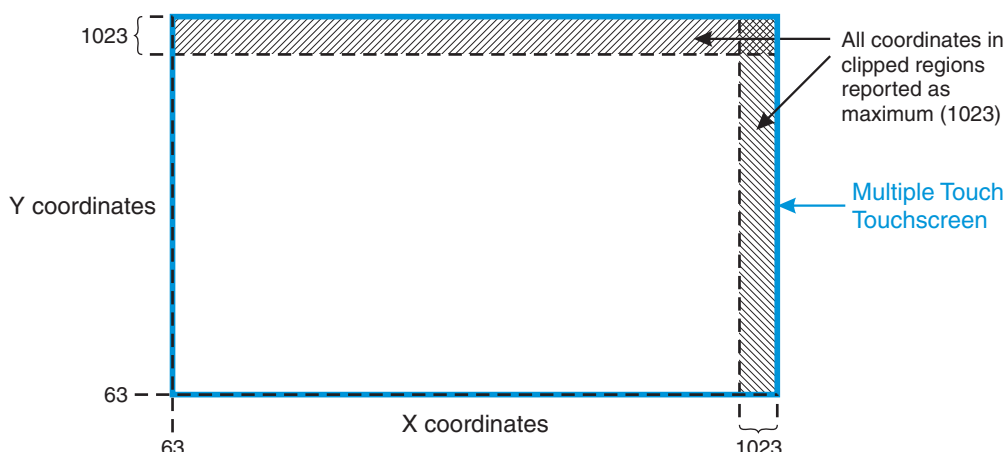
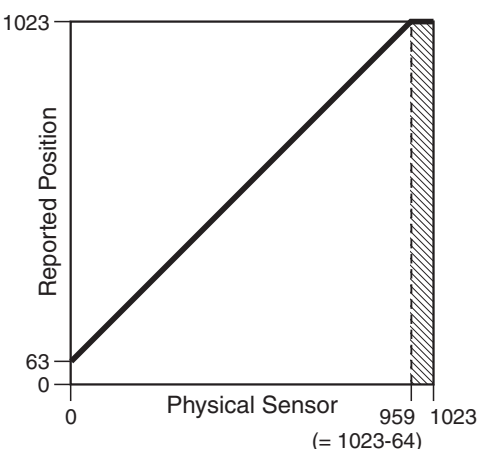
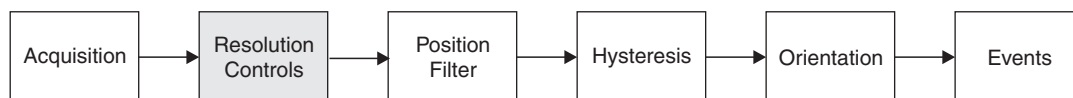


FIGURE 4-14: EXAMPLE RESOLUTION FIELDS – MAPPING OF REPORTED POSITIONS



These six resolution controls operate first on the positional data received from the touchscreen. The other touchscreen controls will be relative to the resolution set. For example, a hysteresis setting of 2 will have a different physical meaning on the same screen if the resolution is 200 or 2000. Figure 4-15 shows the processing order of all the touchscreen settings and shows how the resolution controls affect the other settings.

FIGURE 4-15: ORDER OF TOUCHSCREEN SETTINGS



XRANGE/YRANGE Range: 0 (1023; 10-bit resolution) to minimum and maximum as given on [page 58](#)

XLOCLIP/YLOCLIP/XHICLIP/YHICLIP Range: -128 to +127

XEDGECFG, DXEDGECFG and YEDGECFG Fields

The XEDGECFG and YEDGECFG fields work with the XEDGEDIST and YEDGEDIST fields respectively to perform edge correction. They also work with the XEDGEFGHI/YEDGEFGHI and XEDGEDISTHI/YEDGEDISTHI fields, if separate edge correction is required (see “[XEDGEFGHI, DXEDGEFGHI and YEDGEFGHI Fields](#)” and “[XEDGEDISTHI, DXEDGEDISTHI and YEDGEDISTHI Fields](#)”).

The DXEDGECFG field overrides XEDGECFG when Dual X measurements are being performed. It also overrides XEDGEFGHI if XEDGEFGHI ENABLE = 1 and DXEDGEFGHI ENABLE = 0 (that is, when separate edge correction is required for non Dual X measurements, but not for Dual X measurements).

The fields can also be overridden by other objects for certain types of touches.

See [Section 4.3.1.2 "Edge Correction"](#) for more information on how the various Multiple Touch Touchscreen T100 fields provide edge correction and which other objects contribute to edge correction functionality.

Note that these fields operate on the physical ITO matrix of the touchscreen, and are not affected by the logical orientation set by the CFG1 field.

CORRECTIONGRADIENT: Controls the correction gradient applied when the touch is between the edge of the sensor and the XEDGEDIST/YEDGEDIST coordinate when calculating the reported coordinate. It is used for correcting the nonlinearity seen at the edges of the sensor. A setting of 0 disables edge correction. A value of 1 to 63 sets the gradient value.

Range: 0 (disable), 1 to 63

Typical: 9

XEDGEDIST, DXEDGEDIST and YEDGEDIST Fields

The XEDGEDIST and YEDGEDIST fields work with the XEDGECFG and YEDGECFG fields to perform edge correction. These fields specify the onset point for when the edge correction calculation is applied. The value is specified as if the screen was in 10-bit mode and should be set to half of the typical touch diameter in pixels. This improves linearity at the edge of the screen on certain touchscreen designs. Note that these fields operate on the physical ITO matrix of the touchscreen, and are not affected by the logical orientation set by the CFG1 field.

They also work with the XEDGECFGHI/YEDGECFGHI and XEDGEDISTHI/YEDGEDISTHI fields, if separate edge correction is required (see ["XEDGECFGHI, DXEDGECFGHI and YEDGECFGHI Fields"](#) and ["XEDGEDISTHI, DXEDGEDISTHI and YEDGEDISTHI Fields"](#)).

The DXEDGEDIST field overrides XEDGEDIST when Dual X measurements are being performed.

It also overrides XEDGEDISTHI if XEDGECFGHI ENABLE = 1 and DXEDGECFGHI ENABLE = 0 (that is, when separate edge correction is required for non Dual X measurements, but not for Dual X measurements).

The fields can also be overridden by other objects for certain types of touches.

See [Section 4.3.1.2 "Edge Correction"](#) for more information on how the various Multiple Touch Touchscreen T100 fields provide edge correction and which other objects contribute to edge correction functionality.

For low-end positions, the corrected position is calculated as follows (for X):

$$\text{position} - (((\text{XEDGEDIST} - \text{position}) \times \text{XEDGECFG_CORRECTIONGRADIENT}) / 16)$$

OR (for Y):

$$\text{position} - (((\text{YEDGEDIST} - \text{position}) \times \text{YEDGECFG_CORRECTIONGRADIENT}) / 16)$$

For high-end positions, the corrected position is calculated as follows (for X):

$$\text{position} + (((\text{position} - (1023 - \text{XEDGEDIST})) \times \text{XEDGECFG_CORRECTIONGRADIENT}) / 16)$$

OR (for Y):

$$\text{position} + (((\text{position} - (1023 - \text{YEDGEDIST})) \times \text{YEDGECFG_CORRECTIONGRADIENT}) / 16)$$

NOTE These fields work at 10-bit resolution (that is, a touchscreen range of 1023).
--

Range: 0 to 255

Typical: Half expected touch diameter

GAIN Field

This field sets the gain of the analog circuits in front of the analog to digital converter (ADC) for mutual capacitance measurements.

The effective system gain is a complex combination of the GAIN value, the device gain calibration value and the operating conditions (for example, Cx value, use of XVDD or AVDD, single or dual X line pulsing, and so on). A Gain Calculator, where available, should therefore be used to explore different scenarios and determine a suitable gain value. Contact your Microchip representative for more information.

Range: 0 to 22

DXGAIN Field

This field specifies the Gain to be applied when operating in Dual X mode. A value of 255 in the field turns on auto correction of the gain in Dual X mode.

Range: 0 to 22, 255 (auto correction)

TCHTHR Field

This field specifies the Touch Threshold value. This defines how much a node's touch delta must be to qualify as a potential touch detection. See [Section 4.3.1 "Introduction"](#) for more details.

Range: 0 to 255 (in units of 8-bit deltas)

Typical: 30 to 80

TCHHYST Field

This field controls the level of Touch Hysteresis applied to touch detections. See [Section 4.3.1 "Introduction"](#) for more details.

TCHHYST is capped internally to $(TCHTHR \times 0.75)$. A hysteresis greater than 75% of the Touch Threshold (TCHTHR) is therefore not possible. Note that, under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the Touch Threshold, as this can cause potential problems during touch lift off, double taps, and noise handling.

A setting of 0 means that there is no hysteresis.

Range: 0 (off), 1 to $(TCHTHR \times 0.75)$

INTTHR Field

This field configures the Internal Touch Tracking Threshold. This is the delta level (in 8-bit deltas) above which touches are internally tracked. The default value of 0 means $(TCHTHR - TCHHYST)/2$. See also ["INTTHRHYST Field"](#). See [Section 4.3.1 "Introduction"](#) for more details.

Range: 0 $((TCHTHR - TCHHYST)/2)$, 1 to $\text{MIN}(255, TCHTHR - TCHHYST)$

NOISESF Field

This field specifies the Noise Scaling Factor to apply to the Noise Suppression T72 minimum threshold when searching for touches at the Internal Touch Tracking Threshold (INTTHR).

This feature is present to allow large touches to not be broken up by the Noise Suppression T72 minimum threshold processing in noisy conditions (that is, noise causing a dip in the middle of a touch). In this situation MRGTHR should also be increased to ensure that the large touches are not split up into multiple touches.

Note that NOISESF is not applied when determining whether a touch should be tracked or not. That is, the touch's peak delta must be above the Noise Suppression T72 minimum threshold to be tracked. It is only applied when determining whether a "region of touch" is actually one or multiple touches once the touchscreen has decided to track it.

A setting of 0 turns the feature off. This gives the behavior of using the full Noise Suppression T72 minimum threshold when searching for touches at the internal touch tracking threshold.

Settings of 1 to 255 result in a scaling factor of $(\text{NOISESF}/256)$ being applied to the Noise Suppression T72 minimum threshold when searching for touches at the internal touch tracking threshold.

Range: 0 (no scaling factor), 1 to 255 (scaling factor in units of $1/256$)

CUTOFFTHR Field

This field specifies the threshold on top of $(TCHTHR - TCHHYST)$ above which touch signals are cut off. In other words the threshold is forced to be equal to $(TCHTHR - TCHHYST + \text{CUTOFFTHR})$. The purpose of this feature is to help resolve position jittering of a big touch under extremely noisy conditions.

Note that the cutoff threshold is for mutual capacitance processing and is applied only to the mutual capacitance deltas. When using self capacitance touch as the primary measurement, the touches are tracked using self capacitance touch processing and thresholds.

The value is specified in 8-bit deltas, where a setting of 0 (default) means no cut-off threshold is required.

Range: 0 (OFF), 1 to 255 (8-bit deltas)

MRGTHR Field

The Merge Threshold (MRGTHR) setting allows the host to tune the amount of unevenness in a measured area that the device still considers as one individual touch. Beyond this threshold the device considers there to be multiple touches present on the sensor matrix.

This value may be used to allow the grouping of a large and uneven touch (such as a user's palm) into a single tracked touch. Larger values allow more uneven touches to be tracked as a single touch, but reduce the minimum separation of two true touches. This field is specified in units of signal delta.

Range: 0 to 255

Typical: 5

MRGTHRADJSTR

This field allows the MRGTHR to be increased dynamically when the capacitive coupling of the touchscreen is poor. The value controls the aggressiveness of the increase in merge threshold based on the estimated coupling level. This feature is useful for touchscreens under floating conditions, when large single touches may be broken up due to retransmission effects.

A larger value for MRGTHRADJSTR means that the merge threshold will increase faster as more anti-touch becomes present within a touch region. The value of MRGTHRADJSTR is approximately equal to the applied MRGTHR when the touch/anti-touch ratio is 1.

Note that if MRGTHRADJSTR is less than MRGTHR, this field has no effect.

Range: 0 to 255

MRGHYST Field

The Merge Hysteresis (MRGHYST) setting is used in conjunction with the MRGTHR field. It provides a hysteresis to ensure that once two touches have merged (and a single touch is reported), the distance needed for separation before two touches are reported again is slightly larger than for convergence. This can help to stop oscillations between one and two touches being reported at the boundary of convergence. This field is specified in units of signal delta.

Range: 0 to 255

Typical: 5

DXTHRSF Field

This field specifies the Dual X Threshold Scaling Factor; that is, how much thresholds should be scaled by when dual X measurements are performed. The Threshold Scaling Factor is specified in units of 1/128, where the default value of 0 means 128.

Range: 0 (128), 1 to 255

TCHDIDOWN Field

This field configures the Touch Detect Integration Down setting. This, together with the NEXTTCHDI field (see ["NEXTTCHDI Field"](#)), is used to provide a debounce filter that is applied to each touch as it is being detected. This helps to suppress false detections caused by spurious events like electrical noise.

A per-touch counter is incremented each cycle that a touch is detected. When this counter reaches a preset limit the touch is finally declared to be present. If on any acquisition a delta is not seen to exceed the threshold level, the counter is cleared and the process has to start from the beginning. It takes TCHDIDOWN + 1 cycles from touchdown to when the first touch is actually reported via the $\overline{\text{CHG}}$ pin, with a minimum time of 2 cycles. Once there is a touch detected, the limit is changed to (TCHDIDOWN + 1 + NEXTTCHDI). This allows a shorter detection integration for the first touch detection.

The value is specified in fast acquisition cycles (with no sleep in between).

Range: 0 to 63

TCHDIUP Field

This field configures the Touch Detect Integration Up setting. This, together with the NEXTTCHDI field (see ["NEXTTCHDI Field"](#)), is used to provide a debounce filter that is applied to each touch on release.

A per-touch counter is decremented each cycle that the delta does not exceed the threshold level. If the delta does exceed the threshold, it is incremented again. When the counter reaches zero, the touch is finally declared to be out of detect.

If there is only one touch in detect, it takes TCHDIUP + 1 cycle for the touch to cease to be reported. If there are two or more touches in detect, however, it takes (TCHDIUP + 1 + NEXTTCHDI) cycles for the touches to cease to be reported, with a minimum time of 1 cycle. This allows a shorter release integration if there is only one touch detection.

The value is specified in fast acquisition cycles (with no sleep in between).

Range: 0 to 63

NEXTTCHDI Field

This field configures the Next Touch Detect Integration. This field, together with the TCHDIDOWN or TCHDIUP field, provides Touch Detect Integration. See the descriptions of the TCHDIDOWN and TCHDIUP fields for more detail.

The value is specified in fast acquisition cycles (with no sleep in between).

Range: 0 to 63

JUMPLIMIT Field

This field specifies a limit on the allowed change in touch speed per cycle. If this limit is exceeded, the touch is classified as a new touch and all touchdown logic takes place on the new touch.

Note that low values can hinder fast movements, such as flicks.

The range for this field is 0 to 255, where 0 is off (no limit) and 1 to 255 is the allowed change in speed in units of the reported touchscreen resolution per millisecond.

Range: 0 (off, no limit), 1 to 255

MOVFILTER Field

This field controls the position filter that is used for filtering the reported touch position.

SPEEDRESP: Controls the system response to touch speed. Increasing this value results in the system response increasing with touch speed and less lag.

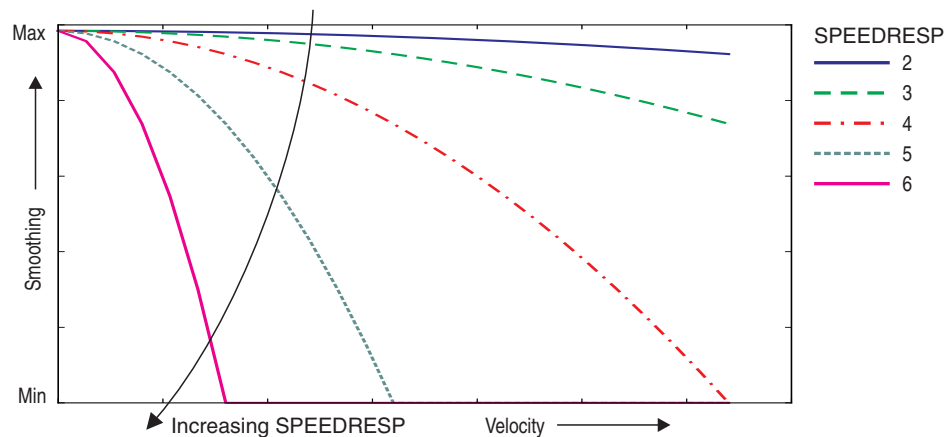
[Figure 4-16](#) shows how varying the SPEEDRESP setting affects the level of smoothing applied. Increasing the value of SPEEDRESP results in the application of less smoothing for the same estimated touch velocity.

Note that [Figure 4-16](#) is for a particular value of MOVSMOOTH. MOVSMOOTH sets the maximum amount of smoothing to apply (at very low speeds), while SPEEDRESP controls how quickly that smoothing falls off as movement speed increases.

The range for SPEEDRESP is 0 to 8, where a value of 0 means 4.

SPEEDRESP Range: 0 (4), 1 to 8.

FIGURE 4-16: VARYING SPEEDRESP



MEDOFF: Disables the use of a median filter for pre-processing the position information. The median filter introduces a small delay to the system. If the system noise levels are low and the minimum delay is required this should be disabled.

DISABLE: Disables the movement filter.

MOVSMOOTH Field

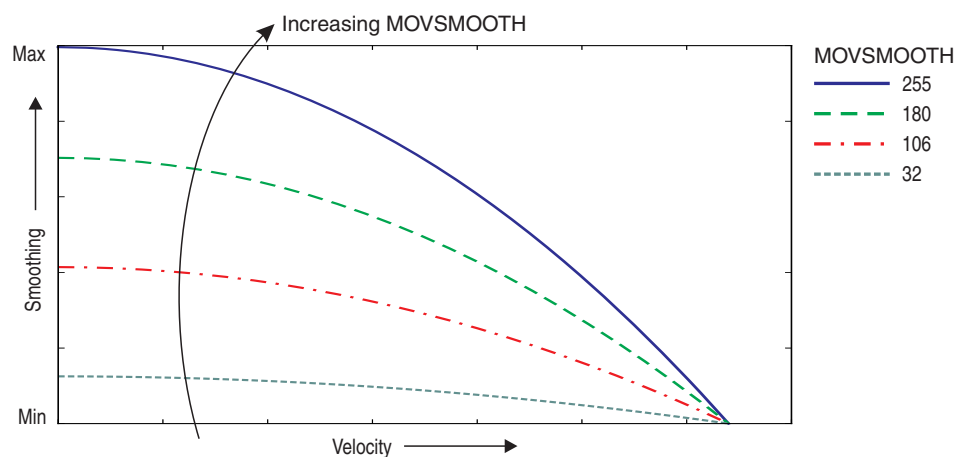
The Smooth Control limits the maximum amount of filtering that can be applied to the touch screen position (that is, the level of smoothing). Increasing the setting results in an increase in the maximum filtering. If the resolution of the touchscreen is modified this value may need adjusting to achieve the same level of performance.

Figure 4-17 shows the variation in the level of position smoothing applied with variation in speed for different settings of MOVSMOOTH.

Range: 0 (193), 1 to 255

Typical: 65 to 193

FIGURE 4-17: VARYING MOVSMOOTH



MOVPRED Field

This field determines the level of prediction that the algorithm applies by controlling the amount of history used to estimate the position.

Increasing the MOVPRED setting increases the level of prediction at a given speed, as more data is used to predict the position.

The amount of prediction employed is dependent on the speed of travel. As the touch speed reduces the level of prediction is also reduced.

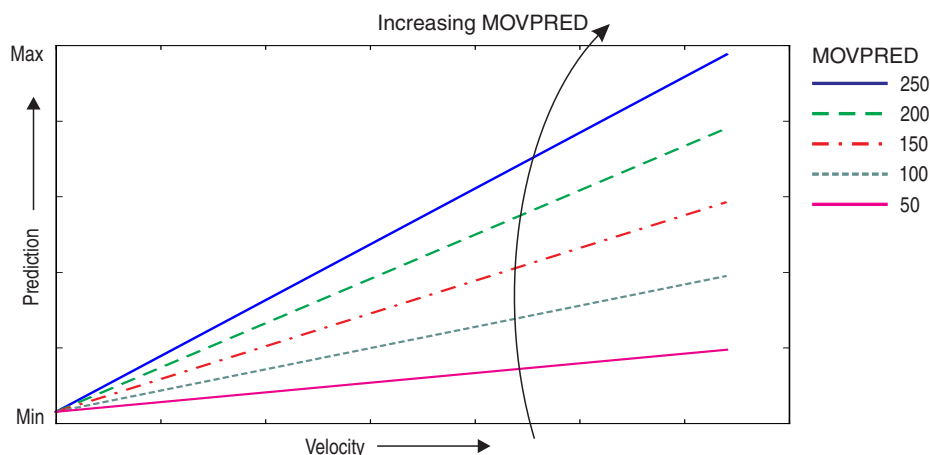
Note that setting too high a value may result in system overshooting a position if the touch changes direction abruptly. Values outside the typical range are not recommended.

Figure 4-18 shows the effect of varying the MOVPREd setting on the level of prediction applied dependent on the touch velocity.

Range: 0 (129), 1 to 255

Typical: 65 to 161

FIGURE 4-18: VARYING MOVPREd



MOVHYSTI and MOVHYSTN Fields

These fields are applied to the reported touch positions to provide simple filtering.

When the user first touches the touchscreen (that is, when the user's finger has just touched the touchscreen), the Initial Movement Hysteresis (MOVHYSTI) setting is applied to the reported position. Once the user's finger starts to move, the MOVHYSTI must be exceeded in a positive or negative X or Y direction (that is, in one of four directions) for position updates to be reported. This allows the host to differentiate between an intended drag and a simple press.

Once the MOVHYSTI limit has been exceeded, the Next Movement Hysteresis (MOVHYSTN) setting is applied to the calculated positions. The MOVHYSTN is applied separately to the X and Y axes and only in the opposite direction to the most recently reported movement in each axis. This allows small movements in the direction of travel to be reported while small backwards movements are not reported. A simple jitter filter, therefore, can be implemented to prevent unwanted movement reports due to small movement changes (jitter) as the position moves back and forth (providing that the movement is less than the hysteresis applied).

The MOVHYSTN setting is internally reduced while the touch is moving. This avoids "steps" when drawing, whilst still allowing jitter to be suppressed when the touch is stationary. The effect of this control is reduced when the touch is in motion. The extent of this reduction is determined by the MOVFILTER SPEEDRESP control (which can be overridden by MOVHYSTCFG SPEEDRESP) and the MOVHYSTCFG MINSPEED control. If the movement position filter is turned off, there will be no internal reduction effect on moving touches.

If MOVHYSTI is not required, set it to the same value as the MOVHYSTN setting.

The units for these fields are the least significant bits of position.

Range: 0 to 4095 (configuration error if out of range)

MOVHYSTN Typical: 0 to 10

MOVHYSTI Typical: 50

AMPLHYST Field

This field configures the hysteresis applied to the AMPL touch status reports (if enabled) that are sent to the host. This avoids excessive messaging to the host (see ["TCHAU Field"](#)). The units are in 8-bit deltas.

Range: 0 to 255

SCRAREAHYST Field

This field configures the hysteresis (in nodes) applied to the TCHAREA, ATCHAREA and INTTHRAREA screen status reports (if enabled) that are sent to the host to avoid excessive messaging to the host (see ["SCRAUX Field"](#)).

Range: 0 to 255

INTTHRHYST Field

This field controls the Internal Touch Tracking Threshold Hysteresis that is applied to the internally tracked touch detections. See [Section 4.3.1 "Introduction"](#) for more details.

INTTHRHYST is capped internally to $(INTTHR \times 0.75)$. A hysteresis greater than 75% of the Internal Touch Tracking Threshold (INTTHR) is therefore not possible. Note that under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the Internal Touch Tracking Threshold, as this can cause potential problems during touch lift off, double taps, and noise handling.

A setting of 0 means that there is no hysteresis.

Range: 0 (off), 1 to $(INTTHR \times 0.75)$

XEDGEFCGHI, DXEDGEFCGHI and YEDGEFCGHI Fields

The XEDGEFCGHI and YEDGEFCGHI fields work with the XEDGEDISTHI and YEDGEDISTHI fields respectively to perform edge correction on the high sense lines. Note that these fields operate on the physical ITO matrix of the touchscreen, and are not affected by the logical orientation set by the CFG1 field.

The DXEDGEFCGHI field overrides either XEDGEFCGHI or YEDGEFCGHI when Dual X measurements are being performed (depending on whether uniform or separate edge correction is chosen by the XEDGEFCGHI ENABLE bit).

The XEDGEFCGHI, DXEDGEFCGHI and YEDGEFCGHI fields can also be overridden by other objects for certain types of touches.

See [Section 4.3.1.2 "Edge Correction"](#) for more information on how the various Multiple Touch Touchscreen T100 fields provide edge correction and which other objects contribute to edge correction functionality.

CORRECTIONGRADIENT: Controls the correction gradient applied when the touch is between the edge of the sensor and the XEDGEDISTHI/YEDGEDISTHI coordinate when calculating the reported coordinate. It is used for correcting the nonlinearity seen at the edges of the sensor. A setting of 0 disables edge correction. A value of 1 to 63 sets the gradient value.

CORRECTIONGRADIENT Range: 0 (disable), 1 to 63

CORRECTIONGRADIENT Typical: 9

ENABLE: Enables separate edge correction.

If this bit is set to 1, XEDGEFCGHI, DXEDGEFCGHI or YEDGEFCGHI, as appropriate, can be used specify the separate edge correction for the high sense line edge of the Multiple Touch Touchscreen T100 X or Y axis, as appropriate (along with XEDGEDISTHI, DXEDGEDISTHI or YEDGEDISTHI). This allows separate control of the edge correction that is applied to the top and bottom edges, so that sensors with different non-linear responses at the top and bottom of the X or Y axis can be corrected for.

If this bit is set to 0, uniform edge correction is used, as determined by the XEDGEFCG, DXEDGEFCG and YEDGEFCG fields.

XEDGEDISTHI, DXEDGEDISTHI and YEDGEDISTHI Fields

The XEDGEDISTHI, DXEDGEDISTHI and YEDGEDISTHI fields work with the XEDGEFCFGHI, DXEDGEFCFGHI and YEDGEFCFGHI fields to perform separate edge correction. These fields specify the onset point for when the edge correction calculation is applied. The value is specified as if the screen was in 10-bit mode and should be set to half of the typical touch diameter in pixels. This improves linearity at the edge of the screen on certain touchscreen designs. Note that these fields operate on the physical ITO matrix of the touchscreen, and are not affected by the logical orientation set by the CFG1 field.

The DXEDGEDISTHI field overrides either XEDGEDISTHI or XEDGEDIST when Dual X measurements are being performed (depending on whether uniform or separate edge correction is chosen by the XEDGEFCFGHI ENABLE bit; see “[XEDGEFCFGHI, DXEDGEFCFGHI and YEDGEFCFGHI Fields](#)”). That is, it overrides XEDGEDIST if XEDGEFCFGHI ENABLE is set to 0, and it overrides XEDGEDISTHI if XEDGEFCFGHI ENABLE is set to 1.

See [Section 4.3.1.2 “Edge Correction”](#) for more information on how the various Multiple Touch Touchscreen T100 fields provide edge correction and which other objects contribute to edge correction functionality.

With separate edge correction, the corrected position for high-end positions is calculated as follows (for X):

$$\text{position} + (((\text{position} - (1023 - \text{XEDGEDISTHI})) \times \text{XEDGEFCFGHI_CORRECTIONGRADIENT}) / 16)$$

OR (for Y):

$$\text{position} + (((\text{position} - (1023 - \text{YEDGEDISTHI})) \times \text{YEDGEFCFGHI_CORRECTIONGRADIENT}) / 16)$$

Note that these fields work at 10-bit resolution (that is, a touchscreen range of 1023). See “[XEDGEDIST, DXEDGEDIST and YEDGEDIST Fields](#)” for the calculated correction on low-end positions).

Range: 0 to 255

Typical: Half expected touch diameter

CFG2 Field

CONFTHR: Specifies a debounce filter for the TCHSTATUS TYPE of a reporting touch changing to FINGER (see “[TCHSTATUS Field](#)”). This delays a contacting touch class from being reported as a finger under very noisy conditions, which would then block other class transitions.

The threshold is specified in acquisition cycles. Higher values will introduce higher delays between the transitions to finger touch type and may lead to missing touches during finger taps.

CONFTHR Range: 0 to 7 (acquisition cycles)

MOVHYSTCFG Field

This field provides additional control over the scaling of Next Movement Hysteresis (see “[MOVHYSTI and MOVHYSTN Fields](#)”) with touch speed. SPEEDRESP controls the rate of scaling and MINSPEED controls the speed at which the scaling starts.

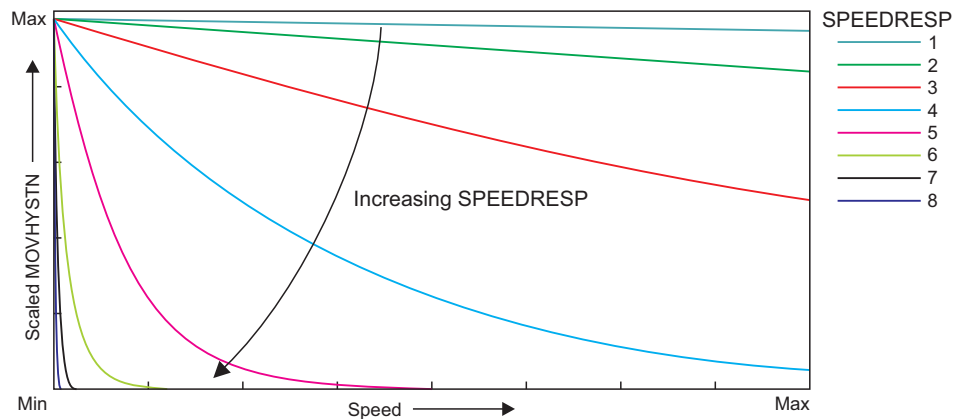
SPEEDRESP: Configures the Speed Response. This controls the rate at which movement hysteresis is reduced with increasing touch speed after the minimum speed (specified by MINSPEED) has been reached. Increasing the value of SPEEDRESP increases the rate at which Movement Hysteresis is reduced with increasing speed (see [Figure 4-19](#)).

By using this control with MOVHYSTN, the touchscreen can be configured to apply a large movement hysteresis to slow moving touches but a small movement hysteresis to faster moving touches.

If the movement filter is disabled this control has no effect.

The value for SPEEDRESP is in the range 0 to 8, where the default value of 0 means use MOVFILTER SPEEDRESP (see “[MOVFILTER Field](#)”). Values above 8 are capped to 8.

FIGURE 4-19: EFFECT OF SPEEDRESP ON MOVEMENT HYSTERESIS



SPEEDRESP Range: 0 (Use MOVFILTER SPEEDRESP), 1 to 8

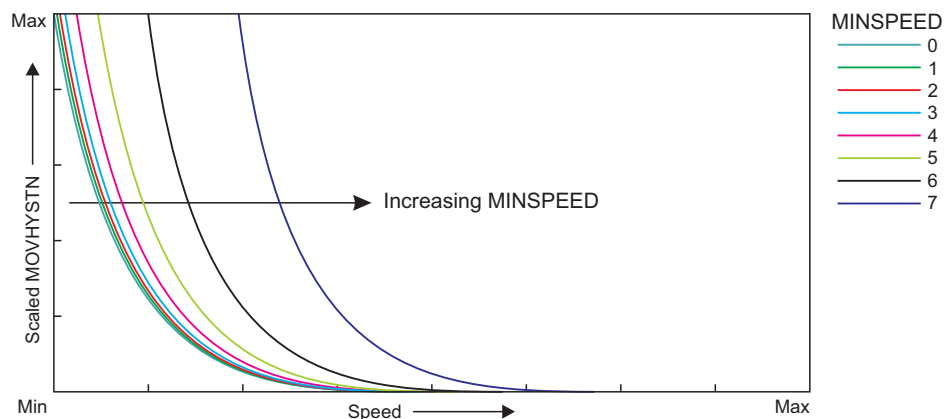
SPEEDRESP Typical: 4

MINSPEED: This control adjusts the minimum touch speed that must be reached before movement hysteresis scaling occurs. Once the minimum speed has been reached, any further increase in touch speed will result in the Movement Hysteresis being reduced at the rate determined by SPEEDRESP (see Figure 4-20). Increasing the value of MINSPEED increases the minimum touch speed in an exponential fashion, such that increasing MINSPEED by 1 doubles the required minimum touch speed.

This control allows the system to be configured so that the full value of MOVHYSTN is used even when position jitter prevents the touch speed reaching 0, while retaining the rate of MOVHYSTN reduction with speed.

If the movement filter is disabled this control has no effect.

FIGURE 4-20: EFFECT OF MINSPEED ON MOVEMENT HYSTERESIS



MINSPEED Range: 0 (Disabled), 1 to 7

MINSPEED Typical: 2

AMPLCOEFF Field

This field specifies the Scaled Amplitude Coefficient. Together with the AMPLOFFSET field, this configures the scaling applied to the AMPL touch status reports (if enabled) that are sent to the host (see "TCHAU Field"). The scaling is as follows:

$$\text{AMPL}_{\text{scaled}} = (\text{AMPLCOEFF} \times \text{AMPL}_{\text{original}}) + \text{AMPLOFFSET}$$

where $\text{AMPL}_{\text{scaled}}$ has a minimum value of 1 and is saturated to a maximum of 255.

AMPLCOEFF is the multiplier for the amplitude and is a 5.3 format scale factor, giving a range of 0.125 to 31.875. The default value of 0 means 8 (= 1.0).

The scaling has no effect on internal touch processing.

Range: 0 (8 = 1.0), 1 to 255 (units of 0.125; 1/8)

AMPLOFFSET Field

This field specifies the Scaled Amplitude Offset. This is the offset for the touch amplitude. See [“AMPLCOEFF Field”](#) for more details.

Range: –128 to 127

JUMPLIMITMOV Field

This field configures the Jump Limit for Moving Touches.

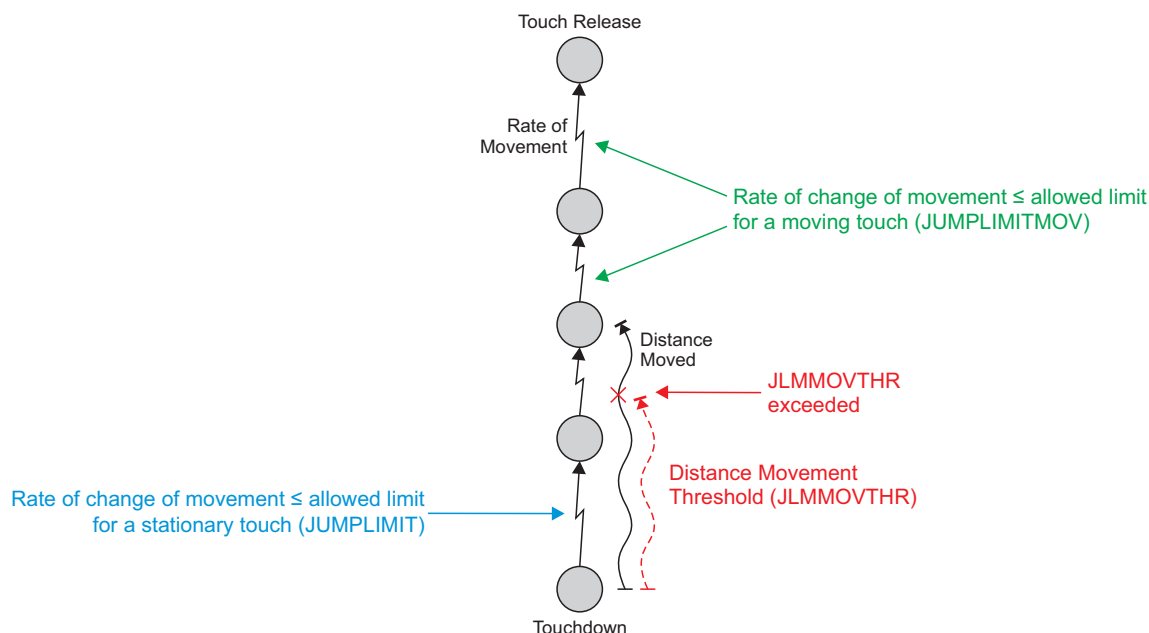
This is the limit on the allowed change in touch speed per cycle of moving touches (that is, touches that have exceeded JLMMOVTHR; see [“JLMMOVTHR Field”](#)). This field operates in a similar way to the JUMPLIMIT field (see [“JUMPLIMIT Field”](#)), but it is only applied to touches the movement of which has exceeded JLMMOVTHR.

JUMPLIMIT and JLMMOVTHR must both be configured to non zero values for the JUMPLIMITMOV field to have an effect.

The range for this field is 0 to 255, where 0 means no limit and 1 to 255 is the allowed change in speed. The value is specified in units of the reported touchscreen resolution per millisecond.

Range: 0 (no limit), 1 to 255 (reported position units per ms)

FIGURE 4-21: JUMP LIMIT THRESHOLDS FOR MOVING TOUCHES



JLMMOVTHR Field

This field configures the Movement Threshold for the Jump Limit for Moving Touches. This is the minimum distance a touch must move before JUMPLIMITMOV is used instead of JUMPLIMIT (see [“JUMPLIMIT Field”](#)). Note that the distance is along the actual path of the movement, and not a straight line.

The range for this field is 0 to 4095, where 0 disables the use of JUMPLIMITMOV and 1 to 4095 specifies a distance in reported position units.

Range: 0 (disabled), 1 to 4095 (distance in reported position units)

JLMMOVINTTHR Field

This field configures the Movement Integration Threshold for the Jump Limit for Moving Touches. This is the minimum touch movement that can count toward the total movement against which JLMMOVTHR is compared. By setting this control to a value larger than the stationary jitter radius, this value can be used to prevent the jitter on taps and stationary touches from being accumulated and causing JLMMOVTHR to be exceeded.

The range for this field is 0 to 255, specified in reported position units.

Range: 0 to 255 (threshold in reported position units)

4.3.3 CONFIGURATION CHECKS

A Multiple Touch Touchscreen T100 object causes a configuration check to be performed in the following circumstances:

- When the object is enabled (that is, the ENABLE bit is set in the CTRL field)
- If the object is enabled, when certain fields are changed

In addition, some fields will cause an automatic recalibration to be performed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3 "Command Processor T6 Object"](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

The XORIGIN/YORIGIN and XSIZE/YSIZE settings should be checked to ensure that:

- Part of the object is not placed off the edge of the sensor matrix. The matrix size is specified in the Information Block (see [Section 1.3 "Information Block"](#))
- The object does not overlap with other enabled touch objects
- In the case of Multiple Touch Touchscreen T100 instance 0, the object does not occupy the same Y line as another enabled object if the Shieldless T56 is also enabled.
- XSIZE and YSIZE are greater than 2.

TABLE 4-21: CONFIGURATION CHECKS FOR MULTIPLE TOUCH TOUCHSCREEN T100 (TOUCH_MULTITOUCHSCREEN_T100)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes ⁽¹⁾ (2)	None
CFG1	No	No	None
SCRAUX	No	No	None
TCHAUX	No	No	None
TCHEVENTCFG	No	No	None
AKSCFG	No	No	None
NUMTCH	No	No	None
XYCFG	Yes	No	None
XORIGIN	Yes	Yes	Error if out of range, or if non zero and self-capacitance measurements are enabled.
XSIZE	Yes	Yes	Error if out of range.
XPITCH	Yes	No	None
XLOCLIP	Yes	No	None
XHICLIP	Yes	No	None
XRANGE	Yes	No	None
XEDGECFG	No	No	None
XEDGEDIST	No	No	None
DXXEDGECFG	No	No	None
DXXEDGEDIST	No	No	None

**TABLE 4-21: CONFIGURATION CHECKS FOR MULTIPLE TOUCH TOUCHSCREEN T100
(TOUCH_MULTITOUCHSCREEN_T100)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
YORIGIN	Yes	Yes	Error if out of range, or if non zero and self-capacitance measurements are enabled.
YSIZE	Yes	Yes	Error if out of range
YPITCH	Yes	No	None
YLOCLIP	Yes	No	None
YHICLIP	Yes	No	None
YRANGE	Yes	No	None
YEDGECFG	No	No	None
YEDGEDIST	No	No	None
GAIN	Yes	Yes	None
DXGAIN	Yes	Yes	None
TCHTHR	Yes	No	None
TCHHYST	Yes	No	None
INTTHR	Yes	No	None
NOISESF	No	No	None
CUTOFFTHR	No	No	None
MRGTHR	Yes	No	None
MRGTHRADJSTR	Yes	No	None
MRGHYST	No	No	None
DXTHRSF	No	No	None
TCHDIDOWN	No	No	None
TCHDIUP	No	No	None
NEXTTCHDI	No	No	None
JUMPLIMIT	Yes	No	None
MOVFILTER	Yes	No	None
MOVSMOOTH	Yes	No	None
MOVPRD	Yes	No	None
MOVHYSTI	No	No	None
MOVHYSTN	No	No	None
AMPLHYST	No	No	None
SCRAREAHYST	No	No	None
INTTHRHYST	Yes	No	None
XEDGEFGHI	No	No	None
XEDGEDISTHI	No	No	None
DXXEDGEFGHI	No	No	None
DXXEDGEDISTHI	No	No	None
YEDGEFGHI	No	No	None
YEDGEDISTHI	No	No	None
CFG2	No	No	None
MOVHYSTCFG	No	No	None

**TABLE 4-21: CONFIGURATION CHECKS FOR MULTIPLE TOUCH TOUCHSCREEN T100
(TOUCH_MULTITOUCHSCREEN_T100)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
AMPLCOEFF	No	No	None
AMPLOFFSET	No	No	None
JUMPLIMITMOV	Yes	No	None
JLMMOVTHR	No	No	None
JLMMOVINTTHR	No	No	None

1 If the ENABLE bit is toggled on or off.

2 If the ENABLE bit is cleared and *all* of the touch objects are now disabled, no calibration takes place.

4.3.4 MESSAGES

The Multiple Touch Touchscreen T100 object reports the following screen and touch status information:

- Screen status information, such as the number of nodes affected and the number of reported touches
- Finger touches detected by the Multiple Touch Touchscreen T100 object
- Grip suppression by the Grip Suppression T40 object
- Screen suppression by the Touch Suppression T42 object
- Glove touches detected by the Glove Detection T78 object

NOTE

- To be able to track and report touches in self capacitance measurements, the Multiple Touch Touchscreen T100 object needs to have data for both axes (see “CTRL Field” and [Appendix A “Measurement Processing on mXT144U”](#)).

There are three types of message reported from a Multiple Touch Touchscreen T100. The first report ID from a Multiple Touch Touchscreen T100 is for the global screen status messages that indicate the state of the touchscreen. The second report ID is reserved. Subsequent report IDs are for each of the reportable tracked touches.

4.3.4.1 First Report ID – Screen Status Messages

The first report ID from the Multiple Touch Touchscreen T100 object is used to output screen status messages.

[Table 4-22](#) shows the message data for the first report ID.

**TABLE 4-22: MESSAGE DATA FOR MULTIPLE TOUCH TOUCHSCREEN T100
(TOUCH_MULTITOUCHSCREEN_T100)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	SCRSTATUS	DETECT	SUP	Reserved					
2-9	AUXDATA[]	Auxiliary data							

A screen status message is triggered when one of the reported data fields reaches a particular trigger value, as summarized in [Table 4-23](#).

TABLE 4-23: MESSAGE TRIGGERS – SCREEN STATUS MESSAGES

Data Field		Effect on Message Generation
SCRSTATUS Bits	AUXDATA[] Fields	
SUP		Message generated if value changes (that is, if full screen suppression is in effect or no longer in effect)
DETECT		Message generated if value changes (that is, if there are any touches reporting or not).
	NUMRPTTCH	Message generated if value changes

TABLE 4-23: MESSAGE TRIGGERS – SCREEN STATUS MESSAGES

Data Field		Effect on Message Generation
SCRSTATUS Bits	AUXDATA[] Fields	
	TCHAREA	Message generated if value > SCRAREAHYST
	ATCHAREA	Message generated if value > SCRAREAHYST
	INTTHRAREA	Message generated if value > SCRAREAHYST

SCRSTATUS Field

This field indicates the status of the touchscreen.

DETECT: This bit is set to indicate that there is at least one reporting touch on the touchscreen.

SUP: This bit is set to indicate that full screen suppression is in effect.

AUXDATA[] Fields

These fields report user-configurable information about the screen. The items to report, and thereby the content of the AUXDATA[] fields, is determined by which bits are enabled in the SCRAUX configuration field (see “SCRAUX Field”).

4.3.4.2 Second Report ID – Reserved

The second report ID is reserved for future use.

TABLE 4-24: MESSAGE DATA FOR MULTIPLE TOUCH TOUCHSCREEN T100 (TOUCH_MULTITOUCHSCREEN_T100)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1 – 9	Reserved	Reserved							

4.3.4.3 Subsequent Report IDs – Touch Status Messages

Subsequent report IDs are used to output the touch status messages. There are 5 report IDs, one for each of the 5 touches. Table 4-25 shows the message data for a Multiple Touch Touchscreen T100 object for these report IDs.

TABLE 4-25: MESSAGE DATA FOR MULTIPLE TOUCH TOUCHSCREEN T100 (TOUCH_MULTITOUCHSCREEN_T100)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	TCHSTATUS	DETECT	TYPE			EVENT			
2	XPOS	X position LSByte							
3		X position MSByte							
4	YPOS	Y position LSByte							
5		Y position MSByte							
6 – 9	AUXDATA[]	Auxiliary data							

A touch status message is triggered when one of the reported data fields reaches a particular trigger value or the touch type changes, as summarized in Table 4-26.

TABLE 4-26: MESSAGE TRIGGERS – TOUCH STATUS MESSAGES

Data Field		Effect on Message Generation
TCHSTATUS Bits	AUXDATA[] Fields	
EVENT		Message generated if there is an event
TYPE		Message generated if TYPE changes
	VECT	No effect on message generation
	AMPL	Message generated if delta change > AMPLHYST
	AREA	No effect on message generation

TABLE 4-26: MESSAGE TRIGGERS – TOUCH STATUS MESSAGES

Data Field		Effect on Message Generation
TCHSTATUS Bits	AUXDATA[] Fields	
	HW	No effect on message generation
	PEAK	No effect on message generation
	AREAHW	No effect on message generation

NOTE

- The RPTEACHCYCLE bit in the CFG1 field can be used to generate a touch status message for each touch that is reporting, irrespective of whether any triggering events have happened. Note that this applies to all types of touches. See “CFG1 Field” for more information.
- Only a single touch will be reported if a touch is present in a touchscreen area that Retransmission Compensation T80 has detected as having moisture.

Note:

TCHSTATUS Field

This field indicates the status of the touch.

EVENT: These bits indicate the event that has just occurred for this finger touch (see [Table 4-27](#)).

TABLE 4-27: EVENT CODES

Event	Name	Description
0	NO EVENT	No specific event has occurred
1	MOVE	The touch position has changed
2	UNSUP	The touch has just been unsuppressed by the touch suppression features of other objects
3	SUP	The touch has been suppressed by the touch suppression features of other objects
4	DOWN	The touch has just come within range of the sensor
5	UP	The touch has just left the range of the sensor
6	UNSUPSUP	Both UNSUP and SUP events have occurred (in either order)
7	UNSUPUP	Both UNSUP and UP events have occurred (in either order)
8	DOWNSUP	Both DOWN and SUP events have occurred (in either order)
9	DOWNUP	Both DOWN and UP events have occurred (in either order)

TYPE: These bits indicates the type of touch that is being reported (see [Table 4-28](#)). A touch status message is generated if the type changes.

TABLE 4-28: TOUCH TYPES

Type	Name	Description
0	RESERVED	Reserved for future use.

TABLE 4-28: TOUCH TYPES (CONTINUED)

Type	Name	Description
1	FINGER	The touch is considered to be a finger that is contacting the screen.
5	GLOVE	The touch is a glove touch. Note that touches will be reported as GLOVE events only if the CTRL GLOVERPTEN bit in the Glove Detection T78 object is set to 1; otherwise they will be reported as FINGER events.
6	LARGE TOUCH	The touch is a suppressed large touch. When the Touch Suppression T42 CFG SUPTCHRPTEN bit is set, a touch that would normally be suppressed is reported. When SUPTCHRPTEN is cleared, the suppression method returns to normal behavior.

DETECT: This bit is set to 1 to indicate that the touch is reporting and present within the range of the sensor.

XPOS Field

This field reports the X position.

YPOS Field

This field reports the Y position.

AUXDATA[] Fields

These fields hold the user configurable (type specific) information about the touch, as determined by the TYPE code in the TCHSTATUS field.

NOTE AUXDATA[] may contain zero value fields and in some cases these may be treated as special values within the product's operating system. It may be necessary, therefore, for the product's driver code to replace certain zero values with non zero values (for example, if the value is used for major axis length in Android devices).

If the touch is a finger touch, the content of these fields is determined by which bits are enabled in the TCHAUX configuration field. The AUXDATA[] bytes are filled in the same order as the TCHAUX bits, depending on which bits are enabled. See "[TCHAUX Field](#)" for more details.

4.3.5 DIAGNOSTIC DEBUG DATA

The diagnostic debug data modes for the Multiple Touch Touchscreen T100 object is shown in [Table 4-29](#). The diagnostic debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in [Table 4-29](#).

The debug modes are described in the following sections.

TABLE 4-29: DIAGNOSTIC DEBUG COMMANDS

Command	Name	Description
0x36	Touch Status Data Mode	The Diagnostic Debug T37 object holds the status data associated with each touch. See Section 4.3.5.1 "Touch Status Data Mode" .
0xF4	Touchscreen Mode	The Diagnostic Debug T37 object holds the current configuration data for the Multiple Touch Touchscreen T100 object. Specifically, it gives the current touch threshold in use as an 8-bit value. See Section 4.3.5.2 "Touchscreen Mode" .

4.3.5.1 Touch Status Data Mode

Figure 4-22 shows the organization of the data in the pages for the Touch Status Data mode.

FIGURE 4-22: DIAGNOSTIC DEBUG T37 FIELDS – TOUCH STATUS DATA MODE

Page	Data Index	Data[]
Page 0	0 – 18	Touch status for touch 0
	19 – 37	Touch status for touch 1
	38 – 56	Touch status for touch 2
	57 – 75	Touch status for touch 3
	76 – 94	Touch status for touch 4
	95 – 127	Reserved

Assumes page size = 128

Where the touch status data for each touch consists of the following:

Byte	Data
0	TCHSTATUS from Multiple Touch Touchscreen T100 messages (see “TCHSTATUS Field”)
1	X filtered position LSByte
2	X filtered position MSByte
3	Y filtered position LSByte
4	Y filtered position MSByte
5	X unfiltered position whole part LSByte
6	X unfiltered position whole part MSByte
7	Y unfiltered position whole part LSByte
8	Y unfiltered position whole part MSByte
9	Reserved
10	Reserved
11	Reserved
12	Reserved
13	X unfiltered position fractional part
14	Y unfiltered position fractional part
15 – 18	Reserved

4.3.5.2 Touchscreen Mode

Figure 4-23 shows the organization of the data in the pages for the touchscreen mode. This shows the current configuration data for the Multiple Touch Touchscreen T100 object.

FIGURE 4-23: DIAGNOSTIC DEBUG T37 DATA – TOUCHSCREEN MODE

Page	Data Index	Data[]
Page 0	0	Touch Threshold
	1 – 6	Reserved
	7	Coupling Level (from Retransmission Compensation T80)
	8 – 127	Reserved

Assumes page size = 128

5.0 SIGNAL PROCESSING OBJECTS

5.1 Introduction

Signal processing objects process the data from other objects, for example to provide filtering operations. [Table 5-1](#) lists the signal processing objects on the mXT144U.

TABLE 5-1: SIGNAL PROCESSING OBJECTS

Object	Description
One-touch Gesture Processor T24 (PROCI_ONETOUCHGESTUREPROCESSOR_T24)	Operates on the data from a Touchscreen object. A One-touch Gesture Processor T24 converts touches into one-touch finger gestures (for example, taps, double taps and drags). See Section 5.2 “One-touch Gesture Processor T24 Object” .
Two-touch Gesture Processor T27 (PROCI_TWOTOUCHGESTUREPROCESSOR_T27)	Operates on the data from a One-touch Gesture Processor T24 object. A Two-touch Gesture Processor T27 converts touches into two-touch finger gestures (for example, pinches, stretches and rotates). See Section 5.3 “Two-touch Gesture Processor T27 Object” .
Grip Suppression T40 (PROCI_GRIPSUPPRESSION_T40)	Suppresses false detections caused, for example, by the user gripping the edge of the touchscreen. See Section 5.4 “Grip Suppression T40 Object” .
Touch Suppression T42 (PROCI_TOUCHSUPPRESSION_T42)	Suppresses false detections caused by unintentional large touches by the user. See Section 5.5 “Touch Suppression T42 Object” .
Shieldless T56 (PROCI_SHIELDLESS_T56)	Allows a sensor to use true single-layer co-planar construction. See Section 5.6 “Shieldless T56 Object” .
Lens Bending T65 (PROCI_LENSBENDING_T65)	Compensates for lens deformation (lens bending) by attempting to eliminate the disturbance signal from the reported deltas. See Section 5.7 “Lens Bending T65 Object” .
Noise Suppression T72 (PROCG_NOISESUPPRESSION_T72)	Performs various noise reduction techniques during touchscreen signal acquisition. See Section 5.8 “Noise Suppression T72 Object” .
Glove Detection T78 (PROCI_GLOVEDETECTION_T78)	Allows for the reporting of glove touches. See Section 5.9 “Glove Detection T78 Object” .
Retransmission Compensation T80 (PROCI_RETRANSMISSIONCOMPENSATION_T80)	Limits the negative effects on touch signals caused by poor device coupling to ground. See Section 5.10 “Retransmission Compensation T80 Object” .
Touch Sequence Processor T93 (PROCI_TOUCHSEQUENCELOGGER_T93)	Captures a sequence of touch and release locations to allow double taps to be detected. See Section 5.11 “Touch Sequence Processor T93 Object” .
Self Capacitance Noise Suppression T108 (PROCG_NOISESUPSELFAP_T108)	Suppresses the effects of external noise within the context of self capacitance touch measurements. See Section 5.12 “Self Capacitance Noise Suppression T108 Object” .
Symbol Gesture Processor T115 (PROCI_SYMBOLGESTURE_T115)	Detects arbitrary shaped gestures as a series of ordinal strokes. These are typically symbols drawn by the user for interpretation by the host as wake-up gestures or other application triggers. See Section 5.13 “Symbol Gesture Processor T115 Object” .
Sensor Correction T121 (PROCI_TSENSORCORRECTION_T121)	Allows adjustments to be made to mutual measurements from an associated touchscreen sensor. See Section 5.14 “Sensor Correction T121 Object” .

5.2 One-touch Gesture Processor T24 Object

5.2.1 INTRODUCTION

A One-touch Gesture Processor T24 object configures the on-chip gesture processing for one-touch gestures (such as taps, double taps, presses, flicks and drags).

The device supports the one-touch gestures listed in [Table 5-2](#).

TABLE 5-2: ONE-TOUCH GESTURES

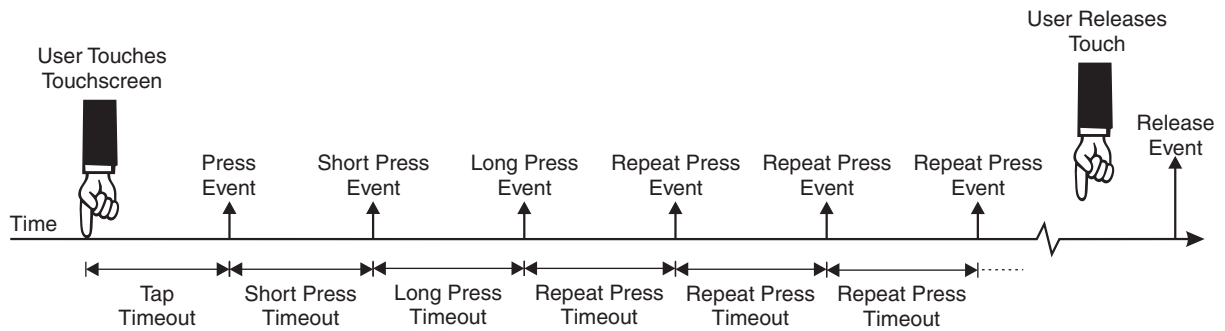
Gesture	Description
Press	A press occurs when the user touches and holds the touch surface. No significant movement takes place while the user's finger is on the touch surface. This could be used to select a number from a displayed numeric keypad. The same mechanism could be used to autorepeat the selected number if the user continues to press on the displayed number.
Release	A release occurs when the user removes their finger from the touch surface.
Tap	A tap occurs when the user quickly touches and releases the touchscreen. No significant movement takes place while the user's finger is on the touchscreen. It is characterized by a short touch duration. This could be used to activate a hyperlink on a displayed web page.
Double Tap	A double tap occurs when the user quickly touches and releases the touchscreen twice in quick succession. No significant movement takes place while the user's finger is on the touchscreen, or between successive touches. It is characterized by short touch durations, and a short gap between the first release and the second touch. This could be used to select a word in a displayed document.
Flick	A flick occurs when the user quickly touches the touchscreen, moves their finger a short distance across the surface and releases touch. It is characterized by a short touch duration. This could be used to display the next image in a sequence of images.
Throw	A throw occurs when the user touches the touchscreen, moves their finger across the surface, and releases the touch in a quick flick-type motion. Conceptually, it can be considered like a drag motion followed by a flick. It is characterized by a large movement across the touchscreen and a rapid acceleration of the touch just before the release.
Drag	A drag occurs when the user touches the touchscreen, moves their finger across the surface, and releases touch. It is characterized by a large movement across the touchscreen. Depending on the application, multiple drag events may be generated as the user moves their finger. This could be used to select a sentence in a displayed document.
Tap-and-press	A tap-and-press occurs when the user generates both a tap gesture and a press gesture in quick succession.
Tap-and-touch	A tap-and-touch occurs when the user generates a tap and then places their finger back on the screen in quick succession. The event is generated as soon as the second touch-down is validated. This gesture is useful for implementing a "knock-on" feature which allows the user's product to be woken up by a tap-and-touch gesture on the touchscreen.

The host can combine one-touch gestures to create two-touch and multitouch gestures, either within the host operating system or in the driver code. For example, two simultaneous tap gestures on the touchscreen can be combined to form a two-touch tap. The mXT144U also provides on-chip touchscreen gesture processing for three specific two-touch gestures: pinch, rotate and stretch (see the Two-touch Gesture Processor T27 in [Section 5.3 "Two-touch Gesture Processor T27 Object"](#)).

5.2.1.1 Press and Tap Gestures

Press and Tap gestures are created from a series of press events. These are generated when the touchscreen is touched, the touch is held, and then the touch is released. [Figure 5-1](#) shows the most general case.

FIGURE 5-1: SEQUENCE OF PRESS EVENTS



In this case:

1. The user touches the touchscreen.
2. If the touch is held for longer than the Tap Timeout, a Press event is generated.
3. If the user continues to hold the touch, and the Short Press Timeout is exceeded, a short press event is generated.
4. If the user continues to hold the touch, and the Long Press Timeout is exceeded, a long press event is generated.
5. If the user continues to hold the touch, and the Repeat Press Timeout is exceeded, a repeat press event is generated.
6. If the user continues to hold the touch, further repeat press events are generated at regular intervals whenever the Repeat Press Timeout is exceeded.
7. When the user releases the touchscreen, a release event is generated.

Note that in [Figure 5-1](#) all three press events are enabled. The One-touch Gesture Processor T24 object, however, allows you to enable or disable the generation of the short, long and repeat press events, as required (see ["PROCESS Field"](#)). All touches always generate release events.

5.2.1.2 Double Tap, Tap-and-Touch and Tap-and-Press Gestures

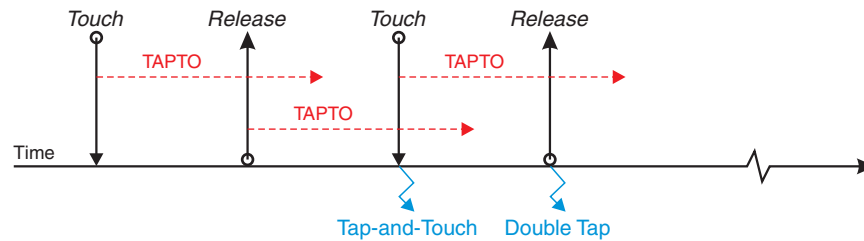
Double Tap, Tap-and-Touch and Tap-and-Press Gestures are created from two successive touch events. The second touch is reported as a Double Tap, Tap-and-Touch or Tap-and-Press event depending on the timing of the second touch. The generation of all three gestures is enabled by the PROCESS DBLTAPEN control (see ["PROCESS Field"](#)).

[Figure 5-2](#) shows how the three events are distinguished from each other:

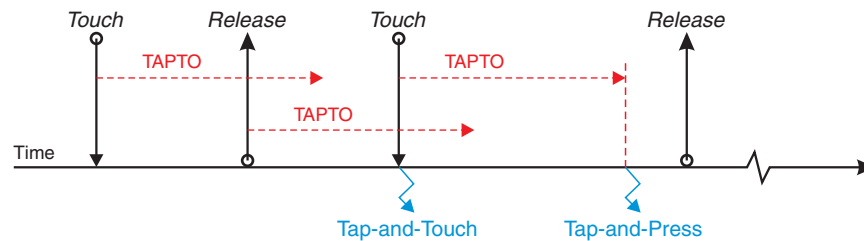
- A Tap-and-touch event is generated at the point of touchdown, if a second touch event occurs before TAPTO expires after a valid Tap. Note that this occurs whether or not the Tap event is enabled.
- A Double Tap event is generated only if a second valid Tap occurs. In this case the second touchdown must be before TAPTO expires after the first touch release, and the second release must occur before TAPTO expires after the second touchdown.
- A Tap-and-press is generated only if there is a second touchdown before TAPTO expires after the first release, and the second press is held for longer than TAPTO before being released. The Tap-and-press event is generated in this case at the point when the TAPTO timer expires.

FIGURE 5-2: SEQUENCE OF EVENTS WITH TAP-AND-TOUCH EVENT

Tap-and-Touch with Double Tap



Tap-and-Touch with Tap-and-Press



5.2.2 CONFIGURATION

TABLE 5-3: CONFIGURATION FOR ONE-TOUCH GESTURE PROCESSOR T24
(PROCI_ONETOUCHGESTUREPROCESSOR_T24)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved			DISTSF	DISRPTTAPOTH	DISRPTTAPMOV	RPTEN	ENABLE
1	NUMGEST	Maximum number of reported gestures							
2	GESTEN	LPRESS	SPRESS	DRAG	FLICK	DBLTAP	TAP	RELEASE	PRESS
3		Reserved					TAPTCH	THROW	RPRESS
4	PROCESS	Reserved		THROWEN	FLICKEN	DBLTAPEN	REPEN	LONGEN	SHORTEN
5	TAPTO	Reserved	Tap Timeout						
6	FLICKTO	Reserved	Flick Timeout						
7	DRAGTO	Reserved	Drag Timeout						
8	SPRESSTO	Reserved	Short Press Timeout						
9	LPRESSTO	Reserved	Long Press Timeout						
10	REPPRESSTO	Reserved	Repeat Press Timeout						
11	FLICKTHR	Flick Threshold LSByte							
12		Flick Threshold MSByte							
13	DRAGTHR	Drag Threshold LSByte							
14		Drag Threshold MSByte							
15	TAPTHR	Tap Threshold LSByte							
16		Tap Threshold MSByte							
17	THROWTHR	Throw Threshold LSByte							
18		Throw Threshold MSByte							

CTRL Field

ENABLE: Enables the use of this One-touch Gesture Processor T24 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

DISRPTTAPMOV: Suppresses the reporting of moving taps. If this bit is set to 1, reporting of all tap related (tap, tap-and-press, tap-and-touch, double tap) gesture events is disabled if the touch is moving. This check is performed on each gesture independently.

DISRPTTAPOTH: Suppresses the reporting of Tap related gestures. If this bit is set to 1, reporting of all tap related (tap, tap-and-press, tap-and-touch, double tap) gesture events is disabled if there is a touch down/up event registered against any other touch that does not have gesture processing enabled on it via the NUMGEST field.

NOTE This feature should be used only when there is only one gesture process enabled via the NUMGEST configuration field (NUMGEST = 1).
--

DISTSF: Specifies the Distance Scaling Factor for the reported DIST value in the One-touch Gesture Processor T24 message. If this bit is set to 0, no downscaling is applied to the reported DIST value. If this bit is set to 1, the DIST message field is scaled to half its value before being output.

NUMGEST Field

This field specifies the maximum number of touches that are to be reported. For example, if this field is set to 1, only the first touch is reported as a gesture and all other touches do not generate messages.

This field does not remove report IDs. Report IDs are fixed to the maximum number of touch gestures.

Range: 1 to 4

GESTEN Field

Enables one-touch gestures and reporting.

PRESS: Press event reporting is enabled if set to 1, and disabled if set to 0.

RELEASE: Release event reporting is enabled if set to 1, and disabled if set to 0.

TAP: Tap event reporting is enabled if set to 1, and disabled if set to 0.

DBLTAP: Double Tap event reporting is enabled if set to 1, and disabled if set to 0.

FLICK: Flick event reporting is enabled if set to 1, and disabled if set to 0.

DRAG: Drag event reporting is enabled if set to 1, and disabled if set to 0.

SPRESS: Short Press event reporting is enabled if set to 1, and disabled if set to 0.

LPRESS: Long Press event reporting is enabled if set to 1, and disabled if set to 0.

RPRESS: Repeat Press event reporting is enabled if set to 1, and disabled if set to 0.

THROW: Throw event reporting is enabled if set to 1, and disabled if set to 0.

TAPTCH: Tap-and-touch event reporting is enabled if set to 1, and disabled if set to 0.

PROCESS Field

This field enables and disables the press events and some of the one-touch gestures that can be processed by the device.

SHORTEN: Enables short presses to be processed by the device. Short presses are enabled if set to 1, and disabled if set to 0.

LONGEN: Enables long presses to be processed by the device. Long presses are enabled if set to 1, and disabled if set to 0.

REPEN: Enables repeat presses to be processed by the device. Repeat presses are enabled if set to 1, and disabled if set to 0.

DBLTAPEN: Enables double tap, tap-and-touch and tap-and-press gestures to be processed by the device. Double taps are enabled if set to 1, and disabled if set to 0.

FLICKEN: Enables flicks to be processed by the device. Flicks presses are enabled if set to 1, and disabled if set to 0.

THROWEN: Enables throws to be processed by the device. Throws are enabled if set to 1, and disabled if set to 0.

TAPTO Field

This field specifies the timeout for a tap gesture. This is the maximum duration of a tap. A tap gesture is when the user touches and releases a touch, without moving their finger. If this sequence takes less time than the tap timeout, a tap event is generated. The timeout is specified in units of 4 ms. A value of zero means 75 (300 ms). The timeout should not be set shorter than the active cycle period.

This value also controls the generation of double tap events. A double tap event consists of a press, release, press, and final release. If the time between each of these actions is less than the tap timeout, a double tap event is generated.

The user must move their finger less than the Tap Threshold (TAPTHR) during a tap to generate a tap or double tap gesture. Also, the distance between consecutive taps must be less than the Tap Threshold to generate a double tap gesture.

If the timeout expires and the touch is still present, the press processing commences.

Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

FLICKTO Field

This field specifies the timeout for a flick gesture. This is the maximum duration of a flick gesture. A flick gesture is when the user presses the touchscreen, moves their finger, and then releases the touchscreen. A flick event is generated if a touch moves more than the flick threshold (defined by the FLICKTHR field), in less time than the tap timeout (TAPTO), and is then released within the FLICKTO. The flick timeout should not be set shorter than the active cycle period.

The flick timeout is specified in units of 4 ms. A value of zero means 75 (300 ms).

See also [“FLICKTHR and TAPTHR Fields”](#).

Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

DRAGTO Field

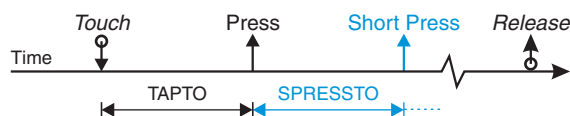
This field specifies the timeout for detecting a new drag event. This is the maximum time between two drag events. A drag gesture is when the user presses the touchscreen and moves their finger. This generates a drag event. If the user moves their finger again within the drag timeout period, another drag event is generated. If the user's finger remains stationary for longer than the drag timeout period, drag processing stops. In this case, any enabled press events start to be generated. The drag timeout is specified in units of 4 ms. A value of zero means 75 (300 ms). The timeout should not be set shorter than the active cycle period. See also [“DRAGTHR Field”](#).

Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

SPRESSTO Field

This field specifies the Short Press Timeout. This is the time between the start of press processing and the generation of a short press event (see [Figure 5-3](#)). The Short Press Timeout is used only if short press processing is enabled using the “SHORTEN” bit (see [“PROCESS Field”](#)).

FIGURE 5-3: SHORT PRESS TIMEOUT

SHORTEN Set

The short timeout is specified in units of 4 ms. A value of zero means 75 (300 ms). The timeout should not be set shorter than the active cycle period.

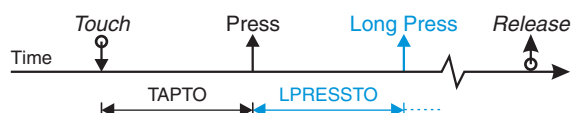
Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

LPRESSTO Field

This field specifies the Long Press Timeout. This is the time taken for the generation of a long press event. The start point for the timeout depends on whether or not short events are also enabled. If short presses are enabled, the timeout is from the short press event; otherwise it is from the start of press processing (see [Figure 5-4](#)). The Long Press Timeout is used only if long press processing is enabled using the “LONGEN” bit (see “[PROCESS Field](#)”).

FIGURE 5-4: LONG PRESS TIMEOUT

LONGEN Only Set



LONGEN and SHORTEN Set



The long timeout is specified in units of 4 ms. A value of zero means 75 (300 ms). The timeout should not be set shorter than the active cycle period.

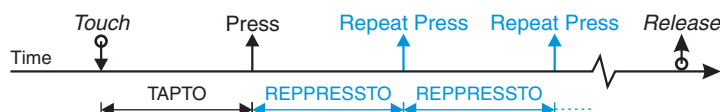
Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

REPPRESSTO Field

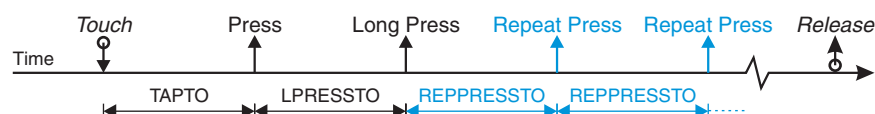
This field specifies the Repeat Press Timeout. This is the time taken for the generation of a repeat press event. The start point for the timeout depends on whether short press and long press events are also enabled (see [Figure 5-5](#)). The repeat timeout also specifies the time between the generation of consecutive repeat press events. The Repeat Press Timeout is used only if repeat press processing is enabled using the “REPEN” bit (see “[PROCESS Field](#)”).

FIGURE 5-5: REPEAT PRESS TIMEOUT

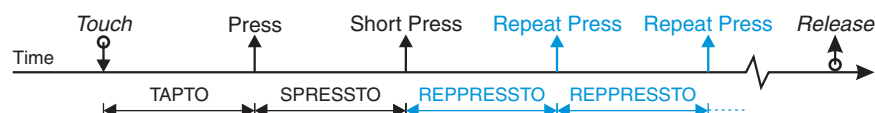
REPEN Only Set



REPEN and LONGEN Set



REPEN and SHORTEN Set



REPEN, LONGEN and SHORTEN Set



The repeat timeout is specified in units of 4 ms. A value of zero means 75 (300 ms). The timeout should not be set shorter than the active cycle period.

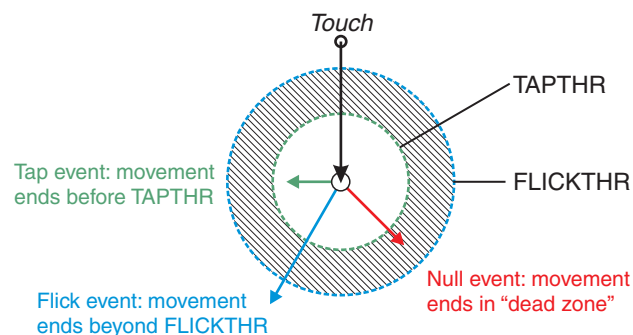
Range: 0 (75 = 300 ms), 1 to 127 (in units of 4 ms)

FLICKTHR and TAPTHR Fields

These fields dictate how a quick touch-and-release action is to be interpreted. This action can be interpreted as either a tap or a flick gesture. These two gestures are essentially the same. The only difference is the amount of movement that occurs between the touch and the release. With a flick, the user's finger moves a significant distance before the touch is release. With a tap there is minimal or no movement.

If the movement is less than the Tap Threshold (TAPTHR), the gesture is interpreted as a tap. If the movement is greater than or equal to the Flick Threshold (FLICKTHR), the gesture is interpreted as a flick. If the movement is in the “dead zone” between the two thresholds, no gesture is generated for the touch. This is summarized in Figure 5-6.

FIGURE 5-6: FLICK AND TAP THRESHOLDS



The thresholds are specified in units of touchscreen distance.

It may be useful to set both the TAPTHR and FLICKTHR values to the same values initially. This means there is no “dead zone” and so either a tap or flick gesture is always reported. If the design needs to be fine tuned, the two controls can later be set to different values to create a “dead zone”. Note that, in this case, FLICKTHR must be greater than or equal to TAPTHR.

See also “[TAPTO Field](#)” and “[FLICKTO Field](#)”.

DRAGTHR Field

This value specifies how far the user must move their finger across the touchscreen to generate a drag event. See also “[DRAGTO Field](#)”. The drag threshold is specified in units of touchscreen distance.

THROWTHR Field

This field specifies the speed on the drag or release from the drag that is needed to generate a throw gesture. The speed of the user's finger must be greater than or equal to THROWTHR for a throw gesture to be generated. The speed is specified as:

$$2 \times \text{touchscreen_distance_per_ms}$$

A minimum time of 4 ms for the release is assumed. If the release takes less than 4 ms, the One-touch Gesture Processor T24 object uses a time of 4 ms in its calculations.

5.2.3 MESSAGES

The message data for a One-touch Gesture Processor T24 object is shown in [Table 5-4](#).

**TABLE 5-4: MESSAGE DATA FOR ONE-TOUCH GESTURE PROCESSOR T24
(PROCI_ONETOUCHGESTUREPROCESSOR_T24)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved				Event			
2	XPOSMSB	X position bits 4 to 11							
3	YPOSMSB	Y position bits 4 to 11							
4	XYPOSLSB	X position bits 0 to 3				Y position bits 0 to 3			
5	DIR	Gesture direction							
6	DIST	Gesture distance LSByte							
7		Gesture distance MSByte							
8	XYPOSMSB	X position bits 12 to 15				Y position bits 12 to 15			

STATUS Field

This field reports which single-touch event has occurred. [Table 5-5](#) gives the possible values and their meanings.

TABLE 5-5: EVENTS

Event	Description
0	Reserved
1	Press
2	Release
3	Tap
4	Double-tap
5	Flick
6	Drag
7	Short Press
8	Long Press
9	Repeat Press
10	Tap-and-press
11	Throw
12	Tap-and-touch
13 to 15	Reserved

XPOSMSB, YPOSMSB, XYPOSLSB and XYPOSMSB Fields

These fields report the X and Y position of the gesture. XYPOSLSB contains bits 0 to 3 of the position, XPOSMSB/YPOSMSB contains bits 4 to 11 of the position and XYPOSMSB contains bits 12 to 15 of the position. Thus, the position is constructed as follows:

$$X \text{ position} = ((XYPOSMSB \& 0xF0) \ll 8) | (XPOSMSB \ll 4) | ((XYPOSLSB \& 0xF0) \gg 4)$$

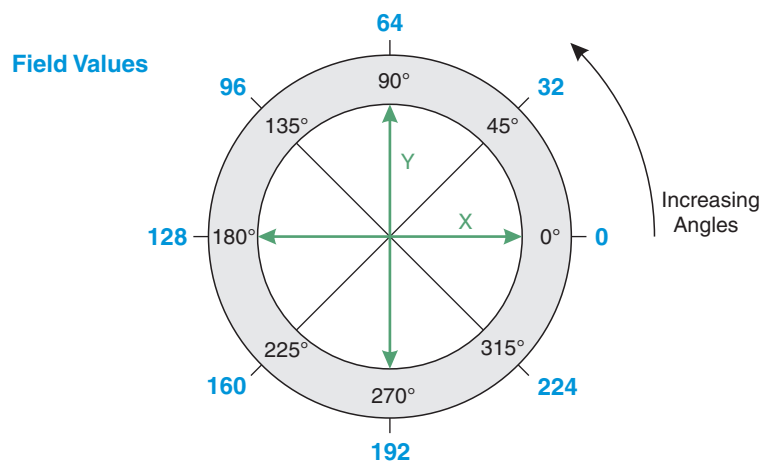
$$Y \text{ position} = ((XYPOSMSB \& 0x0F) \ll 12) | (YPOSMSB \ll 4) | (XYPOSLSB \& 0x0F)$$

The position is reported in 16-bit format within the range specified by the Multiple Touch Touchscreen T100 XRANGE and YRANGE settings.

DIR Field

The DIR field contains the direction of the flick or throw when a flick or throw event is generated. The direction is specified in 1/256 of 360°, as in [Figure 5-7](#). Note that if CTRL DISTSF is set to 1, the direction will be scaled down by half. In other words, the reported distance is equal to $DIST \times 2^{DISTSF}$.

FIGURE 5-7: DIRECTION AND ANGLE VALUES



Note: The X and Y positions are before any processing of the Multiple Touch Touchscreen T100 ORIENT field.

DIST Field

The DIST field contains the distance of the flick or throw when a flick or throw event is generated.

5.3 Two-touch Gesture Processor T27 Object

5.3.1 INTRODUCTION

NOTE This object operates only if the linked One-touch Gesture Processor T24 object has also been set up correctly and enabled for use (see [Section 5.2 “One-touch Gesture Processor T24 Object”](#)).

A Two-touch Gesture Processor T27 object configures the on-chip touchscreen gesture processing for the two-touch gestures: pinch, rotate and stretch (see [Table 5-6](#)). Other two-touch gestures can be generated by the host from one-touch gestures (see [Section 5.2.1 “Introduction”](#)).

TABLE 5-6: TWO-TOUCH GESTURES

Gesture	Description
Pinch	A pinch occurs when the user touches and holds the touch surface with two fingers and brings them together. It is not necessary for both fingers to move. One finger can remain stationary while the other moves towards it. It is the relative (decreasing) distance between the fingers that determines the pinch gesture.
Rotate	A rotate occurs when the user touches and holds the touch surface with two fingers and then moves them in a circle around each other. It is not necessary for both fingers to move. One finger can remain stationary while the other moves around it. It is the relative angle between the fingers that determines the rotate gesture.
Stretch	A stretch is the inverse of a pinch. The user touches and holds the touch surface with two fingers and moves them apart. It is not necessary for both fingers to move. One finger can remain stationary while the other moves away from it. It is the relative (increasing) distance between the fingers that determines the pinch gesture.

The two-touch gestures in [Table 5-6](#) can be combined. For example, combining a rotate gesture with a stretch or pinch gesture will form an increasing or decreasing spiral gesture.

5.3.2 CONFIGURATION

**TABLE 5-7: CONFIGURATION FOR TWO-TOUCH GESTURE PROCESSOR T27
(PROCI_TWOTOUCHGESTUREPROCESSOR_T27)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved			DISTSF	Reserved		RPTEN	ENABLE
1	NUMGEST	Maximum number of reported gestures							
2	Reserved	Reserved							
3	GESTEN	STRETCH	ROTATE	PINCH	Reserved				
4	ROTATETHR	Reserved	Rotate Threshold						
5	ZOOMTHR	Zoom Threshold LSByte							
6		Zoom Threshold MSByte							

CTRL Field

ENABLE: Enables the use of this Two-touch Gesture Processor T27 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows this object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

DISTSF: Specifies the Distance Scaling Factor for the reported SEP value in the Two-touch Gesture Processor T27 message. If this bit is set to 0, no downscaling is applied to the reported SEP value. If this bit is set to 1, the SEP message field is scaled to half its value before being output.

NUMGEST Field

This field specifies the maximum number of two-touch gestures to be reported. Note that each gesture is made up of a pair of touches so the number of gestures specified represents half the number of touches. For example, if this field is set to 1, the first and second touch are paired as a gesture and are reported; all other touches are ignored by the processor.

Note that this field does not remove report IDs. Report IDs are fixed to the maximum number of touch gestures.

Range: 1

GESTEN Field

This field enables two-touch gestures and reporting.

PINCH: Pinch event reporting is enabled if set to 1, and disabled if set to 0.

ROTATE: Rotate event reporting is enabled if set to 1, and disabled if set to 0.

STRETCH: Stretch event reporting is enabled if set to 1, and disabled if set to 0.

ROTATETHR Field

This field specifies the minimum change in angle between two points for generating two-touch rotate gestures. The rotate threshold is specified in 1/256 of 360° (see “[DIR Field](#)” for details) with a maximum allowed angle of less than 180°. The reported angle is affected by the order in which the touches are applied.

Range: 0 to 127

ZOOMTHR Field

This field specifies the minimum change in distance (in units of touchscreen distance) required between two simultaneous touches to generate a stretch or pinch gesture.

5.3.3 MESSAGES

The message data for a Two-touch Gesture Processor T27 object is shown in [Table 5-8](#).

**TABLE 5-8: MESSAGE DATA FOR TWO-TOUCH GESTURE PROCESSOR T27
(PROCI_TWOTOUCHGESTUREPROCESSOR_T27)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	STRETCH	ROTATE	PINCH	ROTATEDIR	Reserved			
2	XPOSMSB	Center of gesture X position bits 4 to 11							
3	YPOSMSB	Center of gesture Y position bits 4 to 11							
4	XYPOSLSB	X position l bits 0 to 3				Yposition bits 0 to 3			
5	ANGLE	Gesture angle							
6	SEP	Gesture separation LSByte							
7		Gesture separation MSByte							
8	XYPOSMSB	X position bits 12 to 15				Y position bits 12 to 15			

STATUS Field

This field reports which two-touch events have occurred. A two-touch gesture may consist of more than one two-touch gestures. For example, a rotate may be combined with a stretch to create a spiraling zoom effect.

ROTATEDIR: Indicates the direction of the rotate event: 0 = rotation in the direction of increasing angles, 1 = rotation in the direction of decreasing angles (see “[DIR Field](#)” for details). This bit should be ignored if the ROTATE bit is not set.

PINCH: If set to 1, a pinch event has occurred.

ROTATE: If set to 1, a rotate event has occurred.

STRETCH: If set to 1, a stretch event has occurred.

XPOSMSB, YPOSMSB, XYPOSLSB and XYPOSMSB Fields

These fields report the X and Y position of the center of the gesture. XYPOSLSB contains bits 0 to 3 of the position, XPOSMSB/YPOSMSB contains bits 4 to 11 of the position and XYPOSMSB contains bits 12 to 15 of the position. Thus, the position is constructed as follows:

X position = $((XYPOSMSB \& 0xF0) \ll 8) \mid (XPOSMSB \ll 4) \mid ((XYPOSLSB \& 0xF0) \gg 4)$

Y position = $((XYPOSMSB \& 0x0F) \ll 12) \mid (YPOSMSB \ll 4) \mid (XYPOSLSB \& 0x0F)$

The position is reported in 16-bit format within the range specified by the Multiple Touch Touchscreen T100 X RANGE and Y RANGE settings.

ANGLE Field

The ANGLE field reports the angle between two touches in a two-touch gesture. The angle between the two touches, and its direction, is determined by the relative X and Y positions of the second touch from the first touch. The angle is reported in 1/256 of 360° (see ["DIR Field"](#) for details).

SEP Field

The SEP field reports the separation (distance) between the two touches in a two-touch gesture as a 16-bit number. Note that if CTRL DISTSF is set to 1, the separation will be scaled down by half. In other words, the reported separation is equal to $SEP \times 2^{\text{DISTSF}}$.

This value could be used by the host, for example, as an indication of the speed of a rotate or stretch gesture.

5.4 Grip Suppression T40 Object

5.4.1 INTRODUCTION

The Grip Suppression T40 object suppresses false detections around the edge of the touchscreen, such as those caused by the user gripping the touchscreen on a handheld device.

The Grip Suppression T40 object processes the measurement data received from the Multiple Touch Touchscreen T100 object.

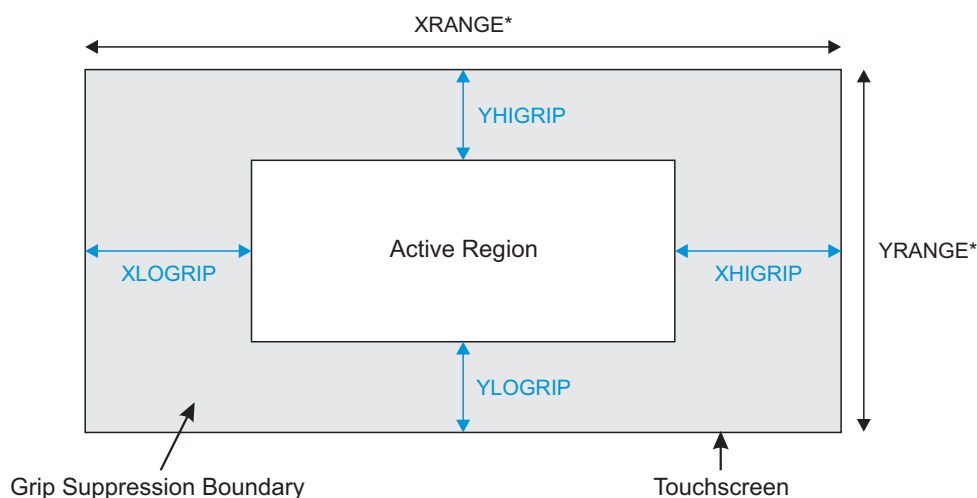
The precise nature of the grip suppression is described in the following sections.

5.4.1.1 Grip Suppression Boundary

The grip suppression mechanism defines a grip suppression boundary around the edge of the Multiple Touch Touchscreen T100 within which grip suppression may occur.

Figure 5-8 shows the settings that define the grip suppression boundary.

FIGURE 5-8: GRIP SUPPRESSION BOUNDARY FIELDS



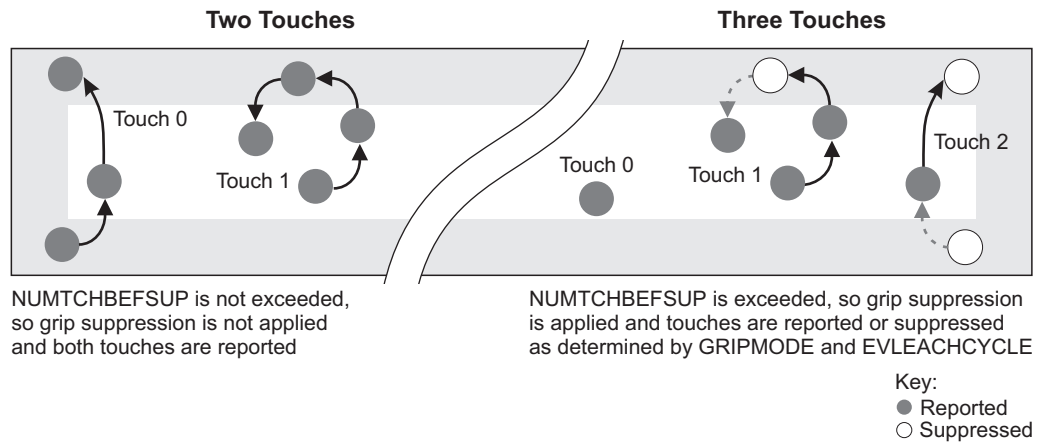
* XRANGE and YRANGE fields in a Multiple Touch Touchscreen T100

5.4.1.2 Number of Touches

Grip suppression takes place only when more than a certain number of touches are present, as specified by the NUMTCHBEFSUP field. When the number of touches on the sensor is less than or equal to the value of NUMTCHBEFSUP, grip suppression is not performed. When more than NUMTCHBEFSUP touches present, then all touches will be tested to see if they should be suppressed or not according to the grip mode and evaluation mode settings.

For example, Figure 5-9 shows the effect of specifying NUMTCHBEFSUP = 2 (that is, grip suppression is applied if three or more touches are present on the screen).

FIGURE 5-9: EXAMPLE – NUMTCHBEFSUP = 2



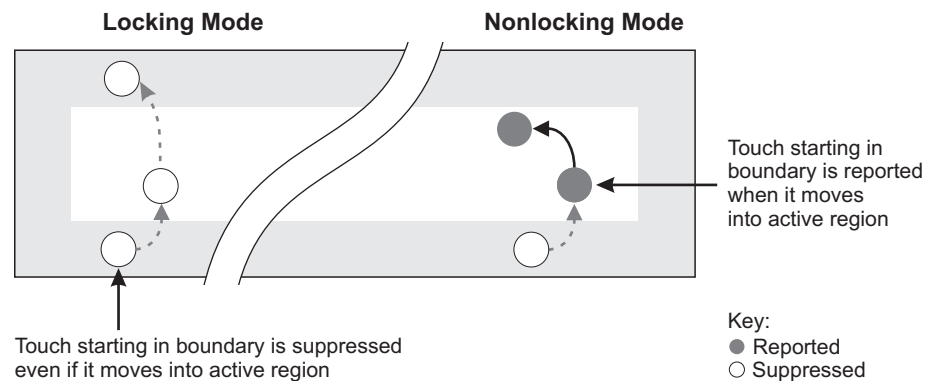
Note that the default value of zero effectively means that grip suppression will be evaluated as soon as a touch is placed on the screen (that is, at least one touch must be present for grip suppression to be performed).

5.4.1.3 Grip Mode

The grip mode (as defined by the GRIPMODE bit in the CTRL field) defines what happens to touches moving from the grip suppression boundary to the active region. Specifically, it defines whether a suppressed touch is locked (that is, remains suppressed) or becomes reported (see [Section 5.4.1.3 “Grip Mode”](#)):

- In locking mode, once a touch in the grip suppression boundary has been suppressed, it remains suppressed even if the touch subsequently moves into the active region.
- In non-locking mode, a touch that has been suppressed while in the grip suppression boundary becomes reported if it moves into the active region.

FIGURE 5-10: GRIP MODE



5.4.1.4 Evaluation Mode

The EVLEACHCYCLE control in the CTRL field determines when the touch is evaluated so that it can be suppressed or reported, as appropriate, as the touch is moved.

Touches can be evaluated for grip suppression as follows:

- Evaluated per touchdown – a touch is evaluated for grip suppression at the start of the touch (that is, on touchdown).
- Evaluated per cycle – a touch is evaluated for grip suppression on every cycle. This means that the state of the touch can change cycle by cycle during the touch depending on its position.

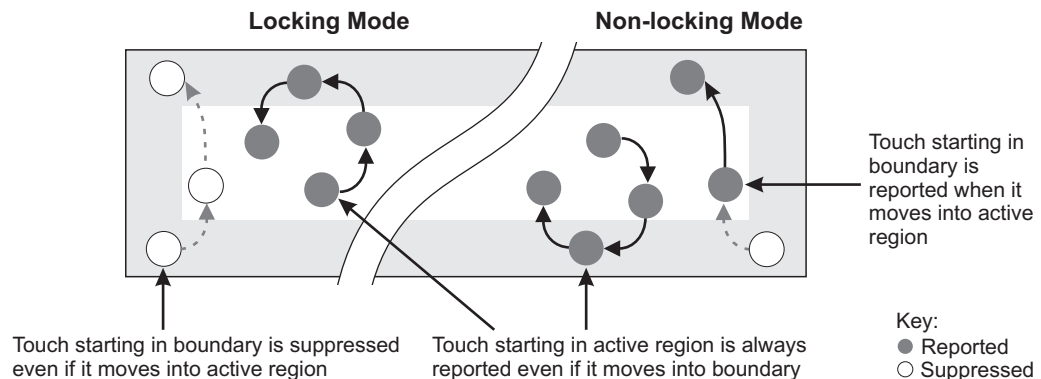
Between them, the EVLEACHCYCLE control and the GRIPMODE control (see [Section 5.4.1.3 “Grip Mode”](#)) create four combinations of behavior. Roughly speaking, GRIPMODE controls whether to report (unsuppress) a touch and EVLEACHCYCLE controls whether to suppress a touch.

If per touchdown evaluation is chosen (EVLEACHCYCLE = 0):

- In locking grip mode (GRIPMODE = 0), a touch that starts inside the grip suppression boundary is suppressed, and remains suppressed even if it subsequently moves into the active region.
- In non-locking grip mode (GRIPMODE = 1), a touch that starts in the grip suppression boundary is initially suppressed but becomes reported when it moves into the active region.
- In both grip modes, touches that start in the active region are always reported.

See [Figure 5-11](#) for a further explanation.

FIGURE 5-11: PER TOUCHDOWN EVALUATION

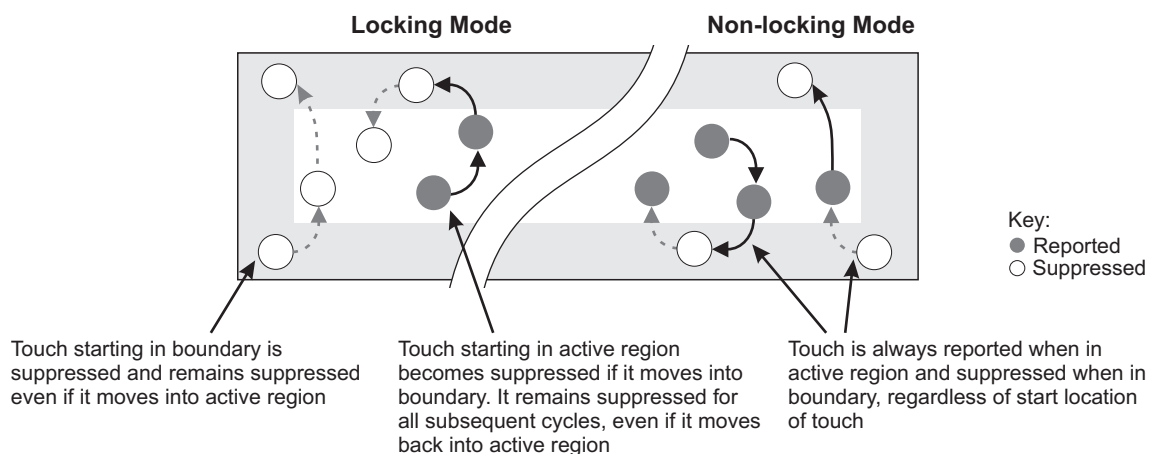


If per cycle evaluation is chosen (EVLEACHCYCLE = 1):

- In locking grip mode (GRIPMODE = 0), if a touch starts in the grip suppression boundary, or moves into the grip suppression boundary, it becomes suppressed and remains suppressed even if it subsequently moves into the active region.
- In non-locking grip mode (GRIPMODE = 1), the touch is suppressed when in the grip suppression boundary and reported when in the active region. Thus, in non-locking mode, the state of the touch will change cycle by cycle as it moves between the active region and the grip suppression boundary, regardless of where the touch started.

See [Figure 5-12](#) for a further explanation.

FIGURE 5-12: PER CYCLE EVALUATION



5.4.2 CONFIGURATION

**TABLE 5-9: CONFIGURATION FOR GRIP SUPPRESSION T40
(PROCI_GRIPSUPPRESSION_T40)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved		EVLEACHCYCLE	GRIPMODE	Reserved			ENABLE
1	XLOGRIP	Grip suppression X low boundary							
2	XHIGRIP	Grip suppression X high boundary							

**TABLE 5-9: CONFIGURATION FOR GRIP SUPPRESSION T40
(PROCI_GRIPSUPPRESSION_T40) (CONTINUED)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
3	YLOGRIP	Grip suppression Y low boundary							
4	YHIGRIP	Grip suppression Y high boundary							
5	Reserved	Reserved							
6	NUMTCHBEFSUP	Reserved		NUMTCHBEFSUP					

CTRL Field

This field enables or disables grip suppression.

ENABLE: Enables the use of the Grip Suppression T40 object. If this bit is set to 1, the object is enabled and grip suppression takes place when appropriate. If this bit is set to 0, the object is disabled and grip suppression never occurs.

GRIPMODE: Sets the grip mode. Set this bit to 1 for non-locking mode, and 0 for locking mode. See [Section 5.4.1.3 “Grip Mode”](#) for more details.

EVLEACHCYCLE: Sets the evaluation mode. If this bit is set to 1, the system evaluates whether to suppress or report each touch within the grip boundary on every cycle. If this bit is set to 0, the system evaluates whether to suppress or report each touch within the grip boundary on touchdown and thereafter, as the touch moves, the subsequent behavior is determined by the grip mode alone. See [Section 5.4.1.4 “Evaluation Mode”](#) for more details.

Note that, in both cases, the current number of touches on the screen must exceed NUMTCHBEFSUP for grip suppression to be performed.

XLOGRIP, XHIGRIP, YLOGRIP and YHIGRIP Fields

These fields specify a grip suppression boundary around the edge of the Multiple Touch Touchscreen T100 that this object processes. Inside this boundary is the active region of the Touchscreen. XLOGRIP and XHIGRIP define the width of the boundaries on the low and high X edges respectively and YLOGRIP and YHIGRIP define the width of the boundaries on the low and high Y edges respectively. See [Figure 5-8](#) for more information on how the controls define the grip suppression boundary.

NOTE	The boundary settings are applied before the Multiple Touch Touchscreen T100 orientation controls (see “CFG1 Field”).
-------------	--

TXLOGRIP/XHIGRIP Range: 0 to 255

YLOGRIP/YHIGRIP Range: 0 to 255

NUMTCHBEFSUP Field

NUMTCHBEFSUP: Specifies the Number of Touches Before Grip Suppression is activated. When the number of touches on the sensor is less than or equal to the value of NUMTCHBEFSUP, grip suppression is not performed. When the total number of touches on the screen exceeds NUMTCHBEFSUP, then the touches are evaluated to see if they should be reported or suppressed, according to the GRIPMODE and EVLEACHCYCLE controls in the CTRL field (see [“CTRL Field”](#)).

Note that the default value of zero means that at least one touch must be present for grip suppression to be performed (that is, grip suppression will be evaluated as soon as a touch is placed on the screen). Similarly, setting the value to 5 (maximum number of reported touches) or above, results in grip suppression never been applied.

See [Section 5.4.1.2 “Number of Touches”](#) for more details.

NUMTCHBEFSUP Range: 0 to 4 (number of touches to be exceeded),
5 to 63 (grip suppression never occurs)

5.4.3 MESSAGES

The Grip Suppression T40 object does not send messages directly. Instead the Multiple Touch Touchscreen T100 reports changes in the suppression state in its touch status message for a touch (see [Section 4.3.4 "Messages"](#) for an explanation of the SUP and UNSUP events).

The linked Multiple Touch Touchscreen T100 object reports an UNSUP event in the touch status message for a touch in the following circumstance:

- In non-locking mode, if the touch becomes unsuppressed (reported) after having previously been suppressed.

The linked Multiple Touch Touchscreen T100 object reports a SUP event in the touch status message for a touch in the following circumstances:

- If per cycle evaluation is chosen (EVLEACHCYCLE = 1), when the touch that moves from the active region into the grip suppression boundary
- If per cycle evaluation is chosen (EVLEACHCYCLE = 1), if the touch that is in the grip suppression boundary when NUMTCHBEFSUP becomes satisfied (that is, another touch is put down)

5.5 Touch Suppression T42 Object

5.5.1 INTRODUCTION

A Touch Suppression T42 object processes the measurement data received from a particular instance of a Multiple Touch Touchscreen T100 object. It suppresses false detections from unintentional touches.

The Touch Suppression T42 object provides the following types of touch suppression:

- **Large Object Touch Suppression** – Detects unintentional touches from a large body area, such as from a face, ear or palm. If an unintentional touch is detected, and both distance touch suppression and the reporting of suppressed large object touches are not in use, the entire Multiple Touch Touchscreen is suppressed. Otherwise, the large touch is suppressed, with further suppression determined by the behavior of the additional distance touch suppression. The suppression is released only when all the touches are released.
- **Maximum Touch Suppression** – Suppresses all touches if more than a specified number of touches has been detected. Any touches that are already detected will also be suppressed. The touches remain suppressed until all the touches are released or a recalibration occurs.
- **Distance Touch Suppression** – Operates in conjunction with large object suppression to suppress false touches if they are less than a specified distance from a suppressed touch. Any touches greater than this distance from a suppressed touch remain unsuppressed. Large object touch suppression needs to be enabled for distance touch suppression to operate.
- **Maximum Screen Area Suppression** – Triggers a full screen suppression when at least a specified number of nodes are in detect on the touchscreen. Note that number of nodes in detect are not necessarily part of a single touch.
- **Edge Touch Suppression** – Operates in conjunction with large object suppression to suppress false touches if they are on the same edge as a suppressed large object. Any touches not covering the edge sensor line that a suppressed touch also covers, remain unsuppressed. Note that large object touch suppression also needs to be enabled for edge touch suppression to operate.

Note that large object touch suppression (without distance touch suppression or suppressed large object touches reporting), and maximum touch suppression and maximum screen area suppression all lock the Touchscreen and with it any Adjacent Key Suppression locks that are currently applied.

5.5.2 CONFIGURATION

**TABLE 5-10: CONFIGURATION FOR TOUCH SUPPRESSION T42
(PROCI_TOUCHSUPPRESSION_T42)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved	EDGESUP	SUPDISTEN	DISTLOCK	DISLOBJ	SHAPEEN	Reserved	ENABLE
1	Reserved	Reserved							
2	MAXAPPRAREA	Maximum approach area threshold							
3	MAXTCHAREA	Maximum touch area threshold							
4	SUPSTRENGTH	Suppression aggressiveness							
5	SUPEXTTO	Suppression extension timeout							
6	MAXNUMTCHS	Maximum touches							
7	SHAPESTRENGTH	Shaped-based aggressiveness							
8	SUPDIST	Suppression distance							
9	DISTHYST	Suppression distance hysteresis							
10	MAXSCRNAREA	Maximum screen area							
11	CFG	Reserved					SUPTCHRPTEN	RELAXDIAGSUP	RELAXCLOSESUP
12	Reserved	Reserved							
13	EDGSUPSTR	Edge suppression strength							

CTRL Field

This field enables or disables large touch suppression.

ENABLE: Enables the use of this Touch Suppression T42 object. The object is enabled if set to 1, and disabled if set to 0.

SHAPEEN: Enables or disables ear suppression. This is a second level of rejection logic that is based upon the touch shape. Ear suppression is enabled if this bit is set to 1, and disabled if this bit is set to 0.

DISLOBJ: Disables the large object touch suppression. This allows the maximum touch suppression to be used exclusively. Large object touch suppression is disabled if this bit is set to 1, and enabled if it is set to 0 (the default).

DISTLOCK: Locks a distance touch suppression. That is, once a touch has been suppressed by distance touch suppression, it remains suppressed even if it moves outside the suppression boundary. Distance touch suppression is locked if this bit is set to 1, and unlocked if it is set to 0.

SUPDISTEN: Enables distance touch suppression, as defined by the SUPDIST field. Distance touch suppression is enabled if set to 1, and disabled if set to 0.

EDGESUP: Enables edge suppression. This operates in addition to large object touch suppression, such that if a large touch is suppressed on an edge X or Y line then all other touches on that line are also suppressed. Note that an edge line is the lowest or highest X or Y line at the edge of the sensor (for example, X0, MAXX, Y0 or MAXY).

If this bit is set to 1, then if a touch is suppressed while touching the edge of the touchscreen, all other touches that are touching that edge are also suppressed. When a suppressed touch includes these sensor edge lines, then all other touches along that edge are also suppressed, irrespective of their size.

If this bit is set to 0, then if a touch is suppressed while touching the edge of the touchscreen, no other touches that are touching that edge are suppressed (unless they are so large that they themselves are suppressed).

For Example:

Touch 1 is a suppressed large touch that overlaps X0

Touch 2 is a normal finger touch that would not normally be suppressed but which also overlaps X0

Suppression of these touches occurs as follows:

- If EDGESUP = 0: only Touch 1 is suppressed; Touch 2 is not suppressed
- If EDGESUP = 1: both Touch 1 and Touch 2 are suppressed

Note that EDGESUP relates only to the edge at which a suppressed touch is present. In the example above, if Touch 1 does not move and continues to be suppressed, but Touch 2 is removed and touches down on the sensor on another edge line (MAXX, Y0 or MAXY), then Touch 2 is not suppressed even if EDGESUP is set to 1.

MAXAPPRAREA Field

This field specifies the Maximum Approach Area Threshold. This is specified as the number of channels above the threshold based on Multiple Touch Touchscreen T100 INTTHR. If the number of channels exceeds this threshold, the touch is automatically treated as a large unintentional touch and the touch is suppressed without any further checks or suppression calculations.

The maximum Maximum Approach Area Threshold is specified as the number of channels, where a value of 0 means 40 channels.

Range: 0 (40 channels), 1 to 255 (number of channels)

MAXTCHAREA Field

This field specifies the Maximum Touch Area Threshold. This is specified as the number of channels above the Touch Threshold (TCHTHR – THCHYST) specified in the Multiple Touch Touchscreen T100 object (see ["TCHTHR Field"](#)). If the number of channels in a touch that are above (TCHTHR – THCHYST) exceeds this threshold, the touch is automatically treated as a large unintentional touch. In this case the touch is suppressed without any further checks or suppression calculations.

The touch area is specified as the number of channels, where a value of 0 means 35 channels.

Range: 0 (35 channels), 1 to 255 (number of channels)

SUPSTRENGTH Field

This field specifies the Suppression Strength. This controls the aggressiveness of the Touch Suppression T42 object (see [Table 5-11](#)). A higher value reduces the aggressiveness and allows larger touches to be reported, and a lower value increases the aggressiveness and allows only small touches to be reported.

A value of 0 means 128; that is, normal aggressiveness.

A value of 255 disables both edge suppression and large object suppression. It does not, however, affect Maximum Touch Suppression (as defined by MAXAPPAREA and MAXTCHAREA) so large touches in the center of the touchscreen can still be suppressed.

Range: See [Table 5-11](#)

TABLE 5-11: SUPSTRENGTH

Value	Meaning
0	Treated as a value 128: normal aggressiveness (default)
1 to 127	Increases aggressiveness (allows only small touches to be reported)
128	Normal aggressiveness
129 to 254	Reduces aggressiveness (allows larger touches to be reported)
255	Disables both Edge Touch Suppression and Large Object Touch Suppression (Maximum Touch Suppression is unaffected)

SUPEXTTO Field

Suppression extension timeout may extend the time needed to make a decision as to whether a touch is to be suppressed or not. The timeout applies only if the Touch Suppression T42 object cannot determine whether a touch is an actual touch or a rogue touch and requires more evidence to reach the proper decision.

The process will keep extending the DI, up to the value in the SUPEXTTO field, if it cannot determine whether a touch is an actual touch or a rogue touch until this timer expires. When this timer expires, the touch is suppressed.

The SUPEXTTO field is specified as the number of cycles, where a value of 0 disables the timeout and the determination process never expires. The effect of this is to allow the device to sleep after the normal DI period and then continue the decision process.

Range: 0 (never expires), 1 to 255 (timeout in cycles)

MAXNUMTCHS Field

This field specifies the maximum number of touches that will be reported before touches are suppressed by the maximum touch suppression. If more than MAXNUMTCHS touches are detected, the touchscreen suppresses all touches. This includes those touches already detected. The touches remain suppressed until all touches are released or a recalibration occurs.

A value of 0 ensures that all touches will be reported.

Range: 0 to 4 (maximum number of touches minus 1)

SHAPESTRENGTH Field

This field specifies the shape-based suppression strength. It is valid only if the second-level ear suppression is enabled (that is, the SHAPEEN bit of the CTRL field is set) and controls the shape-based classifier.

The range for SHAPESTRENGTH is 0 to 31, where the default value of 0 is treated as 10.

Values less than 10 result in more aggressive touch suppression; values greater than 10 result in less aggressive touch suppression.

Increasing the shape strength value above the default value of 10 allows more elongated or curved touches to be reported. Decreasing the shape strength value below the default results in the suppression of more touches so that only particularly square-shaped or circle-shaped touches are reported.

For most practical use cases, a SHAPESTRENGTH value between 5 and 20 is adequate.

Range: 0 (10), 1 to 31

SUPDIST Field

This field specifies the suppression distance for distance touch suppression. This feature suppresses false touches if they are less than SUPDIST distance from a touch that is suppressed by the large object suppression. Any touches greater than this distance from a suppressed touch remain unsuppressed.

Distance touch suppression is enabled by the SUPDISTEN bit in the CTRL field.

The distance is specified as the number of nodes, where a value of 0 means a distance of 5 nodes.

Range: 0 (5), 1 to 255 (number of nodes)

DISTHYST Field

This field specifies the hysteresis for distance touch suppression. The hysteresis is specified as the number of nodes.

Range: 0 to 255

MAXSCRNAREA Field

This field is used to determine total number of nodes in detect on the touchscreen that will trigger a full screen suppression when a palm is present. The number of nodes in detect is MAXSCRNAREA × 4. Note that number of nodes in detect are not necessarily part of a single touch.

A value of 0 (default) disables this feature. This feature is not disabled by DISLOBJ.

Range: 0 (Off), 1 to 255

Typical: 32 (128 nodes)

CFG Field

RELAXCLOSESUP: When enabled, alters the suppression algorithm to treat groups of medium-sized touches differently. If groups of close touches are triggering the suppression mechanism incorrectly, enabling this mode may improve performance.

RELAXDIAGSUP: When enabled, alters the second level ear suppression algorithm to treat diagonal "close" touches (close in terms of simple search) differently. As such, this only applies when ear suppression is enabled. If lines of small touches are triggering the ear suppression mechanism incorrectly, enabling this mode may improve performance.

SUPTCHRPTEN: Enables the reporting of touches that are suppressed by the large object touch suppression. When this feature is enabled, suppressed touches are reported via the Multiple Touch Touchscreen T100 object with a specific indication. Suppressed touch reporting is enabled if this bit is set to 1, and disabled if set to 0.

EDGSUPSTR Field

This field specifies the Edge Suppression Strength. This controls the aggressiveness of touch suppression at the edges of the sensor.

A higher value reduces the aggressiveness and allows larger touches to be reported at the edges. A lower value increases the aggressiveness and allows only small touches to be reported at the edges. The default value of 0 means 73, which is the normal aggressiveness value. Values above 127 are capped to 127. A value of 255 will disable edge classification.

Range: 0 (73), 1 to 127, 255 (disable edge classification)

5.5.3 CONFIGURATION CHECKS

A Touch Suppression T42 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed (as listed in [Table 5-12](#))

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

**TABLE 5-12: CONFIGURATION CHECKS FOR TOUCH SUPPRESSION T42
(PROCI_TOUCHSUPPRESSION_T42)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes	No	None
APPRTHR	No	No	None
MAXAPPRAREA	Yes	No	None
MAXTCHAREA	Yes	No	None
SUPSTRENGTH	Yes	No	None
SUPEXTTO	No	No	None
MAXNUMTCHS	No	No	None
SHAPESTRENGTH	Yes	No	None
SUPDIST	No	No	None
DISTHYST	No	No	None
MAXSCRNAREA	No	No	None
CFG	No	No	None
EDGSUPSTR	Yes	No	None

5.5.4 MESSAGES

No messages are generated directly for the Touch Suppression T42 object. Screen suppression messages are reported through the linked Multiple Touch Touchscreen T100 object (see [Section 4.3.4 "Messages"](#)).

5.6 Shieldless T56 Object

5.6.1 INTRODUCTION

The Shieldless T56 object has several features that allow the use of shieldless touch sensor stacks.

The Shieldless T56 object uses Optimal Integration to adjust the integration window and timing to maximize the signal-to-noise ratio from the sensor.

NOTE The entire Shieldless T56 object must be backed up and followed by a device reset when the object is enabled or disabled for use.

Note that display noise suppression for shieldless designs is provided by the filtering features of the Lens Bending T65 object (See [Section 5.7 “Lens Bending T65 Object”](#)).

5.6.2 CONFIGURATION

**TABLE 5-13: CONFIGURATION FOR SHIELDLESS T56
(PROCI_SHIELDLESS_T56)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved						RPTEN	ENABLE
1	Reserved	Reserved							
2	OPTINT	Reserved							OPTINTEN
3	INTTIME	Optimal integration time							
4	INTDELAY[0]	Optimal integration delay for X0							
		...							
4+n-1	INTDELAY[n-1]	Optimal integration delay for Xn-1							

Note: n = maximum number of X lines supported by the device (12 on mXT144U)

CTRL Field

ENABLE: Enables the use of this Shieldless T56 object. The object is enabled if set to 1, and disabled if set to 0. Note that the Shieldless T56 object must be backed up and the device reset when this object is enabled or disabled for use.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

OPTINT Field

This field sets the options for an Optimal Integration.

OPTINTEN: If this bit is set to 1, a Delay Compensation per X line is applied and the specified optimal integration time (INTTIME) is used. If this bit is set to 0, Delay Compensation is not applied per X line, and the Acquisition Configuration T8 object's Integration Window is applied.

INTTIME Field

This field sets the Optimal Integration Time. A non-zero value specifies the Optimal Integration Time plus 1. The units for this field are 24 MHz clock cycles (41.67 ns). Note that INTTIME is specified in units of 24 MHz, regardless of the actual clock speed of the device.

Range: 0 (no time), 1 to 255

INTDELAY[] Fields

These fields set the Optimal Integration Delays for each of the X lines when Optimal Integration is enabled. A non-zero value specifies the Optimal Integration Delay plus 1 for X line n . The units for this field are 24 MHz clock cycles. Note that INTDELAY is specified in units of 24 MHz, regardless of the actual clock speed of the device.

Range: 0 (no delay), 1 to 255

5.6.3 CONFIGURATION CHECKS

A Shieldless T56 object causes a configuration check to be performed in the following circumstances:

- When certain fields, including the CTRL field, are changed (as listed in [Table 5-14](#)).

In addition, some fields will cause an automatic recalibration to be performed (see [Table 5-14](#)).

Note that a check for Y line sharing is carried out during the configuration check.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

**TABLE 5-14: CONFIGURATION CHECKS FOR SHIELDLESS T56
(PROCI_SHIELDLESS_T56)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes ⁽¹⁾	Error if enabled and Multiple Touch Touchscreen T100 sharing Y0 with another enabled touch object
OPTINT	Yes ⁽²⁾	Yes ⁽²⁾	None
INTTIME	No	Yes	None
INTDELAY[]	No	Yes	None

Note 1: If the ENABLE bit is toggled on or off.

Note 2: If the OPTINTEN bit is toggled on or off.

5.6.4 MESSAGES

The message data for the Shieldless T56 object is shown in [Table 5-15](#).

**TABLE 5-15: MESSAGE DATA FOR SHIELDLESS T56
(PROCI_SHIELDLESS_T56)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved							UNCAL

STATUS Field

This field reports the current state of the Shieldless T56 object.

UNCAL: Indicates that optimal integration has been enabled but no on-chip calibrated value has been set for the integration time.

NOTE On-chip calibration is not possible on the mXT144U. However, UNCAL bit set always set to 1 in any generated message, even though on-chip calibration is not supported.

5.7 Lens Bending T65 Object

5.7.1 INTRODUCTION

The Lens Bending T65 object implements the lens bending correction algorithm. When the sensor suffers from the screen deformation (that is, lens bending) the delta values obtained by the normal measurement procedure are corrupted by the low frequency disturbance component (that is, the sensor measurements bend). The amount of bending depends on the following:

- The mechanical and electrical characteristics of the sensor
- The amount and location of the force applied by the user touch to the sensor

The Lens Bending algorithm aims to eliminate this disturbance signal from the reported deltas, and measure the bend component at the same time.

A Lens Bending T65 object processes the measurement data received from a particular instance of a Multiple Touch Touchscreen T100 object.

On the mXT144U there are 3 instances of the Lens Bending T65 object. Each instance corrects for the effects of lens distortion from a different measurement source. See [Table 5-16](#) for details.

TABLE 5-16: LENS BENDING T65 INSTANCES

Instance	Purpose
Instance 0	Corrects the mutual capacitance measurements
Instance 1	Corrects the self capacitance Y line measurements
Instance 2	Corrects the self capacitance X line measurements

5.7.2 CONFIGURATION

**TABLE 5-17: CONFIGURATION FOR LENS BENDING T65
(PROCI_LENSBENDING_T65)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	DISHIST	Reserved		DISRELEASE	DISPRESS	Reserved	RPTEN	ENABLE
1	GRADTHR	Gradient Threshold							
2	YLONOISEMUL	Y low noise multiplier LSByte							
3		Y low noise multiplier MSByte							
4	YLONOISEDIV	Y low noise divisor LSByte							
5		Y low noise divisor MSByte							
6	YHINOISEMUL	Y high noise multiplier LSByte							
7		Y high noise multiplier MSByte							
8	YHINOISEDIV	Y high noise divisor LSByte							
9		Y high noise divisor MSByte							
10	LPFILTCOEF	Low pass filter coefficient							
11	FORCESCALE	Force scale LSByte							
12		Force scale MSByte							
13	FORCETHR	Force threshold							
14	FORCETHRHYST	Force threshold hysteresis							
15	FORCEDI	Force DI							
16	FORCEHYST	Force update hysteresis							
17	ATCHRATIO	DSRATIO				Reserved	ATCHRATIO		
18 – 19	Reserved	Reserved							
20	EXFRCTHR	Excessive force threshold							
21	EXFRCTHRHYST	Excessive force threshold hysteresis							
22	EXFRCTO	Excessive force timeout							

CTRL Field

ENABLE: If set to 1, enables the use of this object.

RPTEN: Allows this object to send messages to the host through the Message Processor T5 object. Specifically it enables reporting of the estimated amount of deformation. Reporting is enabled if set to 1, and disabled if set to 0.

DISPRESS: If set to 1, disables press messages related to the amount of deformation.

DISRELEASE: If set to 1, disables release messages related to the amount of deformation.

DISHIST: Disables inter-line history. The inter-line history feature stores the results of the calculations on one line so that they can be used to affect the next. If this bit is set to 1, the use of this inter-line history is disabled.

NOTE For an instance that processes only one line of data (that is X or Y lines only), no inter-line history should be used. The DISHIST bit should therefore be set to 1 to disable the history.

GRADTHR Field

This field specifies the maximum amount of bend gradient present in the measured deltas. Sensor measurements with gradient smaller than GRADTHR are considered part of the bend and excluded from further processing. Sensor measurements with a gradient equal or greater than the GRADTHR are considered part of the touch and preserved.

Note that if the DSRATIO setting is changed (see [“ATCHRATIO Field”](#)), the gradient threshold should be adjusted accordingly.

The range of values for GRADTHR setting is 0 to 255 in units of 8-bit deltas. The default value of zero means 15.

Range: 0 (15), 1 to 255

YLONOISEMUL, YLONOISEDIV, YHINOISEMUL and YHINOISEDIV Fields

These fields allow the noise that is picked up at the edge of the screen to be scaled when the amount of noise at the edge is different to the rest of the screen.

These fields are specified using 8.8 format, where the upper 8 bits represent the integer part of the number and the lower 8 bits represent the fractional part of the number in units of 1/256. For example, a value of 1.128 represents an integer of 1 and a decimal fraction of 0.5. The default value of 0 means 256 (1.0); that is multiply by 1.

Range: 0 (256), 1 to 65535

LPFILTCOEF Field

This field specifies the low pass filter coefficient used to separate touches and bend. Increasing the value of LPFILTCOEF makes the algorithm less selective between touch and bend, but enables the algorithm to be applied with fast-changing bend.

Higher values of LPFILTCOEF should be used with higher values of the GRADTHR parameter.

The range of values is 0 to 15. The default value of zero means 5.

Range: 0 (5), 1 to 15

FORCESCALE Field

This field scales the size of the reported deformation (bend size). The value is represented in 8.8 format, where a default of 0 means 256. (see the YLONOISEMUL, YLONOISEDIV, YHINOISEMUL and YHINOISEDIV fields for details).

Range: 0 (256), 1 to 65535

FORCETHR Field

This field configures the Force Threshold above which a force event is reported. If the force is greater than or equal to this threshold, then a force event is reported. In addition, when the force transitions from below this threshold to above this threshold, a press event is reported. The default value of 0 means 64.

Range: 0 (64), 1 to 255

FORCETHRHYST Field

This field configures the Force Threshold Hysteresis below which a release event is reported. If the force is transitions below $\text{FORCETHR} - \text{FORCEHYST}$, then a release event is reported and force events stop being reported.

Range: 0 to 255

FORCEDI Field

This field specifies the Detect Integration for the deformation press and release events. To suppress false detections caused by spurious events like electrical noise, the device incorporates a detection integrator (FORCEDI) counter mechanism to provide signal filtering. A counter is incremented each measurement cycle that a force is detected. When this counter reaches the FORCEDI setting limit the force is finally declared to be present. If on any acquisition a force is not seen to exceed FORCETHR the counter is cleared and the process has to start from the beginning.

NOTE The detection integration counter associated with FORCEDI is processed only when the associated measurement for the Lens Bending T65 instance is made (see [Table 5-16](#)). This may result in many more cycles being performed before FORCEDI elapses if the configuration is such that the associated measurement is not made on every cycle.

An opposite process is applied when a force leaves detection. The counter is decremented each cycle that the force is below $\text{FORCETHR} - \text{FORCETHRHYST}$, and incremented again when the force exceeds FORCETHR . When the counter reaches zero, the force is finally declared to be out of detect. In this case there is an additional extra cycle (that is, the number of cycles is $\text{FORCEDI} + 1$).

The range for this field is 0 to 255, where 0 is the same as 1.

Range: 0 (1), 1 to 255 (number of cycles)

FORCEHYST Field

This field configures the Force Hysteresis for the reporting threshold (that is, the amount of change necessary for the force to be reported). A value of 255 disables the force value reporting.

Range: 0 to 254, 255 (disable reporting)

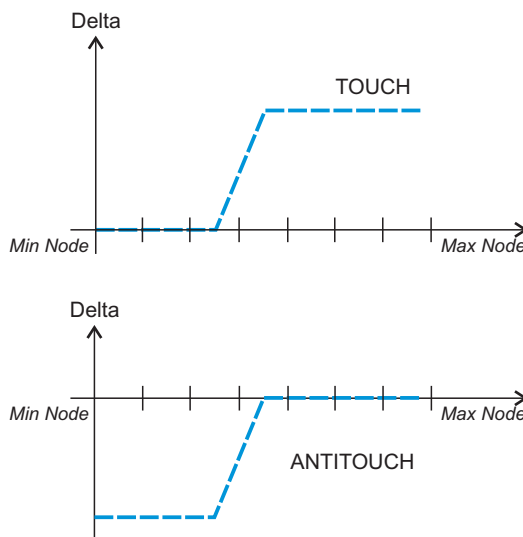
ATCHRATIO Field

ATCHRATIO: Specifies the ratio between the positive and negative slew rate limit. This parameter is used to bias the touch estimate towards a touch where the touch covers more than 50% of the nodes on the line being measured. Due to the presence of LCD noise, a touch covering $\geq 50\%$ of the nodes on the line being measured can be interpreted either as a touch or as an antitouch as shown in [Figure 5-13](#).

This ambiguity in interpretation of the measured delta is resolved using ATCHRATIO. A higher ATCHRATIO biases the lens bending algorithm towards a touch.

If the value specified for the ATCHRATIO field exceeds the maximum value allowed, it is internally capped to the maximum (4).

ATCHRATIO Range: 0 (2), 1 to 4

FIGURE 5-13: LARGE TOUCH INTERPRETATION AS TOUCH OR ANTITOUCH DUE TO LCD NOISE

DSRATIO: Defines the Downsampling Ratio. This is how many lines should be used in the anti touch algorithm. For example, if DSRATIO is set to 2, then every other line is analyzed, if it is set to 3 then every third line is analyzed, and so on. This reduces the processing time for lens bending on larger screens. The range for DSRATIO is 0 to 3, where the default value of 0 means 2. In addition, a value of 15 means disable downsampling.

If the value specified for the DSRATIO field exceeds the maximum value allowed, it is set to the default value of 2.

DSRATIO Range: 0 (2), 1 to 3, 15 (downsampling disabled)

EXFRCTHR Field

This field specifies the Excessive Force Threshold. The effect of this field depends on the object instance and the type of data:

- For a Lens Bending T65 instance that handles self capacitance measurement type data, if the reported deformation (bend size) on the current primary measurement type exceeds this threshold, then the device will start to use the mutual capacitance measurement type. Note that the deformation that is checked against this threshold is the one being reported (that is, the one being re-scaled by the FORCESCALE control).
- For a Lens Bending T65 instance that handles mutual capacitance measurement type data, if the reported deformation on the current primary measurement type exceeds this threshold, then the device will generate a positive edge event for handling via the Dynamic Configuration Controller T70 object.

The range for EXFRCTHR is 0 to 255 (in units of force), where a value of 0 disables this feature.

Range: 0 (disabled), 1 to 255

EXFRCTHRHYST Field

This field specifies the Excessive Force Threshold Hysteresis. The effect of this field depends on the object instance and the type of data:

- For a Lens Bending T65 instance that handles self capacitance measurement type data, if the reported deformation (bend size) on the current primary measurement type is below ($\text{EXFRCTHR} - \text{EXFRCTHRHYST}$), then the system will stop using the mutual capacitance measurement type once the timeout has expired (see [“EXFRCTO Field”](#)), but only if the mutual capacitance measurement type is already running. Note that the deformation that is checked against this threshold is the one being reported (that is, the one being re-scaled by the FORCESCALE field).

- For a Lens Bending T65 instance that handles mutual capacitance measurement type data, if the reported deformation (bend size) on the current primary measurement type is below (EXFRCTHR – EXFRCTHRHYST), then the device will generate a negative edge event for handling via the Dynamic Configuration Controller T70 object, but only once the timeout has expired (see “EXFRCTO Field”).

Range: 0 to (EXFRCTHR – 1)

EXFRCTO Field

This field specifies the Excessive Force Timeout. The effect of this field depends on the object instance and the type of data:

- For a Lens Bending T65 instance that handles self capacitance measurement type data, this timeout determines how long the system will use the mutual capacitance measurement type after the reported deformation (bend size) has dropped below (EXFRCTHR – EXFRCTHRHYST). Note that if the reported deformation (bend size) goes above (EXFRCTHR – EXFRCTHRHYST) during the timeout period, the timer is reset.
- For a Lens Bending T65 instance that handles mutual capacitance measurement type data, this timeout determines if a negative edge event is generated for handling via the Dynamic Configuration Controller T70 object.

The timeout is specified in units of 200 ms, where the default value of 0 means no timeout.

Range: 0 (no timeout), 1 to 255

5.7.3 CONFIGURATION CHECKS

The Lens Bending T65 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed (as listed in [Table 5-18](#))

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

TABLE 5-18: CONFIGURATION CHECKS FOR LENS BENDING T65 (PROCI_LENSBENDING_T65)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes	None
GRADTHR	No	No	None
YLONOISEMUL	No	Yes	None
YLONOISEDIV	No	Yes	None
YHINOISEMUL	No	Yes	None
YHINOISEDIV	No	Yes	None
LPFILTCOEF	No	No	None
FORCESCALE	No	No	None
FORCETHR	No	No	None
FORCETHRHYST	No	No	None
FORCEDI	No	No	None
FORCEHYST	No	No	None
ATCHRATIO	No	No	None
EXFRCTHR	No	No	None
EXFRCTHRHYST	No	No	None
EXFRCTO	No	No	None

Note 1: If the ENABLE bit is toggled on or off.

5.7.4 MESSAGES

The message data for the Lens Bending T65 object is shown in [Table 5-19](#).

**TABLE 5-19: MESSAGE DATA FOR LENS BENDING T65
(PROCI_LENSBENDING_T65)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Detect	Press	Release	Reserved				
2	FORCE	Force Level							

STATUS Field

DETECT: This bit indicates the current force state. If the current force level is above FORCETHR, this bit is set to 1; if current force level is below (FORCETHR – FORCETHRHYST), this bit is set to 0.

PRESS: Indicates a press event. This bit is set if DETECT changes from 0 to 1; that is, the level of deformation was less than (FORCETHR – FORCETHRHYST) and has now changed to greater than FORCETHR, and has been so for FORCEDI cycles.

RELEASE: Indicates a release event (DETECT = 0). This bit is set if DETECT changes from 1 to 0; that is, the level of deformation was above FORCETHR and has now changed to less than (FORCETHR – FORCETHRHYST), and has been so for FORCEDI cycles.

FORCE Field

This field reports the current calculated force. This is rescaled by FORCESCALE and updated once it changes by at least FORCEHYST.

5.8 Noise Suppression T72 Object

5.8.1 INTRODUCTION

The Noise Suppression T72 object provides a noise suppression algorithm to suppress the effects of external noise.

5.8.1.1 Noise Notification Mechanisms

The Noise Suppression T72 object needs to be made aware of when a noise source appears and disappears (for example, when a charger is turned on or off) or when the current noise level rises to a level beyond which the current acquisition parameters can cope. The following notification mechanisms are available:

- **Software trigger** – The host can set the NOISEON bit in the CFG1 field when a noise source appears. For example, this bit should be set when a noisy charger is plugged into the user's product. Note that the alternative use of the NOISEON bit in the CALCFG1 field is not recommended.

IMPORTANT! Enabling and disabling the Noise Suppression T72 object dynamically using the CTRL ENABLE bit in response to noise is not recommended, as this can cause unstable behavior. The correct procedure is to enable the object at startup and then use one of the notification methods above whenever noise is detected.

5.8.1.2 Noise Measurement

The Noise Suppression T72 object schedules the acquisition of numerous noise measurements so that suitable actions can be taken to best minimize the effects of such noise (see [Section 5.8.1.3 “Transitions Between Noise States”](#) and [Section 5.8.1.4 “Frequency Hopping”](#)).

The NLGAINDUALX and NLGAINSINGX fields determine the gain to be used for the noise lines measurements.

Suggested initial settings for the Noise Suppression T72 object are given in [Table 5-20](#).

TABLE 5-20: SUGGESTED INITIAL SETTINGS – NOISE MEASUREMENT

Field	Suggested Initial Setting
NLGAINDUALX	Field needs tuning; depends on whether Dual X is running or not. Start with Multiple Touch Touchscreen T100 Dual X gain.
NLGAINSINGX	Field needs tuning; depends on whether Dual X is running or not. Start with Multiple Touch Touchscreen T100 non Dual (single) X gain.

5.8.1.3 Transitions Between Noise States

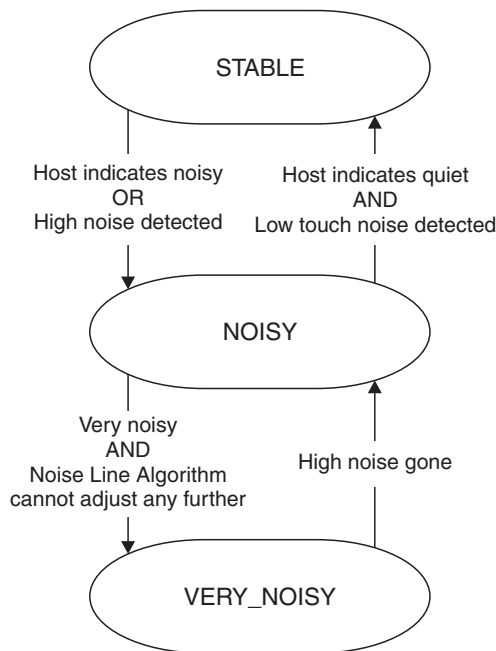
As the noise suppression algorithm is notified of increasing or decreasing levels of noise, it transitions between the STABLE, NOISY and VERY_NOISY states (see [Figure 5-14](#)). In each state it can dynamically load a new set of acquisition parameters to cope with the current level of noise. This provides a framework within which the behavior of the Noise Suppression T72 object can be modified to provide better noise handling at the expense of touch performance and/or touch message rate. The thresholds that determine the transitions between the states are specified by the NOISLOWNLTHR, VNOISLOWNLTHR, STABHIGNLTHR and NOISHIGNLTHR fields.

The Noise Suppression T72 object permits noise state promotions and demotions in one of two circumstances:

- If the measured noise exceeds STABHIGNLTHR or NOISHIGNLTHR (when in STABLE or NOISY state respectively), or falls below VNOISLOWNLTHR or NOISLOWNLTHR (when in VERY_NOISY or NOISY state respectively).
- If the measured noise exceeds the maximum value of NLTHR, and all available hops have been taken (see [“HOPCNT Field”](#) and [“HOPCNTPER Field”](#)).

The DISNLTHRSCHG bit in CFG3 field (see [“CFG3 Field”](#)) determines whether one or both circumstances are used. When DISNLTHRSCHG is set to 0, both circumstances will cause a noise state promotion or demotion. When DISNLTHRSCHG is set to 1, such promotions and demotions occur in the first circumstance only. In this case, the NLTHR functionality specifically manages frequency hopping only, making its configuration much simpler.

FIGURE 5-14: NOISE SUPPRESSION STATES



On entry to a new state, the Noise Suppression T72 object loads the appropriate set of control, frequency and ADCs per X parameters and touchscreen is recalibrated to use the new settings.

Settings for any object on the device can also be loaded using the event-handling capabilities of the Dynamic Configuration Controller T70 object (see [Section 6.9 “Dynamic Configuration Controller T70 Object”](#)). The current state is sent as an event to the Dynamic Configuration Controller T70 object on entry to a new state. The Dynamic Configuration Controller T70 object can then respond to this event and load new parameters. For example, by responding to a transition from the NOISY to VERY_NOISY state, more aggressive settings for the Multiple Touch Touchscreen T100 object can be loaded.

5.8.1.4 Frequency Hopping

The noise suppression algorithm can change the burst frequency if the current burst frequency becomes too noisy. A set of five candidate frequencies can be defined for each state and the Noise Suppression T72 will use one of these frequencies when in operation. At suitable intervals between touch measurement cycles, the Noise Suppression T72 object perform background noise scans for each of the five frequency presets to ensure that the optimal frequency is always selected. The sets of candidate frequencies are configured using the STABFREQ, NOISFREQ and VNOIFREQ fields.

The Noise Suppression T72 object provides two frequency hopping mechanisms:

- **Internal Analysis of the Noise Lines Threshold** – For frequency hopping to occur, the measured noise must meet one of the following conditions:
 - The measured noise is greater than the maximum NLTHR (as defined by MINNLTHR + INCNLTHR; see [“MINNLTHR Field”](#) and [“INCNLTHR Field”](#)); and the sensor is in either an in-touch or no-touch state
 - The measured noise is less than the maximum NLTHR and greater than the current NLTHR / 2; the sensor is in an in-touch state for a specified hop evaluation period (see [“HOPEVALTO Field”](#)); and the allowed hops have not all been taken (see [“HOPCNT Field”](#) and [“HOPCNTPER Field”](#)).
- **Continuous frequency hopping mechanism** – Frequency hops occur whenever suitable candidate frequencies are determined by comparing the measured noise against the NOTCHMINDIFF, TCHMINDIFF, NOTCHMINHOP and TCHMINHOP fields. In this case, the decision to hop is free of any restrictions placed by the complex comparisons to NLTHR, and is made irrespective of the magnitude of measured noise at a given juncture.

The DISNLTHRHOP bit in the CFG3 field determines which mechanism is used (see [“CFG3 Field”](#)).

Suggested initial settings for frequency hopping are given in [Table 5-21](#).

TABLE 5-21: SUGGESTED INITIAL SETTINGS – FREQUENCY HOPPING

Field	Suggested Initial Setting
CFG3	216 (OPTBGSCANEN = 1, TCHHOFFEN ⁽¹⁾ = 1, DISNLTHRHOP = 1, DISNLTHRSCHG = 1)
BGSCAN	11 (SUPFRQHOPEN=1, SUPPCOEFF=3)

1. For details regarding TCHHOFFEN, see [Section 5.8.1.6 “Background Scanning and Touch Hold-off Qualification”](#)

The subsequent suggested initial settings for the Noise Suppression T72 object depend on the specific configuration of DISNLTHRHOP and DISNLTHRSCHG, as shown in [Table 5-22](#).

TABLE 5-22: SUGGESTED INITIAL SETTINGS – FREQUENCY HOPPING

Field	DISNLTHRHOP = 0 (DISNLTHRSCHG = X)	DISNLTHRHOP = 1	
		DISNLTHRSCHG = 0	DISNLTHRSCHG = 1
HOPST	5	0 (default)	0 (default)
HOPCNT	5 (default)	255 ⁽¹⁾	255 ⁽¹⁾
HOPCNTPER	10 (default)	1 ⁽¹⁾	1 ⁽¹⁾
HOPEVALTO	5	5	Not used
MINNLTHR	Field needs tuning ⁽²⁾	Field needs tuning ⁽²⁾	Not used
INCNLTHR	MINNLTHR	MINNLTHR	Not used
FALLNLTHR	Field needs tuning	Field needs tuning	Not used
STABFREQ[]	Fields need tuning	Fields need tuning	Fields need tuning
NOISFREQ[]	Fields need tuning	Fields need tuning	Fields need tuning
VNOIFREQ[]	Fields need tuning	Fields need tuning	Fields need tuning
NOTCHMINDIFF	0 (default)	Field needs tuning	Field needs tuning
TCHMINDIFF	0 (default)	Field needs tuning	Field needs tuning
NOTCHMINHOP	0 (default)	Field needs tuning	Field needs tuning
TCHMINHOP	0 (default)	0 (default)	Field needs tuning

1. Setting prevents unnecessary noise state promotions.

2. Depends on whether Dual X is running or not. Start with Multiple Touch Touchscreen T100 TCHTHR / 2.

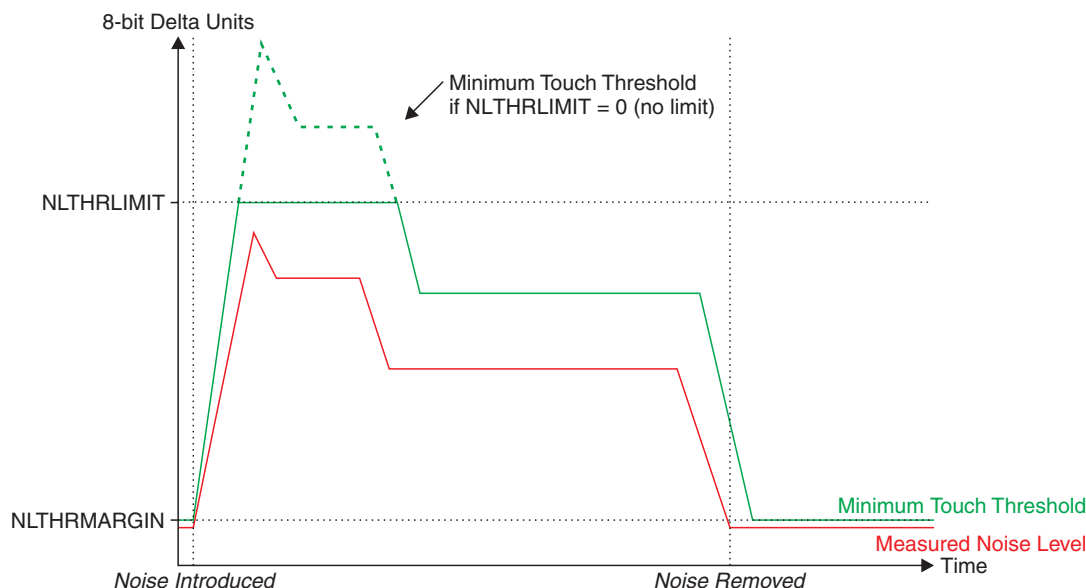
5.8.1.5 Minimum Touch Threshold

The Noise Suppression T72 object can provide a Minimum Touch Threshold below which the touch objects should consider all delta data to be caused by noise. This is controlled by the MINTHRADJ field (MINTHRFACTOR and MINTHRFRAC bits) and the NLTHRMARGIN field. See the other objects in this protocol guide for details about how they respond to a minimum threshold being supplied by the Noise Suppression T72 object. In general, they will desensitize to avoid interpreting deltas caused by noise as touches.

The Minimum Touch Threshold output by the Noise Suppression T72 object is a gain-adjusted value derived as follows:

$$\text{Minimum Touch Threshold} = \left(\frac{NL \times ((MINTHRFACTOR \times 16) + MINTHRFRAC)}{16} \right) + NLTHRMARGIN$$

where NL is the measured noise level.

FIGURE 5-15: MINIMUM TOUCH THRESHOLD

$$\text{Minimum Touch Threshold} = (\text{Measured Noise Level} \times ((\text{MINTHRFACTOR} \times 16) + \text{MINTHRFRAC}) / 16) + \text{NLTHRMARGIN}$$

Suggested initial settings for the Noise Suppression T72 object are given in [Table 5-23](#).

TABLE 5-23: SUGGESTED INITIAL SETTINGS – MINIMUM TOUCH THRESHOLD

Field	Suggested Initial Setting
NLTHRMARGIN	2
MINTHRADJ	19 (MINTHRFACTOR = 1, MINTHRFRAC = 3)
NLTHRLIMIT	Field needs tuning

5.8.1.6 Background Scanning and Touch Hold-off Qualification

For the Noise Suppression T72 object to perform effective frequency hopping, continuous noise measurements are made for each of the candidate presets. This allows for the selection of the optimal settings that can best avoid the effects of noise.

The Noise Suppression T72 object maintains two sets of IIR filter measurements (carousels), each set associated with a specific touch or no-touch state.

A pre-qualification process is applied to background noise measurements. This collates the measurements until all the measurements are deemed to have been collated within a period of consistent touch-state reporting. Once this is achieved the carousel filters are updated and any subsequent frequency hopping determination is made. If a touch state transition occurs during a measurement cycle, the noise measurements made during the transition are rejected and a new carousel collation begins. This qualification process is referred to as Touch Hold-off.

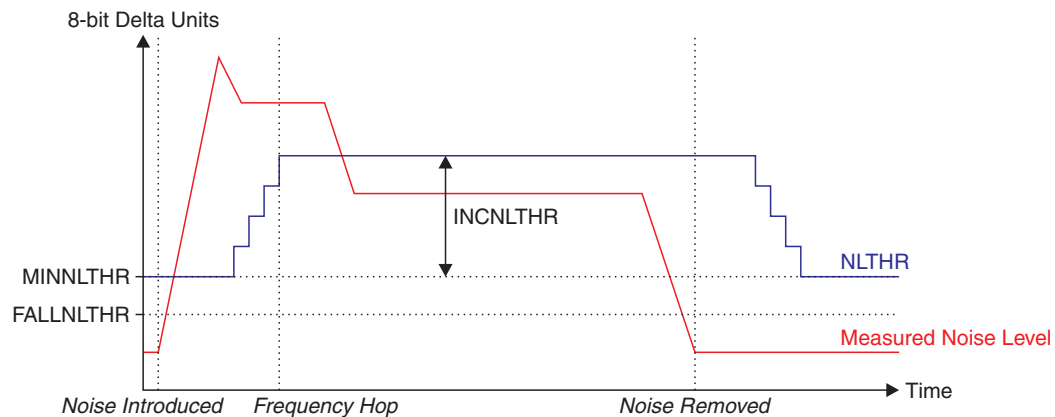
The Touch Hold-off mechanism can improve the frequency hopping candidate selection process, as the filter carousel content is more accurate when repeated touch transitions take place. Without this process, mis-reports from false touches and/or momentary loss of touch determination can reduce the effectiveness of any frequency hopping.

Touch Hold-off is selectable using the TCHOFFEN bit in the CFG3 field (see [“CFG3 Field”](#)).

5.8.1.7 Noise Lines Threshold

The Noise Suppression T72 object maintains an internal Noise Lines Threshold (NLTHR) against which the measured noise can be compared. This allows an appropriate action to be taken, such as transitioning to a different noise state or choosing a new burst frequency (see [Section 5.8.1.4 “Frequency Hopping”](#)).

The INCNLTHR, FALLNLTHR and MINNLTHR fields are used to determine how the NLTHR is increased and decreased (see [Figure 5-16](#)).

FIGURE 5-16: NOISE LINES THRESHOLD

Suggested initial settings for the Noise Suppression T72 object are given in [Table 5-24](#).

TABLE 5-24: SUGGESTED INITIAL SETTINGS – NOISE LINES THRESHOLD

Field	Suggested Initial Setting
MINNLTHR	Field needs tuning; Depends on whether Dual X is running or not. Start with Multiple Touch Touchscreen T100 TCHTHR / 2
INCNLTHR	MINNLTHR
FALLNLTHR	Field needs tuning

5.8.1.8 Adjusting the Number of ADCs per X

Each of the five candidate frequencies has a corresponding explicit no touch and touch ADCs Per X settings, which are applied whenever frequency hopping occurs. This facilitates the following capabilities:

- In normal conditions, the combination of frequency and ADCs Per X can be controlled to ensure message report rate requirements can be met.
- In very high noise conditions, a larger number of ADCs Per X could be used to improve noise performance at the expense of message rate and power consumption.
- In low noise conditions, or with no touch detected, a smaller number of ADCs Per X could be used to help reduce power consumption.

5.8.1.9 Dual X Drive

The Noise Suppression T72 object provides an alternative burst mode on the X lines known as Dual X Drive.

With Dual X Drive, X pulses occur on pairs of X lines instead of single X lines. When Dual x is enabled, the noise suppression algorithm can use Dual X Drive to generate acquisition pulses on pairs of X lines instead of single X lines (X0 and X1, then X1 and X2, and so on). Note that Dual X Drive occurs only on those X lines that fall within the area of the sensor covered by an enabled Multiple Touch Touchscreen T100 object.

Dual X mode is useful when finger touches will cover more than one X line on a closely spaced X sensor matrix as it improves the signal-to-noise ratio (SNR). Note, however, that it will affect accuracy and linearity on the X edge. Furthermore, the enlarged touch means that the two-touch separation is increased and the classification of stylus touches is made more difficult.

Dual X operation is selected according to the DUALXMODE bit in the STABCTRL, NOISCTRL or VNOICTRL field (as appropriate).

5.8.1.10 Resetting the Noise Suppression Algorithm

The noise suppression algorithm is reset whenever any of the following occurs:

- The ENABLE bit in the CTRL field is changed (note that it is not recommended to do this dynamically)
- The CALCFG1 field is changed

5.8.1.11 General Initial Settings

Suggested initial settings for fields that have not been listed in earlier sections are given in [Table 5-25](#).

TABLE 5-25: SUGGESTED INITIAL SETTINGS

Field	Suggested Initial Setting
CTRL	ENABLE = 1 Other bits as required by user
CALCFG1	0
CFG1	0 when a potential noise source is not present (for example, a charger is turned off) 1 (NOISEON = 1) when a potential noise source is notified by the host (for example, a charger is turned on)
CFG2	0
BLKNLTHR	0
STABCTRL	0
STABTCHAPX[]	Fields need tuning
STABNOTCHAPX[]	Fields need tuning
STABHIGHNLTHR	Field needs tuning
NOISCTRL	9 (AUTO = 1, DUALXMODE = 1)
NOISCHAPX[]	Fields need tuning
NOISNOTCHAPX[]	Fields need tuning
NOISLOWNLTHR	Field needs tuning
NOISHIGHNLTHR	Field needs tuning
NOISCNT	3 (NOTCH = 3)
VNIOCTRL	9 (AUTO = 1, DUALXMODE = 1)
VNOIFREQ[]	Fields need tuning
VNOITCHAPX[]	Fields need tuning
VNOINOTCHAPX[]	Fields need tuning
VNOILOWNLTHR	Field needs tuning
VNOICNT	3 (NOTCH = 3)

5.8.1.12 Event Triggers

The Noise Suppression T72 object does not directly request a change in the acquisition parameters to the measurement layer (such as accessed frequency, number of ADCSPERX, detection thresholds, and so on). Instead, the measurement parameters can be altered indirectly by the means of the events mechanism provided by the Dynamic Configuration Controller T70 object. In this case, as soon as a new noise state is entered, the Noise Suppression T72 object will request the triggering of an event to the Dynamic Configuration Controller T70 object.

5.8.2 CONFIGURATION

**TABLE 5-26: CONFIGURATION FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved	RPTDUALX	RPTNLTHR	RPTNOISELVL	RPTSTATE	RPTACQ	RPTEN	ENABLE
1	CALCFG1	Reserved						VNOISEON	NOISEON
2	CFG1	Reserved						VNOISEON	NOISEON
3	CFG2	NOTCHNOISEON	Reserved						NOBLKTCH
4	Reserved	Reserved							
5	HOPCNT	Hop Count							

**TABLE 5-26: CONFIGURATION FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6	HOPCNTPER	Hop Count Period							
7	HOPEVALTO	Hop Evaluation Timeout							
8	HOPST	Hop Suspend Time							
9	NLGAINDUALX	GAIN							
10	MINNLTHR	Minimum Noise Lines Threshold							
11	INCNLTHR	Increase Noise Lines Threshold							
12	FALLNLTHR	Falling Noise Level Threshold							
13	NLTHRMARGIN	Minimum Noise Lines Threshold Margin							
14	MINTHRADJ	MINTHRFACTOR				MINTHRFRAC			
15	NLTHRLIMIT	Minimum Touch Threshold limit							
16	BGSCAN	RATIO				SUPPFREQHOPEN	SUPPCOEFF		
17	NLGAINSINGX	GAIN							
18	BLKNLTHR	Noise line limit for blocking touch processing							
19	CFG3	OPTBGSCANEN	TCHHOFFEN	Reserved	DISNLTHRSCHG	DISNLTHRHOP	NUMPRESETS		
20	STABCTRL	CAL	Reserved			DUALXMODE	Reserved		NOTCHTHR
21	STABFREQ[]	STABLE State Frequency Setting 0							
22		STABLE State Frequency Setting 1							
23		STABLE State Frequency Setting 2							
24		STABLE State Frequency Setting 3							
25		STABLE State Frequency Setting 4							
26	STABTCHAPX[]	STABLE State Touch ADCSPERX Setting 0							
27		STABLE State Touch ADCSPERX Setting 1							
28		STABLE State Touch ADCSPERX Setting 2							
29		STABLE State Touch ADCSPERX Setting 3							
30		STABLE State Touch ADCSPERX Setting 4							
31	STABNOTCHAPX[]	STABLE State No Touch ADCSPERX Setting 0							
32		STABLE State No Touch ADCSPERX Setting 1							
33		STABLE State No Touch ADCSPERX Setting 2							
34		STABLE State No Touch ADCSPERX Setting 3							
35		STABLE State No Touch ADCSPERX Setting 4							
36 – 37	Reserved	Reserved							
38	STABHIGHNLTHR	STABLE State High noise detection level							
39	Reserved	Reserved							
40	NOISCTRL	CAL	Reserved			DUALXMODE	Reserved		AUTO
41	NOISFREQ[]	NOISY State Frequency Setting 0							
42		NOISY State Frequency Setting 1							
43		NOISY State Frequency Setting 2							
44		NOISY State Frequency Setting 3							
45		NOISY State Frequency Setting 4							
46	NOISTCHAPX[]	NOISY State Touch ADCSPERX Setting 0							
47		NOISY State Touch ADCSPERX Setting 1							
48		NOISY State Touch ADCSPERX Setting 2							
49		NOISY State Touch ADCSPERX Setting 3							
50		NOISY State Touch ADCSPERX Setting 4							

**TABLE 5-26: CONFIGURATION FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
51	NOISNOTCHAPX[]	NOISY State No Touch ADCSPERX Setting 0							
52		NOISY State No Touch ADCSPERX Setting 1							
53		NOISY State No Touch ADCSPERX Setting 2							
54		NOISY State No Touch ADCSPERX Setting 3							
55		NOISY State No Touch ADCSPERX Setting 4							
56	Reserved	Reserved							
57	NOISLOWNLTHR	NOISY State Low noise detection level							
58	NOISHIGHLTHR	NOISY State High noise detection level							
59	NOISCNT	LOWNOISTCH				NOTCH			
60	VNOICTRL	CAL	Reserved			DUALXMODE	Reserved		AUTO
61	VNOIFREQ[]	VERY_NOISY State Frequency Setting 0							
62		VERY_NOISY State Frequency Setting 1							
63		VERY_NOISY State Frequency Setting 2							
64		VERY_NOISY State Frequency Setting 3							
65		VERY_NOISY State Frequency Setting 4							
66	VNOITCHAPX[]	VERY_NOISY State Touch ADCSPERX Setting 0							
67		VERY_NOISY State Touch ADCSPERX Setting 1							
68		VERY_NOISY State Touch ADCSPERX Setting 2							
69		VERY_NOISY State Touch ADCSPERX Setting 3							
70		VERY_NOISY State Touch ADCSPERX Setting 4							
71	VNOINOTCHAPX[]	VERY_NOISY State No Touch ADCSPERX Setting 0							
72		VERY_NOISY State No Touch ADCSPERX Setting 1							
73		VERY_NOISY State No Touch ADCSPERX Setting 2							
74		VERY_NOISY State No Touch ADCSPERX Setting 3							
75		VERY_NOISY State No Touch ADCSPERX Setting 4							
76	Reserved	Reserved							
77	VNOILOWNLTHR	VERY_NOISY State Low noise detection level							
78	Reserved	Reserved							
79	VNOICNT	LOWNOISTCH				NOTCH			
80	Reserved	Reserved							
81	NOTCHMINDIFF	Minimum measured filter range for frequency hopping – no touch							
82	TCHMINDIFF	Minimum measured filter range for frequency hopping – in touch							
83	NOTCHMINHOP	Minimum quieter hop filter range for frequency hopping – no touch							
84	TCHMINHOP	Minimum quieter hop filter range for frequency hopping – in touch							
85	BGNLRATIOSTAB	TCH				NOTCH			
86	BGNLRATIONOIS	TCH				NOTCH			
87	BGNLRATIOVNOI	TCH				NOTCH			
88	NLRATIO	NOTSTAB				STAB			

CTRL Field

This is the main control field.

ENABLE: Enables the Noise Suppression T72 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows this object to send messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

RPTACQ: Allows selected frequency or ADCs per X changes to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTSTATE: Allows changes in noise state or blocking of touch processing to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTNOISELVL: Allows noise level changes to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTNLTHR: Allows noise lines threshold changes to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTDUALX: Allows changes between Dual X and non Dual X states to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

CALCFG1 Field

This field repeats bits from CFG1 field so that changing these bits causes a touchscreen recalibration.

NOTE	Use of this field is not recommended.
-------------	---------------------------------------

NOISEON: See “CFG1 Field”.

VNOISEON: See “CFG1 Field”.

CFG1 Field

This field repeats bits from CALCFG1 field so that these bits can be changed without causing a touchscreen recalibration.

NOISEON: This bit can be set to 1 by the host to indicate that a noise-source is activated for the product. The object state-machine will change to the NOISY state if it was previously in the STABLE state.

VNOISEON: Manually forces a change to the VERY_NOISY state for use during setup or development. If this bit is set to 1, the object state machine changes to the VERY_NOISY state if it was previously in the NOISY or STABLE state.

CFG2 Field

NOBLKTCH: This bit can be set to 1 by the host to disable the blocking of touch processing. When this bit is set to 0, touch processing is blocked whenever the condition to request a frequency hop is identified. No frequency change is made if the best frequency is already selected or the maximum number of hops has been performed. This feature is disabled when this bit is set to 1.

NOTCHNOISEON: Enables demotion of noise-level from either NOISY or VERY_NOISY states when clamped to one of these noise-levels by either the NOISEON or VNOISEON bits (CALCFG1 or CFG1 field) respectively. Consequently, this field alters the default behavior when noise-level promotion is enforced by configuration settings established by the host device.

Such a demotion can only occur during continuous touch, whereby noise-levels are deemed acceptably low. Essentially this bit allows the device to assume that noise is present until a touch is introduced for a sufficient time to determine its demotion to a quieter noise state.

If such a demotion occurs, then as soon as touch is removed, the noise-level will revert to the original state, as mandated by the configuration bits specified above.

HOPCNT Field

This field specifies the Hop Count. This limits the number of allowed frequency jumps within the Hop Count Period before the object takes counter-measures to improve performance (for example, frequency hopping and ultimately advancing the state machine). The default value of 0 means 5 hops.

Range: 0 (5), 1 to 255 (number of hops)

HOPCNTPER Field

This field specifies the Hop Count Period. This specifies the time period during which the number of frequency jumps is compared to the HOPCNT limit. The Hop Count Period is specified in units of 100 acquisition cycles, where the default value of 0 means 10 (1000 acquisition cycles).

Range: 0 (10 = 1000 acquisition cycles), 1 to 255

HOPEVALTO Field

This field specifies the Hop Evaluation Timeout. This setting provides the opportunity to hop to a quieter frequency when the currently selected frequency is consistently noisy (but does not suffer from high noise to initiate a frequency hop). The Hop Evaluation Timeout is specified in units of 10 acquisition cycles, where the default value of 0 means 10 (100 acquisition cycles).

Range: 0 (10 = 100 acquisition cycles), 1 to 255

HOPST Field

This field specifies the Hop Suspend Time. This is the time period from when the last frequency hop occurred to when a new frequency hop is allowed. The Hop Suspend Time is specified in units of 10 acquisition cycles, where the default value of 0 indicates that no suspending restriction is placed between hops. This indicates that such hops can take place whenever the IIR filter carousel analysis deems that a more suitable candidate is available.

Range: 0 (no suspending restriction), 1 to 255

NLGAINDUALX Field

This field specifies the Noise Lines Gain for Dual X. Note that this is the gain for use when making noise line measurements; the gain for touch measurements is set by the Multiple Touch Touchscreen T100 object.

GAIN: The noise line gain to be used.

GAIN Range: 0 to 22

MINNLTHR Field

This field specifies the Minimum Noise Lines Threshold (in 8-bit deltas). The noise suppression algorithm sometimes compares the measured noise level to this field, or to a proportion of this field, in its decision making process (see [Figure 5-16](#)). The default value of 0 means 32.

Range: 0 (32), 1 to 255 (8-bit deltas)

INCNLTHR Field

This field specifies the Increment for the Noise Lines Threshold (in 8-bit deltas). This is how high the noise suppression algorithm can increase the reported noise lines threshold (see [Figure 5-16](#)). If this field is set to 0, then the noise suppression algorithm will never try to increase the reported noise lines threshold (and will therefore never try to enforce a Minimum Touch Threshold).

The host should not change this field dynamically at runtime without resetting the noise suppression algorithm.

Range: 0 to 255 (8-bit deltas)

FALLNLTHR Field

This field specifies the Falling Noise Lines Threshold (in 8-bit deltas). This threshold is used by the noise suppression algorithm to decide when it can decrease the noise lines threshold (see [Figure 5-16](#)).

Range: 0 to 255 (8-bit deltas)

NLTHRMARGIN Field

This field specifies the Noise Lines threshold Margin (in 8-bit deltas). This is the amount by which the Minimum Touch Threshold can be increased, compared to the reported noise lines threshold, when the noise suppression algorithm is enforcing a Minimum Touch Threshold (see [Figure 5-15](#)). Note that the gain setting specified by the NLGAIN field is applied. This means that the Minimum Touch Threshold might not be denominated in the same units as the Touch Thresholds in other touch objects if the NLGAIN setting is different to the GAIN setting of other touch objects.

NOTE	The host should not change this field dynamically at runtime without resetting the noise suppression algorithm.
-------------	---

Range: 0 to 255 (8-bit deltas)

MINTHRADJ Field

This field specifies the Minimum Threshold Adjustment. This is used to determine the Minimum Touch Threshold (see [Figure 5-15](#)). The Minimum Touch Threshold is calculated from the measured noise level using the following formula:

$$\text{Minimum Touch Threshold} = \left(\frac{NL \times ((\text{MINTHRFACTOR} \times 16) + \text{MINTHRFRAC})}{16} \right) + NL\text{THRMARGIN}$$

where NL is the measured noise level

MINTHRFRAC: configures the Minimum Touch Threshold Fractional component for use in the minimum touch threshold calculation. This effectively is 1/16 of the scaling applied to measured noise to calculate the Minimum Touch Threshold.

MINTHRFRAC Range: 0 to 15

MINTHRFACTOR: Configures the Minimum Touch Threshold Factor component for use in the minimum touch threshold calculation.

MINTHRFACTOR Range: 0 to 15

NLTHRLIMIT Field

This field specifies the maximum allowed value for the calculated Minimum Touch Threshold (see ["MINTHRADJ Field"](#)). A zero value means that the calculated Minimum Touch Threshold value is not constrained. The effect of this field is shown in [Figure 5-15](#).

Range: 0 (Minimum Touch Threshold is unconstrained), 1 to 255

BGSCAN Field

SUPPCOEFF: Specifies the supplementary filter coefficient. This value is used to scale the supplementary background-scan noise-line delta filter. Larger values for this coefficient result in a more aggressive response to noise measurements. This value is ignored when BGSCAN SUPPFRQHOPEN is set to 0. The range is 1 to 7, where a value of 1 yields the fastest response and 7 yields the slowest. The default value of 0 means 5.

SUPPCOEFF Range: 0 (5), 1 to 7

SUPPFRQHOPEN: Enables supplementary filter frequency hopping. When this bit is set, the supplementary background-scan noise-line delta filter will be utilized. This is used to improve decision performance when frequency hopping.

RATIO: Specifies the in-touch background scan rate. This is the rate at which background scan measurements are inserted into the acquisition cycle during a continuous touch. The range for this control is 0 to 15, where the default value of 0 means 10.

NOTE This field may be overridden by the TCH control in the BGNLRATIOSTAB, BGNLRATIONOIS or BGNLRATIOVNOI fields (see ["BGNLRATIOSTAB, BGNLRATIONOIS and BGNLRATIOVNOI Fields"](#)).

RATIO Range: 0 (10) to 15

NLGAINSINGX Field

This field specifies the Noise Lines Gain for when Dual X is not used. Note that this is the gain for use when making noise line measurements; the gain for touch measurements is set by the Multiple Touch Touchscreen T100 object.

GAIN: The noise line gain to be used.

Range: 0 to 22

BLKNLTHR Field

This field specifies the Block Noise Lines Threshold (in 8-bit deltas). Touch processing is blocked whenever the measured unfiltered noise level exceeds the limit defined by this value. The default value of 0 means this feature is disabled.

Range: 0 (disabled), 1 to 255 (8-bit deltas)

CFG3 Field

NUMPRESETS: Specifies the Number of Frequency Presets to use for background scans. Up to five preset frequencies are usable for background scans. Note that if NUMPRESETS is set to 1, no background scanning is performed and frequency hopping is disabled.

Range: 0 (5), 1 (no background scans), 2 to 5

DISNLTHRHOP: Disables internal analysis of the Noise Lines Threshold as a means to determine frequency hopping. If this bit is set to 0 (default), frequency hopping is determined by the conditions listed in [Section 5.8.1.4 "Frequency Hopping"](#). If this bit is set to 1, Continuous Frequency Hopping is enabled. In this case, frequency hops can occur only as a consequence of comparing the measured noise against the NOTCHMINDIFF, TCHMINDIFF, NOTCHMINHOP and TCHMINHOP fields.

DISNLTHRSCHG: Disables the internally-managed NLTHR functionality from being used to invoke noise state promotions (see [Section 5.8.1.3 "Transitions Between Noise States"](#)).

When this bit is set to 1, transitions between states occur only according to the first circumstance listed in [Section 5.8.1.3 "Transitions Between Noise States"](#) (that is, when the thresholds are passed). In this case (and dependent on DISNLTHRHOP), the NLTHR functionality may only be used to manage frequency hopping, making configuration much simpler.

TCHHOFFEN: Enables Touch Hold-off (see [Section 5.8.1.6 "Background Scanning and Touch Hold-off Qualification"](#)). If this bit is set 1, the IIR filters are held off from being updated until touch status (in-touch or no-touch) is consistent over time. This option improves filter quality during touch and no-touch transitions, but does so at the potential expense of frequency-hopping responsiveness.

OPTBGSCANEN: Indicates that first-touch response is to be optimized for noise detection.

If this bit is set to 1, the background scan rate adopted during the idle timeout period will be the in-touch background scan rate (as configured in the BGSCAN field RATIO control). This ensures the optimal acquisition duration.

If this bit is set to 0, the background-scan rate adopted during the idle timeout period will be that which is used during no-touch detection. This ensures the best noise determination.

STABCTRL, NOISCTRL and VNOICTRL Fields

These fields are used to configure specific operation settings for the appropriate noise states.

NOTCHTHR (STABCTRL only): Indicates how the Minimum Touch Threshold is to be determined in the STABLE state. If this bit is set to 1, the Minimum Touch Threshold is set to zero. If this bit is set to 0, the Minimum Touch Threshold is calculated.

AUTO (NOISCTRL and VNOICTRL only): Controls the transitions into and out of the NOISY/VERY_NOISY state. Specifically, this bit selects whether transitions between the states can be made based on detected noise levels. If this bit is set to 1, the transitions are influenced by the detected noise levels. If this bit is set to 0, the detected noise levels are ignored. In the case of the VERY_NOISY state (VNOICTRL field), this bit effectively enables the use of the VERY_NOISY state.

When this bit is set to 1, the transition up from the STABLE to NOISY state, or the NOISY to VERY_NOISY state, is made using one of the following conditions:

- Number of frequency hops requested > HOPCNT within HOPCNTPER period

OR

- Detected noise \geq STABHIGHNLTHR/NOISHIGHNLTHR

The transition down from the VERY_NOISY to NOISY state, or the NOISY to STABLE state, is made using both the following conditions:

- Detected noise \leq VNOILOWNLTHR/NOISLOWNLTHR for the last LOWNOISTCH cycles that had a touch detected

OR

- No touch has been detected for NOTCH cycles

Note: LOWNOISTCH and NOTCH in the above are in the NOISCNT and VNOICNT fields.

DUALXMODE: Used to specify the conditions for Dual X operation in the appropriate noise states. If this bit is set to 1, Dual X operation is enabled. If this bit is set to 0, Dual X operation is disabled.

NOTE The DUALXMODE settings for each of the noise states can be set independently of each other only if Acquisition Configuration T8 REFMODE is set to 1 (see [“REFMODE Field”](#)). If REFMODE is set to 0, then a configuration error will be generated if the DUALXMODE settings for STABCTRL, NOISCTRL, and VNOICTRL are not identical.

CAL: If this bit is set to 1, it forces a calibration on entry to the state. Note, however, that if this bit is set to 0, it may be overridden and a calibration still performed under certain circumstances (regardless of the setting of this bit).

NOTE Use of this control is not recommended.

STABFREQ[], NOISFREQ[] and VNOIFREQ[] Fields

These fields specify the set of five candidate frequencies that the noise suppression algorithm can select when operating in a particular state.

Range: 0 to 255

STABTCHAPX[], NOISTCHAPX[] and VNOITCHAPX[] Fields

These fields specify the set of five candidate ADCs per X settings that the noise suppression algorithm can select when operating in the appropriate noise states when touch has been detected (see [Section 5.8.1 “Introduction”](#)).

Range: 0 (255), 1 to 255

STABNOTCHAPX[], NOISNOTCHAPX[] and VNOINOTCHAPX[] Fields

These fields specify the set of five candidate ADCs per X settings that the noise suppression algorithm can select when operating in the appropriate noise states when no touch has been detected (see [Section 5.8.1 “Introduction”](#)).

Range: 0 (255), 1 to 255

NOISLOWNLTHR and VNOILOWNLTHR Fields

These fields define the noise threshold (in 8-bit deltas) below which the noise is identified as too low for the current state. When the noise level falls below VNOILOWNLTHR, the noise suppression algorithm enters NOISY state and when the noise level falls below NOISLOWNLTHR, the noise suppression algorithm enters STABLE state.

The range for these fields is 0 to 255, where a value of 0 sets the threshold to the falling noise lines threshold.

Range: 0 (falling noise lines threshold), 1 to 255 (8-bit deltas)

STABHIGHNLTHR and NOISHIGHNLTHR Fields

These fields define the noise threshold (in 8-bit deltas) above which the noise is identified as too high for the current state. When the noise level rises above STABHIGHNLTHR, the noise suppression algorithm enters the NOISY state and when the noise level rises above NOISHIGHNLTHR, the noise suppression algorithm enters the VERY_NOISY state.

The range for these fields is 0 to 255, where a value of 0 sets the threshold to 75% of the noise lines threshold.

Range: 0 (75% of noise line threshold), 1 to 255 (8-bit deltas)

NOISCNT and VNOICNT Fields

These fields provide real-time counters that are used in the NOISY and VERY_NOISY states to allow demotion to the NOISY or STABLE state, as appropriate.

NOTCH: Specifies the number of “no touch” cycles used to identify a low noise condition. This value is specified in units of 100 cycles, where a value of 0 means 1.

NOTCH Range: 0 (1), 1 to 15 (number of cycles in units of 100 cycles)

LOWNOISTCH: Specifies the number of “low noise touched” cycles used to identify a low noise condition. This field is specified in units of 10 cycles.

LOWNOISTCH Range: 0 (do not wait for “low noise touched”),
1 to 15 (number of cycles in units of 10 cycles)

NOTCHMINDIFF Field

This field specifies the No Touch Minimum Differential (in 8-bit deltas). This is the absolute extent between the noisiest and quietest value for the IIR filters, when no touch is being detected, below which the algorithm deems that the data does not yield a clear candidate for frequency hopping selection.

If the range is less than this value, no frequency hopping will occur.

When the range is greater than or equal to the value, the frequency selection algorithm will make use of filters to determine a frequency hopping candidate.

A value of 0 means that effectively no range qualification takes place.

Effectively, the higher the value specified, the more distinct the noise measured for the preset candidates has to be before frequency hopping can occur.

In practical terms, this field is used to reduce excessive hopping between two similarly quiet candidates. This is potentially problematic if the object is also using HOPST and HOPCNT to manage hopping, whereby excessive hops that yield little or no gain would be counter-productive.

Range: 0 (no differential restriction), 1 to 255 (8-bit deltas)

TCHMINDIFF Field

This field specifies the Touch Minimum Differential (in 8-bit deltas). This is the absolute extent between the noisiest and quietest value for the IIR filters, when a touch is being detected, below which the algorithm deems that the data does not yield a clear candidate for frequency hopping selection.

If this range is less than the specified value, no frequency hopping will occur.

When the range is greater than or equal to the specified value, the frequency selection algorithm will make use of filters to determine a frequency hopping candidate.

A value of 0 means that effectively no range qualification takes place.

Effectively, the higher the value specified, the more distinct the noise measured for the preset candidates has to be before frequency hopping can occur.

In practical terms, this field is used to ensure that no hopping will occur until the IIR filters indicate that there is enough of a range for a suitable candidate to be selected. This prevents superfluous frequency hopping from taking place in low noise conditions.

If no differential restriction is applied, then frequency hopping will occur even if measured noise is very similar for all the preset candidates. This may not be beneficial if the nature of the noise is highly transient.

Range: 0 (no differential restriction), 1 to 255 (8-bit deltas)

NOTCHMINHOP Field

This field specifies the No Touch Minimum Hop (in 8-bit deltas). This is the reduction required between the measured noise associated with the currently selected frequency and the measured noise of the proposed (quieter) frequency hopping candidate, when no touch is being detected, that is permitted for the hop to occur.

If this filter differential is less than the specified value, no frequency hopping will occur.

When the filter differential is greater than or equal to the specified value, the selected quieter candidate is deemed to yield a more suitable configuration, and so a frequency hop will occur in accordance with other hopping configuration limitations.

A value of 0 means that effectively no range qualification takes place.

In practical terms, this is used to reduce excessive frequency hopping between two similarly quiet candidates. This is potentially problematic if the object is also using HOPST and HOPCNT to manage hopping (see “HOPST Field” and “HOPCNT Field”), whereby excessive hops that yield little or no gain would be counter-productive.

Range: 0 (no differential restriction), 1 to 255 (8-bit deltas)

TCHMINHOP Field

This field specifies the Touch Minimum Hop (in 8-bit deltas). This is the minimum reduction required between the measured noise associates with the currently utilized preset configuration, and the quieter measured noise of the proposed frequency-hop candidate, when a touch is being detected, that is permitted for the hop to occur.

If this filter differential is less than the specified value, no frequency hopping will occur.

When the filter differential is greater than or equal to the specified value, the selected quieter candidate is deemed to yield a more suitable configuration, and so a frequency hop will occur.

In practical terms, this field is used to reduce excessive frequency hopping between two similarly quiet candidates. This is potentially problematic if the object is also using HOPST and HOPCNT to manage hopping (see [“HOPST Field”](#) and [“HOPCNT Field”](#)), whereby excessive hops that yield little or no gain would be counter-productive.

Range: 0 (no differential restriction), 1 to 255 (8-bit deltas)

BGNLRATIOSTAB, BGNLRATIONOIS and BGNLRATIOVNOI Fields

These fields specify the Background Noise Line Ratio configuration in the STABLE, NOISY and VERY_NOISY states.

NOTCH: Configures the background noise line ratio in the untouched state. The ratio is specified in units of 4 cycles, where the default value of 0 means perform a scan every cycle.

NOTCH Range: 0 (every cycle), 1 to 15 (4 to 60 cycles) (4 cycle increments)

TCH: Configures the background noise line ratio in the touched state. A non-zero setting overrides the BGSCAN RATIO setting (see [“BGSCAN Field”](#)). The ratio is specified in units of 2 cycles, where the default value of 0 means use the BGSCAN RATIO setting.

TCH Range: 0 (use BGSCAN RATIO), 1 to 15 (2 to 30 cycles) (2 cycle increments)

NLRATIO Field

This field specifies the Noise Line Ratio (that is, the noise line scan rates) for the STABLE state and non STABLE states (that is, NOISY and VERY_NOISY states).

STAB: Configures the noise line ratio in the STABLE state. The ratio is specified in units of 4 cycles, where the default value of 0 means perform a scan every cycle.

STAB Range: 0 (every cycle), 1 to 15 (4 to 60 cycles) (4 cycle increments)

NOTSTAB: Configures the noise line ratio in all other states (that is, NOISY and VERY_NOISY states). The ratio is specified in units of 2 cycles, where the default value of 0 means perform a scan every cycle.

NOTSTAB Range: 0 (use every cycle), 1 to 15 (2 to 30 cycles) (2 cycle increments)

5.8.3 CONFIGURATION CHECKS

The Noise Suppression T72 object causes a configuration check to be performed in the following circumstances:

- When the object is enabled (that is, the ENABLE bit is set in the CTRL field)
- When certain fields, including the CTRL field, are changed

In addition, some fields will cause an automatic recalibration to be performed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)), and the device halts until the error has been corrected. To fix the error, the object settings should be checked to verify that they are all within their allowed limits, as stated in the field descriptions.

If a configuration check occurs, some of the Noise Suppression T72 object's processing is reset. Note that this happens whenever any object causes a configuration check, not just the Noise Suppression T72 object.

**TABLE 5-27: CONFIGURATION CHECKS FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes ⁽¹⁾	None
CALCFG1	Yes	Yes	None
CFG1	Yes	No	None
CFG2	Yes	No	None
HOPCNT	Yes	No	None
HOPCNTPER	Yes	No	None
HOPEVALTO	Yes	No	None
HOPST	Yes	No	None
NLGAINDUALX	Yes	Yes	None
MINNLTHR	Yes	No	None
INCNLTHR	No	No	None
FALLNLTHR	Yes	No	None
NLTHRMARGIN	No	No	None
MINTHRADJ	No	No	None
NLTHRLIMIT	No	No	None
BGSCAN	Yes	No	None
NLGAINSINGX	Yes	Yes	None
BLKNLTHR	No	No	None
CFG3	Yes	No	None
STABCTRL	Yes	No	Error if Acquisition Configuration T8 REFMODE is 0, and DUALXMODE is not equal to NOISCTRL DUALXMODE and VNOICTRL DUALXMODE
STABFREQ[]	No	No	None
STABTCHAPX[]	No	No	None
STABNOTCHAPX[]	No	No	None
STABHIGHNLTHR	No	No	None
NOISCTRL	Yes	No	Error if Acquisition Configuration T8 REFMODE is 0, and DUALXMODE is not equal to STABCTRL DUALXMODE and VNOICTRL DUALXMODE
NOISFREQ[]	No	No	None
NOISTCHAPX[]	No	No	None
NOISNOTCHAPX[]	No	No	None
NOISLOWNLTHR	No	No	None
NOISHIGHNLTHR	No	No	None
NOISCNT	No	No	None
VNOICTRL	Yes	No	Error if Acquisition Configuration T8 REFMODE is 0, and DUALXMODE is not equal to STABCTRL DUALXMODE and NOISCTRL DUALXMODE
VNOIFREQ[]	No	No	None

**TABLE 5-27: CONFIGURATION CHECKS FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
VNOITCHAPX[]	No	No	None
VNOINOTCHAPX[]	No	No	None
CFG4	Yes	No	None
NOTCHMINDIFF	No	No	None
TCHMINDIFF	No	No	None
NOTCHMINHOP	No	No	None
TCHMINHOP	No	No	None
BGNLRATIOSTAB	No	No	None
BGNLRATIONOIS	No	No	None
BGNLRATIOVNOI	No	No	None
NLRATIO	No	No	None

Note 1: If the ENABLE bit is toggled on or off.

5.8.4 MESSAGES

The message data for the Noise Suppression T72 object is shown in [Table 5-28](#).

**TABLE 5-28: MESSAGE DATA FOR NOISE SUPPRESSION T72
(PROCG_NOISESUPPRESSION_T72)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS1	Reserved	DUALXCHG	NLTHRCHG	NOISELVLCHG	STATECHG	ACQCHG	TCH	BLKTCH
2	STATUS2	ACQINDEX			Reserved	DUALX	STATE		
3	MINTCHTHR	Current Minimum Touch Threshold							
4	PKNOISELVL	Peak measured noise level							
5	NOISELVL	Current measured noise level							
6	NLTHR	Current noise lines threshold							

STATUS1 Field

This field reports the current status of the noise suppression algorithm.

BLKTCH: Indicates noise suppression algorithm has generated request to block touch processing as it cannot suppress the effects of noise.

TCH: Indicates that a touch is present. This bit is set to 1 if a touch is present, and is set to 0 if no touch is present.

ACQCHG: Indicates that the noise suppression algorithm has changed the number of ADCs per X at least once since the previous message. This bit is set to 1 if the number of ADCs per X has changed, and 0 otherwise.

STATECHG: Indicates that the noise suppression algorithm has changed its state at least once since the previous message. This bit is set to 1 if the state has changed, and 0 otherwise.

The details of the state change are reported in the STATE field (see below).

NOISELVLCHG: Indicates that the measured noise level has changed at least once since the previous message. This bit is set to 1 if the measured noise level has changed, and 0 otherwise.

NLTHRCHG: Indicates that the noise level threshold has changed at least once since the previous message. This bit is set to 1 if the noise level threshold has changed, and 0 otherwise.

DUALXCHG: Indicates the Dual X operational state has changed at least once since the previous message. This bit is set to 1 if the Dual X operational state has changed, and 0 otherwise.

STATUS2 Field

This field reports the current status of the noise suppression algorithm.

STATE: Indicates the current noise suppression algorithm state selected (see [Table 5-29](#)).

TABLE 5-29: NOISE SUPPRESSION ALGORITHM STATES

Value	State
0	Reserved
1	OFF
2	STABLE
3	NOISY
4	VERY_NOISY
5 to 7	Reserved

DUALX: Indicates that Dual X is operational.

ACQINDEX: Indicates the index of the current ADCs per X settings and frequency settings in use (that is, an index into the ADCSPERX and frequency byte arrays between bytes 21 and 35, 41 and 55, and 61 and 75).

MINTCHTHR Field

This field reports the current Minimum Touch Threshold.

PKNOISELVL Field

This field reports the peak measured noise level since the last message was dispatched. This noise level is the value of measured noise that is assessed against:

- STABHIGHNLTHR and NOISHIGHNLTHR, to invoke noise state promotion
- NOISLOWNLTHR and VNOILOWNLTHR to invoke noise state demotion
- BLKNLTHR to invoke blocking of touch processing under high noise levels

NOISELVL Field

This field reports the current measured noise level.

NOTE The actual algorithm used to measure and calculate this value may vary depending on the firmware version and the device. It is therefore important that no meaning is attached to the number reported here, other than that a higher number means a higher measured noise level, a lower number means a lower measured noise level, and a zero means the noise level is not measured.

NLTHR Field

This field reports the current noise lines threshold.

5.9 Glove Detection T78 Object

5.9.1 INTRODUCTION

A Glove Detection T78 object processes the measurement data received from a particular instance of a Multiple Touch Touchscreen T100 object. It allows touches that are above Multiple Touch Touchscreen T100 INTTHR to be classified as potential gloved touches. Any such touch can be reported in the Multiple Touch Touchscreen T100 messages either as a GLOVE event or as a normal finger touch, as selected by the GLOVERPTEN bit in the CTRL field.

When a Glove Detection T78 object is enabled, it has two modes of operation:

- Normal Mode – The Glove Detection T78 object applies vigorous glove classification to small signal touches to minimize the effect of unintentional hovering finger reporting. The EDGESUP and LOCKEDGESUP bits in the CTRL field and the MINAREA, CONFTHR and MINDIST configuration fields are applied only when in Normal Mode. Once a gloved touch is found, the Glove Detection T78 object enters Glove Confidence Mode.
- Glove Confidence Mode – The Glove Detection T78 object expects the user to be wearing gloves so the classification process is much less stringent. Additionally, when in this mode, the Key Array T15 object will also respond to small signal touches.

The Glove Detection T78 object remains in Glove Confidence Mode until any of the following are true:

- The timer controlled by GLOVEMODETO expires.
- The system is reset (for example, this might be done when the touchscreen on the phone or tablet is locked).
- A finger touch is present on the touchscreen. That is, the touch has been above Multiple Touch Touchscreen T100 TCHTHR but has not been below (TCHTHR – TCHHYST).

If the system supports self capacitance measurements (see [Section 3.5.2 “Configuration”](#) and [Appendix A “Measurement Processing on mXT144U”](#)), then the glove touch signals which would not usually be detected by Multiple Touch Touchscreen T100 may be reported using the self capacitance touch measurement by setting SCTGLOVETHR. Touches must be above SCTGLOVETHR in both Normal Mode and Glove Confidence Mode. In Normal Mode all usual classification stages must be passed for self capacitance glove touches, with the exception of DISCRIMTHR.

Hysteresis is provided for the SCTGLOVETHR field. For a glove to enter detection the touch delta must be greater than or equal to SCTGLOVETHR. For a glove to leave detection the touch delta must be less than (SCTGLOVETHR – SCTGLOVEHYST).

Note that when glove detections are being tracked in the Multiple Touch Touchscreen T100 object, mutual capacitance scans are performed each cycle, even if the device is using self capacitance measurements as the primary measurement mechanism (see [Appendix A “Measurement Processing on mXT144U”](#)).

5.9.2 CONFIGURATION

**TABLE 5-30: CONFIGURATION FOR GLOVE DETECTION T78
(PROCI_GLOVEDETECTION_T78)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0					
0	CTRL	GLOVERPTEN	DBGDISCRIMEN	Reserved		LOCKEDGESUP	EDGESUP	Reserved	ENABLE					
1	MINAREA	Reserved			Minimum contact diameter									
2	CONFTHR	Reserved		Confidence threshold classification cycles										
3	MINDIST	Reserved			Minimum distance before classification									
4	GLOVEMODETO	Glove confidence mode timeout												
5	SUPTO	Suppression timeout												
6	SYNCSPERX	ADC sets per X												
7	Reserved	Reserved												
8	DISCRIMTHR	Glove discriminant threshold												
9	SCTGLOVETHR	Self capacitance touch glove threshold												
10	SCTGLOVEHYST	Reserved	Self-capacitance touch glove hysteresis											
11	SCTMINAREA	Reserved			SCTMINAREA									

CTRL Field

This field enables or disables the Glove Detection T78 object.

ENABLE: Enables the use of this Glove Detection T78 object. The object is enabled if set to 1, and disabled if set to 0.

EDGESUP: Enables the edge suppression mechanism. When this bit is set to 1, small signal touches that enter the touchscreen from an edge sensor line cannot be classified as glove touches until they move away from the edge.

LOCKEDGESUP: Enables the edge suppression locking mode. When this bit is set to 1, small signal touches that were suppressed by EDGESUP remain suppressed as they move away from the edge.

DBGDISCRIMEN: Enables the outputting of the glove discriminant. Only the discriminant for a single glove touch can be output using this control. When more than one touch is present, the discriminant that is output via this control could correspond to any touch.

NOTE This bit is for use only during design, under the guidance of Microchip field engineers, to help tune the product for characterization. It should not be used in production. This control should be left at its default value of 0 unless advised otherwise by Microchip.

GLOVERPTEN: If this bit is set to 1, glove touches are reported as a GLOVE event in the Multiple Touch Touchscreen T100 messages. Otherwise, if this bit is set to 0, glove touches are reported as FINGER events. See [Section 4.3.4 "Messages"](#) for more information on touch reporting.

MINAREA Field

This field specifies the Minimum Contact Area in nodes for a small signal touch before it can be considered as a glove touch. The contact area is defined as the number of nodes above Multiple Touch Touchscreen T100 (INTTHR – INTTHRHYST). The MINAREA field can help to distinguish between glove touches and hovering fingers, especially when used in conjunction with CONFTHR.

Range: 0 to 31

CONFTHR Field

This field specifies the Confidence Threshold. This is the number of cycles a touch must consistently be considered as a glove touch before it can be classified.

Range: 0 to 63

MINDIST Field

This field specifies the Minimum Distance a small signal touch must move from its touchdown position before it can be classified as a glove touch. The units for this field is one tenth of a node, such that 255 represents 25.5 nodes.

Range: 0 to 255 (nodes / 10)

GLOVEMODETO Field

This field specifies the Glove Confidence Mode Timeout. This is the time that the system remains in Glove Confidence Mode after a gloved touch is removed from the screen. The timer is reset if a finger touch is present on the sensor before the timeout has expired. The units for this field is 200 ms.

A value of 0 means an infinite timeout. In this case, the system remains in Glove Confidence Mode indefinitely until an action takes it out of Glove Confidence Mode (that is, the system is reset or a finger touch is present).

A value of 255 means an instant timeout such that the system leaves Glove Confidence Mode immediately after a gloved touch is removed from the touchscreen.

Range: 0 (Infinite), 1 to 254 (timeout in 200 ms increments), 255 (Instant)

SUPTO Field

This field specifies a Suppression Timeout. This is the length of time after finger touches are present on the screen before small signal touches can be classified as glove touches. This field is useful, for example, in preventing hovering finger touches from being classified as gloved touches and reported during fast finger taps on the screen while typing. When this setting is enabled, glove touches cannot be classified while fingers are on the touchscreen. The unit for this field is 200 ms, where a value of 0 disables the Suppression Timeout.

Range: 0 (Off) to 255 (timeout in 200 ms increments)

SYNCSPERX Field

This field specifies the minimum ADC sets per X line to be used when glove touches are being detected subject to other considerations. The actual ADCs per X line value used is the maximum of this SYNCSPERX field (if in detect) and the equivalent values in other objects (see [Section 6.6.1 "Introduction"](#) for details).

Range: 1 to 255 (number of ADCs)

DISCRIMTHR Field

This field specifies the Glove Discriminant Threshold. This enables the use of a discriminant based classification threshold for potential glove touches. Note that the value specified by this field applies only to glove touches detected using the mutual capacitance measurement.

A higher value in this field will make it harder for thin or small glove touches to classify, but will decrease the likelihood of false hovering finger detections. A lower value will give less improvement for hovering finger rejection, but will not affect the classification of thinner gloves.

Range: 0 (Off), 1 to 255

SCTGLOVETHR Field

This field specifies the Self Capacitance Touch Glove Threshold in 8 bit delta counts.

If the system supports self capacitance measurements (see ["MEASALLOW Field"](#)) and [Appendix A "Measurement Processing on mXT144U"](#), it is possible for touches that are not detected by the mutual capacitance measurement to be eligible for glove detection if the average of their X and Y peak delta \geq SCTGLOVETHR. Note that EDGESUP, LOCKEDEDGESUP, MINAREA, CONFTHR and MINDIST still apply to eligible self capacitance touches.

If this field has a value less than Auxiliary Touch Configuration T104 INTTHR, then it will have no effect.

Using SCTGLOVETHR to detect thick gloves will allow only a single thick glove to report due to the nature of self capacitance measurements. In order to allow multitouch glove operation, gloves must have deltas above Multiple Touch Touchscreen T100 INTTHR.

Range: 0 (Off), 1 to 255

SCTGLOVEHYST Field

This field specifies the Self Capacitance Touch Glove Hysteresis in 8-bit delta counts. This is the level of hysteresis applied to self capacitance glove detections. See [Section 5.9.1 "Introduction"](#) for more details.

The value is limited to (SCTGLOVETHR / 2); see ["SCTGLOVETHR Field"](#). If (SCTGLOVETHR – SCTGLOVEHYST) is less than Auxiliary Touch Configuration T104 (INTTHR – INTTHRHYST), then the glove touch will leave detect at Auxiliary Touch Configuration T104 (INTTHR – INTTHRHYST). See [Section 5.9.1 "Introduction"](#) for more details.

Range: 0 to (SCTGLOVETHR / 2)

SCTMINAREA Field

This field specifies the Self-Cap Touch Minimum Contact Area. This is the minimum contact area in nodes for self capacitance glove touches before they can be considered as glove touches. The contact area is defined as the number of X lines above Auxiliary Touch Configuration T104 (XINTTHR – XINTHYST) plus the number of Y lines above Auxiliary Touch Configuration T104 + (YINTTHR – YINTHYST).

Range: 0 to 31

5.9.3 CONFIGURATION CHECKS

A Glove Detection T78 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

**TABLE 5-31: CONFIGURATION CHECKS FOR GLOVE DETECTION T78
(PROCI_GLOVEDETECTION_T78)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes	No	None
MINAREA	No	No	None
CONFTHR	No	No	None
MINDIST	Yes	No	None
GLOVEMODETO	No	No	None
SUPTO	No	No	None
SYNCSPERX	No	No	None
DSCRIMTHR	No	No	None
SCTGLOVETHR	No	No	None
SCTGLOVEHYST	No	No	None
SCTMINAREA	No	No	None

5.9.4 MESSAGES

No messages are generated directly for the Glove Detection T78 object. Glove touches are reported through the reporting mechanisms of the linked Multiple Touch Touchscreen T100 object (see [Section 4.3.4 “Messages”](#)).

5.10 Retransmission Compensation T80 Object

5.10.1 INTRODUCTION

The Retransmission Compensation T80 object limits the undesirable effects on the mutual capacitance touch signals caused by poor device coupling to ground, such as poor sensitivity and touch break-up. The user can configure this object to allow the linked Multiple Touch Touchscreen T100 object to compensate for signal degradation due to these effects. If self-capacitance measurements are also scheduled, the Retransmission Compensation T80 object will use the resultant data to enhance the compensation process.

Note that this object will only process signals that are above the low tracking threshold of the linked Multiple Touch Touchscreen T100. This threshold should be configured accordingly.

The Retransmission Compensation T80 object is also capable of compensating for water presence on the sensor if self capacitance measurements are scheduled. In this case, both mutual capacitance and self capacitance measurements are used to detect moisture. Once moisture is touched, the system goes into single touch mode (that is, self capacitance measurements are used to detect single touches in the presence of moisture). A recalibration can also be performed in self capacitance mode once calibrated-in moisture is detected, if this is configured. See [“MOISTCFG Field”](#) for more information.

The TCHIMEASMOV control in the MOISTCFG3 field (see [“MOISTCFG3 Field”](#)) allows for the execution of additional touch dilation to account for inter-measurement movement (that is, potential variation in position between the different types of measurements performed within a single cycle).

The MOISTVLDTHRSF field (see [“MOISTVLDTHRSF Field”](#)) configures a self capacitance dynamic threshold that is aimed at finger touches under very noisy conditions. When dynamic thresholding is enabled, touch dilation is effectively reduced.

The behaviors associated with TCHIMEASMOV and MOISTVLDTHRSF are independent and can counteract one another to some degree if these controls are used together. Care should be taken to apply them appropriately, based upon touch type and noise state.

If either Self Capacitance Global Configuration T109 DISMEASX or DISMEASY are set to 1 (that is, a particular axis has been disabled) due to one axis of self capacitance data not being measurable, then moisture compensation will not run. Note, however, that retransmission compensation will use mutual capacitance data only.

5.10.2 CONFIGURATION

**TABLE 5-32: CONFIGURATION FOR RETRANSMISSION COMPENSATION T80
(PROCI_RETRANSMISSIONCOMPENSATION_T80)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved		DISATCHCHECK	DISINVTCHREM	COMPENHEN	DISGAINLIMIT	RPTEN	ENABLE
1	COMPAIN	Compensation gain							
2	TARGETDELTA	Target delta							
3	COMPTHR	Compensation threshold							
4	ATCHTHR	Antitouch threshold							
5	MOISTCFG	MOISTCALEN	MOISTEN	MOISTDI					
6	Reserved	Reserved							
7	MOISTTHR	Moisture threshold							
8	MOISTINVATCHTHR	Moisture invalid antitouch threshold							
9	MOISTCFG2	MOISTCALMODE	MOISTFIREN	Reserved		MOISTDETINT			
10	COMPSTRTHR	Compensation strength threshold							
11	COMPCFG	Reserved				BNDGBOXTHR			
12	MOISTVLDTHRSF	Moisture validation threshold scale factor							
13	MOISTCFG3	Reserved					TCHIMEASMOV		

CTRL Field

ENABLE: Enables or disables the use of the Retransmission Compensation T80 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

DISGAINLIMIT: Disables gain limiting based on TARGETDELTA when using mutual data for compensation.

NOTE This bit applies only when using mutual capacitance only retransmission compensation.

COMPENHEN: Enables Compensation Enhancement.

NOTE This bit applies only when using mutual capacitance only retransmission compensation.

If this bit is set to 1, retransmission compensation that is using mutual data is enhanced such that compensated touches are made larger (at the possible expense of pinch separation). When this control is enabled to improve large touch performance, COMPGAIN should be reduced to the minimum value required to achieve the desired performance.

DISINVTCHREM: Disables Invalid Touch Removal.

The Retransmission Compensation T80 object identifies regions of potential moisture on a touchscreen from the self capacitance and mutual capacitance measurements. By default the Retransmission Compensation T80 object removes invalid nodes (that is nodes identified as having moisture present). If DISINVTCHREM is set to 1, invalid touches are not removed. This allows small touches to be detected in the presence of moisture.

DISATCHCHECK: If this bit is set to 1, it disables a check for the presence of antitouch to indicate that a touch should be recovered by retransmission compensation (see ["ATCHTHR Field"](#)).

COMPGAIN Field

When mutual capacitance measurements only are allowed, this field specifies the Compensation Gain. This is the strength of the applied compensation when the device is poorly coupled to the user. This field should be tuned based on the screen type, panel thickness, and so on.

If self capacitance measurements are enabled, this field controls whether or not compensation is applied. If this field is set to 0 (default), no compensation is applied and the Retransmission Compensation T80 object provides moisture compensation only. If this field is set to a non zero value, compensation is applied according to TARGETDELTA (a non-zero value effectively has no meaning).

Range (mutual capacitance only): 0 to 255

Range (self capacitance enabled): 0 (no compensation applied),
>0 (compensation applied according to TARGETDELTA)

Typical (mutual capacitance only): 40

TARGETDELTA Field

This field specifies the Target Delta in 8-bit deltas. This is the desired mutual capacitance touch delta for a fully grounded touch. It is used to calculate the coupling level for the mutual capacitance compensation algorithm. It is also used to limit the gain for both the mutual and self capacitance versions of the algorithm.

Range: 0 to 255 (8-bit deltas)

Typical: 125

COMPTHR Field

This field specifies the Compensation Threshold in 8-bit deltas. If the antitouch exceeds ATCHTHR and the compensated delta count reaches the Compensation Threshold, compensation is applied. When this field is set to 0, all compensated nodes are modified.

Range: 0 to 255

Typical: 50 to 80% of TARGETDELTA

ATCHTHR Field

This field specifies the Antitouch Threshold in 8-bit deltas. This threshold specifies the negative peak antitouch in mutual capacitance deltas that must be breached for retransmission compensation to take effect. When this control is disabled, retransmission compensation will run regardless of antitouch presence (as long as other criteria for compensation are met).

<p>NOTE A retransmitting touch typically incurs some antitouch deltas, whereas a well-coupled, low-signal touch will not. The objective of the Retransmission Compensation T80 object, therefore, is to compensate retransmitting touches but not well-coupled, low-signal touches.</p>
--

A typical setting for this control is just below the negative noise floor (that is, equivalent of INTTHR of the associated Multiple Touch Touchscreen T100), but this field may require some tuning depending on the application and the other processing objects used. A value of 0 disables this feature.

Range: 0 (disabled), 1 to 255

MOISTCFG Field

MOISTDI: Specifies the Moisture Detection Integration. This is the number of extra mutual frames requested to confirm the persistent invalid touch present before calling calibration.

It is recommended that this functionality is enabled by setting MOISTDI to a value. If MOISTDI is left at the default value of 0 (disabled), calibrated-in moisture may not be discriminated from noise and moisture compensation will be degraded.

Note that MOISTCALEN must be enabled for this feature to have an effect.

MOISTDI Range: 0 (disabled), 1 to 63 (mutual frames)

MOISTEN: Enables moisture compensation. If this bit is set to 1, the Retransmission Compensation T80 object removes noise due to moisture so that the moisture is not detected as touches.

Note that this feature will run only if self capacitance measurement is configured and enabled (see [“MEASALLOW Field”](#) and [Section 6.11 “Auxiliary Touch Configuration T104 Object”](#)).

MOISTCALEN: Enables recalibration when persistent invalid touches are present (that is, in regions where moisture has been calibrated in).

Note that this feature will run only if self capacitance measurement is configured and enabled (see [“MEASALLOW Field”](#) and [Section 6.11 “Auxiliary Touch Configuration T104 Object”](#)).

MOISTTHR Field

This field configures the Moisture Threshold (in 8-bit delta counts). If non zero, this threshold defines the amount of antitouch that must be present before nodes are marked as being moisture.

Note that an internal limit of Multiple Touch Touchscreen T100 TCHTHR – TCHHYST is taken as the lower limit for this field. Therefore, if this field is set to a value less than Multiple Touch Touchscreen T100 TCHTHR – TCHHYST then Multiple Touch Touchscreen T100 TCHTHR – TCHHYST is used.

Range: 0 (use Multiple Touch Touchscreen T100 TCHTHR), 1 to 255

MOISTINVATCHTHR Field

This field configures the Moisture Invalid Antitouch Threshold. This is the number of invalid antitouch nodes that can be seen before the touchscreen is marked as degraded. If the number of invalid antitouch nodes is greater than or equal to this threshold, the touchscreen will report only touches above (TCHTHR – TCHHYST). In this case a DEGACTV message is sent (see [“STATUS Field”](#)).

A value of 0 disables this feature.

Range: 0 (disabled), 1 to 255

MOISTCFG2 Field

MOISTDETINT: Configures the Moisture Detection Interval. This is the interval between the periodic requests for mutual scans when self capacitance measurements are the primary measurement type. The interval is specified in units of 200 ms.

Range: 0 (disable), 1 to 15 (0.2 to 3 seconds) (200 ms increments)

Typical: 5

MOISTFIREN: Enables FIR Filtering on SCT Deltas.

If this bit is set to 1, FIR filtering with fixed coefficients is applied to the SCT delta values of both axes prior to further processing.

If this bit is set to 0, raw SCT deltas will be used for normal processing.

MOISTCALMODE: Configures the Moisture Calibration Mode. This controls the calibration behavior for moisture.

If this bit is set to 0, then calibration is triggered only when calibrated-in moisture has been removed.

If this bit is set to 1, then calibration is triggered when moisture is either added or removed.

COMPSTRTHR Field

This field defines the Compensation Strength Threshold. This is the strength of the self capacitance deltas that, when reached, causes touch compensation to be applied in the case where there is not enough antitouches present in the mutual data (that is, ATCHTHR is not exceeded; see ["ATCHTHR Field"](#)). The strength of the touch is evaluated as the peak self capacitance delta divided by the number of nodes above threshold.

This field is provided in order to allow touches to be reported, even though retransmission effects are substantially reducing the level of antitouches present. This is caused when touches are placed on the same line of the axis, or fingers are placed over the positions of antitouch associated with retransmission effects (such as touches on all four corners of a square).

This field also prevents hovering fingers being compensated. This value should be increased when the system is compensating unwanted hovering fingers; it should be decreased when small touches on the same X or Y line need to be compensated.

Range: 0 (disable), 1 to 255 (8-bit deltas / number of nodes above threshold)

Typical: 20

COMPCFG Field

This field configures retransmission compensation.

BNDGBOXTHR: Specifies the Bounding Box Threshold. This is the minimum area (in nodes) that must be reached before retransmission compensation is applied. The area is defined as the total rectangular extent of all deltas exceeding the lower touch threshold for the measurement type (mutual or self capacitance) that is being used to do the compensation (see [Section 4.3 "Multiple Touch Touchscreen T100 Object"](#) and [Section 6.11 "Auxiliary Touch Configuration T104 Object"](#) for more information on thresholds). If the area is equal to or greater than BNDGBOXTHR, retransmission compensation is applied.

For areas below BNDGBOXTHR, compensation is not applied, in order to avoid the incorrect boosting of low amplitude touch deltas (such as, noise spikes).

Range: 0 (10), 1 to 15 (number of nodes)

MOISTVLDTHRSF Field

This field specifies the Moisture Validation Threshold Scale Factor. This is used to modify the threshold that is used to validate mutual capacitance deltas as part of moisture processing. Mutual capacitance data is checked against the self capacitance data for invalid nodes. By default, this happens when the Auxiliary Touch Configuration T104 INTTHR is reached, but the MOISTVLDTHRSF field allows a portion of the peak positive touch delta to be used instead. Use of this field can only increase the threshold, not reduce it. Care is needed to use this field appropriately, as it would be possible to raise the threshold above smaller touches if a larger touch is also present. Generally this field should be triggered in noisy conditions for finger touches (which couple in much of the noise).

Note that MOISTFIREN must be enabled for this feature to work. DISINVTCHREM should also be enabled to maintain the reporting of small touches in the presence of larger touches

The value for this fields is specified in units of 1/256 of the peak positive touch delta, where the default value of 0 means the feature is disabled.

Range: 0 (disabled), 1 to 255

MOISTCFG3 Field

TCHIMEASMOV: Specifies the Inter-Measurement Movement for Touches. This prevents fast-moving touches from potentially being marked as moisture due to the timing between different types of touch measurements on a single cycle. The value should be set to at least the number of lines over which a touch can move between the different types of measurement. Larger values might be required for larger touchscreens.

Note that TCHIMEASMOV controls dilation for small touches only (that is, self capacitance delta values below Auxiliary Touch Configuration T104 TCHTHR). For touches above Auxiliary Touch Configuration T104 TCHTHR no dilation is applied.

The minimum effective value for this control is 2, which accounts for a potential positional difference of at least one node between the different types of measurements within a single cycle. The value is specified as the number of lines plus 1, where the default value of 0 means 2 (that is, a single line).

Range: 0 (2 = 1 line), 1 to 7 (0 to 6 lines) (number of lines + 1)

5.10.3 CONFIGURATION CHECKS

A Retransmission Compensation T80 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)). The Retransmission Compensation T80 halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

TABLE 5-33: CONFIGURATION CHECKS FOR RETRANSMISSION COMPENSATION T80 (PROCI_RETRANSMISSIONCOMPENSATION_T80)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes	No	None
COMPGAIN	No	No	None
TARGETDELTA	No	No	None
COMPTHR	No	No	None
ATCHTHR	No	No	None
MOISTCFG	No	No	None
MOISTTHR	Yes	No	Saturated to T100 TCHTHR – TCHHYST
MOISTINVATCHTHR	No	No	None
MOISTCFG2	No	No	None
COMPSTRTHR	No	No	None
COMPCFG	No	No	None
MOISTVLDTHRSF	No	No	None
MOISTCFG3	No	No	None

5.10.4 MESSAGES

The message data for the Retransmission Compensation T80 object is shown in [Table 5-34](#).

TABLE 5-34: MESSAGE DATA FOR RETRANSMISSION COMPENSATION T80 (PROCI_RETRANSMISSIONCOMPENSATION_T80)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	COMPACTV	Reserved			DEGACTV	TCHMOISTDET	MOISTDET	TCHDET

STATUS Field

This field reports the on a number of states. A new message is generated whenever there is a change of state.

TCHDET: Indicates that moisture processing has detected a touch.

MOISTDET: Indicates that moisture processing has detected moisture.

TCHMOISTDET: Indicates that moisture processing has detected a touch in moisture.

DEGACTV: Indicates that moisture processing is running in degraded mode (see [“MOISTINVATCHTHR Field”](#)).

COMPACTV: Indicates that retransmission compensation is actively reconstructing touches.

5.11 Touch Sequence Processor T93 Object

5.11.1 INTRODUCTION

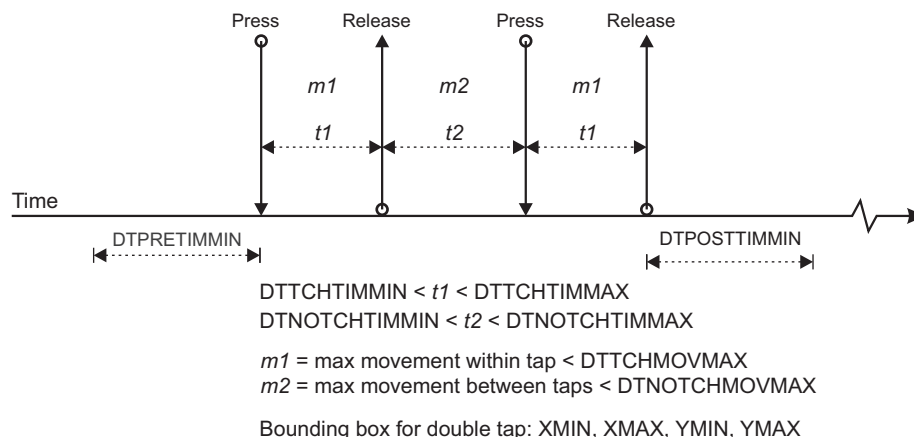
A Touch Sequence Processor T93 object captures a series of touch and release locations. These can be regular finger touches or glove touches. The purpose of this object is to allow a particular pattern of finger touches to be detected and actioned on by the host. Specifically, this object allows for a double tap sequence to be detected, for example to act as a wake-up gesture.

Once the Touch Sequence Processor T93 object has detected a possible double tap sequence, a message is sent to the host. The host can read the message to see the reason for any failures.

5.11.1.1 Double Tap

The fields that control double tap detection are shown in [Figure 5-17](#).

FIGURE 5-17: DOUBLE TAP SEQUENCE



The DTPRETIMMIN and DTPOSTTIMMIN fields are used to specify the amount of time before and after a potential double tap in which there should be no taps. These prevent touch sequences triggering undesired double tap reports.

The DTTCHTIMMIN and DTTCHTIMMAX fields determine the minimum and maximum times allowed between a press and a release.

The DTNOTCHTIMMIN and DTNOTCHTIMMAX fields determine the minimum and maximum times allowed between the end (release) of the first tap and the start (press) of the second tap in a double tap sequence.

Once a double tap sequence has been detected, the sequence detection logic is reset.

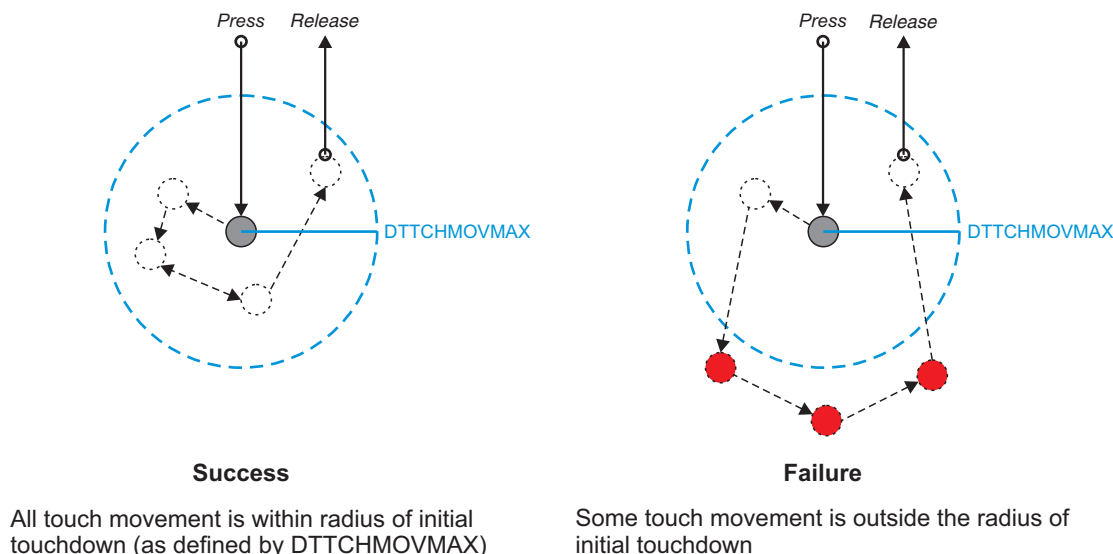
5.11.1.2 Touch Position

A bounding box can be defined to restrict where on the touchscreen a double tap can occur. This is defined by the XMIN, XMAX, YMIN and YMAX fields.

5.11.1.3 Touch Movement

The Touch Sequence Processor T93 object continually monitors a touch to see if it moves outside a specified radius from the touchdown position. This radius is specified using the DTTCHMOVMAX field. If the touch strays outside this radius at any time, then the entire double tap has failed. The effect of using the touch radius is shown in [Figure 5-18](#).

FIGURE 5-18: TOUCH RADIUS



The touches within a double tap can also be constrained to be within a certain distance of each other, as determined by the DTNOTCHMOVMAX field. In addition, the time allowed between the touches within a double tap can also be constrained by the DTTCHMINMIN and DTTCHMINMAX fields.

5.11.2 CONFIGURATION

TABLE 5-35: CONFIGURATION FOR TOUCH SEQUENCE PROCESSOR T93
(PROCI_TOUCHSEQUENCELOGGER_T93)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved				AUTOARMEN	MULTTCHSUPEN	RPTEN	ENABLE
1	XMIN	Low X Bound LSByte							
2		Low X Bound MSByte							
3	XMAX	High X Bound LSByte							
4		High X Bound MSByte							
5	YMIN	Low Y Bound LSByte							
6		Low Y Bound MSByte							
7	YMAX	High Y Bound LSByte							
8		High Y Bound MSByte							
9 – 17	Reserved	Reserved							
18	DTPRETIMMIN	Pre time							
19	DTPOSTTIMMIN	Post time							
20	DTTCHMOVMAX	Maximum movement when touched LSByte							
21		Maximum movement when touched MSByte							
22	DTNOTCHMOVMAX	Maximum movement between release and next touch LSByte							
23		Maximum movement between release and next touch MSByte							
24	DTTCHTIMMIN	Minimum touch time							
25	DTTCHTIMMAX	Maximum touch time							
26	DTNOTCHTIMMIN	Minimum no-touch time							
27	DTNOTCHTIMMAX	Maximum no-touch time							
28	FAILRPTEN	Reserved		PREPOSTTIMMIN	MULTTCH	NOTCHTIM	TCHMOVMAX	NOTCHMOVMAX	Reserved
29		UNHANDLED	Reserved						

CTRL Field

ENABLE: Enables the use of this Touch Sequence Processor T93 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

MULTTCHSUPEN: Enables multiple touch suppression. If this bit is set to 1, the touch sequence is aborted when there is more than one touch simultaneously detected on the touchscreen.

AUTOARMEN: Enables Auto Arm. If this bit is set to 1, the touch sequence logging and double tap detection is carried out continuously.

XMIN, YMIN, XMAX, YMAX Fields

These four fields define a bounding box within which all touch and release events must occur for a double tap sequence to be considered valid. The values for the XMIN, YMIN, XMAX, YMAX Fields are specified in touchscreen pixels, where the default value of 0 for XMAX/YMAX means use Multiple Touch Touchscreen T100 X RANGE/Y RANGE, as appropriate.

XMIN/YMIN Range: 0 to 65535

XMAX/YMAX Range: 0 (use Multiple Touch Touchscreen T100 X RANGE/Y RANGE), 1 to 65535

DTPRETIMMIN Field

This field specifies the Minimum Pre Time for Double Tap Sequence. This is the time for monitoring before a touch. If another touch or release is detected during this period, then this touch is ignored for a double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

DTPOSTTIMMIN Field

This field specifies the Minimum Post Time for Double Tap Sequence. This is the time for monitoring after a touch release. If another touch is detected during this period, then this touch is ignored for a double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

DTTCHMOVMAX Field

This field specifies the Maximum Movement when touched for a Double Tap sequence. This is the maximum amount of movement (in touchscreen pixel counts) that is allowed during one touch. That is, the touch must remain within this distance of the original touchdown for it to be considered part of a valid double tap sequence. See [Section 5.11.1.3 "Touch Movement"](#) for more information.

Range: 0 to 65535 (touchscreen pixels)

Typical: 4095

DTNOTCHMOVMAX Field

This field specifies the Maximum Movement for No Touches (that is, when not touched) for a Double Tap sequence. This is the maximum amount of movement (in touchscreen pixel counts) allowed between a release and the next touch for this to be considered part of a valid double tap sequence.

Range: 0 to 65535 (touchscreen pixels)

Typical: 4095

DTTCHTIMMIN Field

This field defines the Minimum Time between Press and Release for a Double Tap Sequence. This is the minimum amount of time allowed for a touch to be on screen (time between press and release) for this to be considered part of a valid double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

DTTCHTIMMAX Field

This field defines the Maximum Time between Press and Release for a Double Tap Sequence. This is the maximum amount of time allowed for a touch to be on screen (time between press and release) for this to be considered part of a valid double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

Typical: 100 (1.6 seconds)

DTNOTCHTIMMIN Field

This field defines the Minimum Time between Release and next Press for a Double Tap Sequence. This is the minimum amount of time allowed between a release and the next press for this to be considered part of a valid double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

DTNOTCHTIMMAX Field

This field defines the Maximum Time between Release and next Press for a Double Tap Sequence. This is the maximum amount of time allowed between a release and the next press for this to be considered part of a valid double tap sequence. The time is specified in units of 16 ms.

Range: 0 to 255 (0 to 4.08 seconds in 16 ms increments)

Typical: 100 (1.6 seconds)

FAILRPTEN Field

This field controls whether the various Failure Reason reports are enabled in the object's messages (see ["FAILREASON Field"](#)).

NOTCHMOVMAX: Reports a failure if the inter touch distance for double tap has been exceeded. Specifically, a failure is reported if the maximum movement between touches in a double tap sequence is greater than DTNOTCHMOVMAX.

TCHMOVMAX: Reports a failure if the distance of a touch radius is exceeded (See [Section 5.11.1.3 "Touch Movement"](#)). Specifically, a failure is reported if a touch within a double tap sequence moves more than DTTCHMOVMAX from the touchdown point.

NOTCHTIM: Reports a failure if the inter touch timeout for a double tap has been exceeded. Specifically, a failure is reported if the period between the taps within a double tap sequence is greater than DTNOTCHTIMMAX or less than DTNOTCHTIMMIN.

MULTTCH: If MULTTCHSUPEN is set to 1 (see ["CTRL Field"](#)), reports a failure if Multiple Touches are detected on the screen.

PREPOSTTIMMIN: Reports a failure if a touch is detected during the minimum pre or post time periods for double tap sequences, as defined by DTPRETTIMMIN and DTPOSTTIMMIN (see ["DTPRETTIMMIN Field"](#) and ["DTPOSTTIMMIN Field"](#)).

UNHANDLED: Reports a failure if an unhandled failure reason is detected.

5.11.3 CONFIGURATION CHECKS

A Touch Sequence Processor T93 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host via the Command Processor T6 object. The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

**TABLE 5-36: CONFIGURATION CHECKS FOR TOUCH SEQUENCE PROCESSOR T93
(PROCI_TOUCHSEQUENCELOGGER_T93)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	No	None
XMIN	No	No	None
XMAX	No	No	None
YMIN	No	No	None
YMAX	No	No	None
DTPRETIMMIN	No	No	None
DTPOSTTIMMIN	No	No	None
DTTCHMOVMAX	No	No	None
DTNOTCHMOVMAX	No	No	None
DTTCHTIMMIN	Yes	No	Error if DTTCHTIMMIN > DTTCHTIMMAX
DTTCHTIMMAX	Yes	No	Error if DTTCHTIMMAX < DTTCHTIMMIN
DTNOTCHTIMMIN	Yes	No	Error if DTNOTCHTIMMIN > DTNOTCHTIMMAX
DTNOTCHTIMMAX	Yes	No	Error if DTNOTCHTIMMAX < DTNOTCHTIMMIN
FAILRPTEN	No	No	None

Note 1: If the ENABLE bit is toggled on or off.

5.11.4 MESSAGES

The message data for the Touch Sequence Processor T93 object is shown in [Table 5-37](#).

**TABLE 5-37: MESSAGE DATA FOR TOUCH SEQUENCE PROCESSOR T93
(PROCI_TOUCHSEQUENCELOGGER_T93)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved						DBLTAP	Reserved
2	FAILREASON	Reserved				DTFAILREASON			

STATUS Field

DBLTAP: If this bit is set to 1, a double tap has been detected.

FAILREASON Field

This field reports the Failure Reason; that is, it reports the Failure Reason of the last double tap. If this field is non zero, then a failure has occurred.

DTFAILREASON: Reports the Failure Reason code.

Range: see table [Table 5-38](#).

TABLE 5-38: DTFAILREASON FAILURE REASON CODES

Failure Reason Code	FAILRPTEN Bit	Description
0	—	No double tap failure detected
1	NOTCHMOVMAX	Maximum movement between taps > DTNOTCHMOVMAX
2	TCHMOVMAX	Maximum movement during a single tap > DTTCHMOVMAX

TABLE 5-38: DTFAILREASON FAILURE REASON CODES

Failure Reason Code	FAILRPTEN Bit	Description
3	NOTCHTIM	Time between taps > DTNOTCHTIMMAX OR Time between taps < DTNOTCHTIMMIN
4	MULTTCH	Multiple touches detected on screen, but only if MULTTCHSUPEN = 1
5	PREPOSTTIMMIN	Touch detected during DTPRETTIMMIN or DTPOSTTIMMIN
15	UNHANDLED	Unhandled double tap failure reason
All other values	–	Reserved

5.12 Self Capacitance Noise Suppression T108 Object

5.12.1 INTRODUCTION

The Self Capacitance Noise Suppression T108 object provides a noise suppression algorithm to suppress the effects of external noise.

Similar in both design and configuration to the Noise Suppression T72 object, the Self Capacitance Noise Suppression T108 object is the noise suppression interface for self capacitance touch measurements.

Furthermore, to utilize the Self Capacitance Noise Suppression T108 object, the Noise Suppression T72 object must also be enabled and configured, as both objects are essential to managing noise within self capacitance touch measurements.

Such noise can take many forms, including:

- **Internal noise** – due to electrical interferences inherent in the product's architecture (such as LCD display emissions)
- **Environmental noise** – due to exposure to noise within the user's living environment (such as from other electrical equipment in near proximity)
- **Charger noise** – due to noise induced by peripheral equipment used to recharge the product's battery.

5.12.1.1 Noise Notification Mechanisms

The Self Capacitance Noise Suppression T108 object needs to be made aware of when a noise source appears and disappears (for example, when a charger is turned on or off) or when the current noise level rises to a level beyond which the current acquisition parameters can cope. The following notification mechanisms are available:

- **Software trigger** – The host can set the NOISEON bit in the CALCFG1 field when a noise source appears. For example, this bit should be set when a noisy charger is plugged into the user's product. Note that the alternative use of the NOISEON bit in the CFG1 field is not recommended.
- **Automatic transition** – The noise suppression algorithm automatically transitions between the noise states as the noise level exceeds or drops below certain thresholds. These thresholds are specified by the STABHIGNLTHR, NOISLOWNLTHR, NOISHIGNLTHR and VNIOSLOWNLTHR fields.

IMPORTANT! Enabling and disabling the Self Capacitance Noise Suppression T108 object dynamically using the CTRL ENABLE bit in response to noise is not recommended, as this can cause unstable behavior. The correct procedure is to enable the object at startup and then use one of the notification methods above whenever noise is detected.

5.12.1.2 Transitions Between Noise States

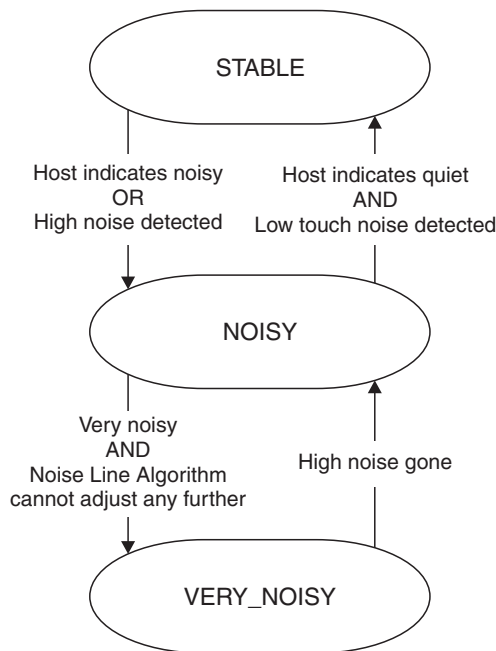
The Self Capacitance Noise Suppression T108 can also dynamically switch between three wholly independent configurations presets, known as noise states. The three noise states are referred to as STABLE, NOISY and VERY_NOISY, and are intended to be configured to counter the effects of increased measured noise. Typically noise states offer differing touch measurement configuration, which improves noise immunity at the expense of acquisition power consumption touch.

Noise state transition as a consequence of increased noise (that is, from STABLE to NOISY, or NOISY to VERY_NOISY) is known as promotion, which typically occurs when noise escalates beyond the effectiveness of frequency hopping (see [Section 5.12.1.3 "Frequency Hopping"](#)).

Conversely, noise state transition as a consequence of decreased noise (that is, from VERY_NOISY to NOISY, or NOISY to STABLE) is known as demotion, which occurs when noise recedes to more satisfactory levels.

[Figure 5-19](#) shows how the noise suppression algorithm transitions between the STABLE, NOISY and VERY_NOISY noise states as it notified of increasing or decreasing levels of noise.

FIGURE 5-19: NOISE SUPPRESSION STATES



In each state the Self Capacitance Noise Suppression T108 object can dynamically load a new set of acquisition parameters to cope with the current level of noise. This provides a framework within which the behavior of the Self Capacitance Noise Suppression T108 can be modified to provide better noise handling at the expense of touch performance and/or touch message rate.

On entry to a new noise state, the Self Capacitance Noise Suppression T108 object loads the appropriate set of control, frequency and ADCs per X parameters and the touchscreen is recalibrated to use the new settings.

5.12.1.3 Frequency Hopping

The Self Capacitance Noise Suppression T108 object works much like the Noise Suppression T72 object. It uses an automatic algorithmic method whereby measurements on special “noise lines” allow an appropriate action to be taken, such as choosing a new frequency.

For touch measurement, there are up to 5 configurable frequency presets, but the actual number that are used is configurable (see NUMPRESETS control in the “CFG3 Field”). Each such preset is accompanied by ADCs Per X configurations, for both touch and no-touch environments. The acquisition index (ACQIDX) addresses the preset in the frequencies array that is currently being used for the line bursts as part of touch measurement.

At any given juncture, one of these presets is used to drive touch measurements and the index of the selected preset is reported in the Self Capacitance Noise Suppression T108 messages.

The algorithm for noise suppression works by analyzing the maximum absolute values of the noise levels (known as noise lines tallest deltas – NLTD) to determine the nature of the noise detected and the consequential best course of action. This data is processed using IIR filters, so that inherent fluctuations in noise measurements can be countered.

Furthermore, continuous noise measurements are also performed automatically, and these are completely separate from the touch requirement. This carousel of measurements, from which a set of IIR filters are updated is referred to as background scanning.

The subsequent determination of the optimal frequency configuration to minimize the effects of noise at a given moment, and the transition to a different preset configuration for touch measurement, is known as frequency hopping.

5.12.1.4 Background Scanning and Touch Hold-off Qualification

For the Self Capacitance Noise Suppression T108 object to perform effective frequency hopping, continuous noise measurements are made for each of the candidate presets. This allows for the selection of the optimal settings that can best avoid the effects of noise.

The Self Capacitance Noise Suppression T108 object maintains two sets of IIR filter measurements (carousels), each set associated with a specific touch or no-touch state.

A pre-qualification process is applied to background noise measurements. This collates the measurements until all the measurements are deemed to have been collated within a period of consistent touch-state reporting. Once this is achieved the carousel filters updated and any subsequent frequency hopping determination is made. If a touch state transition occurs during a measurement cycle, all the noise measurements are rejected and a new carousel collation begins. This process is referred to as Touch Hold-off.

The Touch Hold-off mechanism can improve the frequency hopping candidate selection process, as the filter carousel content is more accurate when repeated touch transitions take place. Without this process, mis-reports from false touches and/or momentary loss of touch determination can reduce the effectiveness of any frequency hopping.

Touch Hold-off is selectable using the TCHOFFEN bit in the CFG3 field (see "CFG3 Field").

5.12.1.5 Suggested Initial Settings

Suggested initial settings for the Self Capacitance Noise Suppression T108 object are given in [Table 5-39](#).

TABLE 5-39: SUGGESTED INITIAL SETTINGS

Field	Suggested Initial Setting
CTRL	ENABLE = 1 Other bits as required by user
CALCFG1	0 when a potential noise source is not present (for example, a charger is turned off) 1 (NOISEON = 1) when a potential noise source is notified by the host (for example, a charger is turned on)
CFG1	0
CFG2	0
CFG3	64 (TCHOFFEN = 1)
NLGAIN	Field needs tuning; depends on whether Dual X is running or not. Start with Multiple Touch Touchscreen T100 XGAIN setting
IIRCOEFF	0
MINNLTDDIFF	6 Field benefits from tuning
MINNLTDHOP	3 Field benefits from tuning
HOPST	0
STABCTRL	0
STABFREQ[]	Fields need tuning
STABTCHAPX[]	Fields need tuning
STABNOTCHAPX[]	Fields need tuning
STABHIGHNLTHR	Field needs tuning
NOISCTRL	1 (AUTO = 1)
NOISFREQ[]	Fields need tuning
NOISCHAPX[]	Fields need tuning
NOISNOTCHAPX[]	Fields need tuning
NOISLOWNLTHR	Field needs tuning
NOISHIGHNLTHR	Field needs tuning
NOISCNT	3 (NOTCH = 3)
VNIOCTRL	15 (AUTO = 1)
VNOIFREQ[]	Fields need tuning
VNOITCHAPX[]	Fields need tuning
VNOINOTCHAPX[]	Fields need tuning

TABLE 5-39: SUGGESTED INITIAL SETTINGS (CONTINUED)

Field	Suggested Initial Setting
VNOILOWNLTHR	Field needs tuning
VNOICNT	3 (NOTCH = 3)
BLKNLTDTHR	Field needs tuning

5.12.2 CONFIGURATION

**TABLE 5-40: CONFIGURATION FOR SELF CAPACITANCE NOISE SUPPRESSION T108
(PROCG_NOISESUPSELFCAPI_T108)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved			RPTLVLEN	RPTSTATEEN	RPTACQIDXEN	RPTEN	ENABLE
1	CALCFG1	Reserved						VNOISEON	NOISEON
2	CFG1	Reserved						VNOISEON	NOISEON
3	CFG2	NOTCHNOISEON	Reserved						DISBLKTCH
4	CFG3	Reserved	TCHHOFFEN	Reserved			NUMPRESETS		
5	NLGAIN	Noise line gain							
6	Reserved	Reserved							
7	IIRCOEFF	Reserved					NLTD		
8	MINNLTDIFF	Minimum measured filter range for NLTD-based frequency hopping algorithm							
9	MINNLTDHOP	Minimum filter change for NLTD-based frequency hopping algorithm							
10	Reserved	Reserved							
11	Reserved	Reserved							
12	HOPST	Hop Suspend Time							
13	STABCTRL	CAL	Reserved						
14	STABFREQ[]	STABLE State Frequency Setting 0							
15		STABLE State Frequency Setting 1							
16		STABLE State Frequency Setting 2							
17		STABLE State Frequency Setting 3							
18		STABLE State Frequency Setting 4							
19	STABTCHAPX[]	STABLE State Touch ADCSPERX Setting 0							
20		STABLE State Touch ADCSPERX Setting 1							
21		STABLE State Touch ADCSPERX Setting 2							
22		STABLE State Touch ADCSPERX Setting 3							
23		STABLE State Touch ADCSPERX Setting 4							
24	STABNOTCHAPX[]	STABLE State No Touch ADCSPERX Setting 0							
25		STABLE State No Touch ADCSPERX Setting 1							
26		STABLE State No Touch ADCSPERX Setting 2							
27		STABLE State No Touch ADCSPERX Setting 3							
28		STABLE State No Touch ADCSPERX Setting 4							
29 – 30	Reserved	Reserved							
31	STABHINLTDTHR	STABLE State high NLTD noise level							
32	Reserved	Reserved							
33	NOISCTRL	CAL	Reserved						AUTO
34	NOISFREQ[]	NOISY State Frequency Setting 0							
35		NOISY State Frequency Setting 1							
36		NOISY State Frequency Setting 2							
37		NOISY State Frequency Setting 3							
38		NOISY State Frequency Setting 4							

**TABLE 5-40: CONFIGURATION FOR SELF CAPACITANCE NOISE SUPPRESSION T108
(PROCG_NOISESUPSELFCAP_T108) (CONTINUED)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
39	NOISTCHAPX[]	NOISY State Touch ADCSPERX Setting 0							
40		NOISY State Touch ADCSPERX Setting 1							
41		NOISY State Touch ADCSPERX Setting 2							
42		NOISY State Touch ADCSPERX Setting 3							
43		NOISY State Touch ADCSPERX Setting 4							
44	NOISNOTCHAPX[]	NOISY State No Touch ADCSPERX Setting 0							
45		NOISY State No Touch ADCSPERX Setting 1							
46		NOISY State No Touch ADCSPERX Setting 2							
47		NOISY State No Touch ADCSPERX Setting 3							
48		NOISY State No Touch ADCSPERX Setting 4							
49	Reserved	Reserved							
50	NOISLONLTDTHR	NOISY State low NLTD noise level							
51	NOISHINLTDTHR	NOISY State high NLTD noise level							
52	NOISCNT	LOWNOISTCH				NOTCH			
53	VNOICTRL	CAL	Reserved						AUTO
54	VNOIFREQ[]	VERY_NOISY State Frequency Setting 0							
55		VERY_NOISY State Frequency Setting 1							
56		VERY_NOISY State Frequency Setting 2							
57		VERY_NOISY State Frequency Setting 3							
58		VERY_NOISY State Frequency Setting 4							
59	VNOITCHAPX[]	VERY_NOISY State Touch ADCSPERX Setting 0							
60		VERY_NOISY State Touch ADCSPERX Setting 1							
61		VERY_NOISY State Touch ADCSPERX Setting 2							
62		VERY_NOISY State Touch ADCSPERX Setting 3							
63		VERY_NOISY State Touch ADCSPERX Setting 4							
64	VNOINOTCHAPX[]	VERY_NOISY State No Touch ADCSPERX Setting 0							
65		VERY_NOISY State No Touch ADCSPERX Setting 1							
66		VERY_NOISY State No Touch ADCSPERX Setting 2							
67		VERY_NOISY State No Touch ADCSPERX Setting 3							
68		VERY_NOISY State No Touch ADCSPERX Setting 4							
69	Reserved	Reserved							
70	VNOILONLTDTHR	VERY_NOISY State Low noise detection level							
71	Reserved	Reserved							
72	VNOICNT	LOWNOISTCH				NOTCH			
73	BLKNLTDTHR	Maximum NLTD threshold before blocking touch processing							
74	Reserved	Reserved							

CTRL Field

This is the main control field.

ENABLE: Enables the Self Capacitance Noise Suppression T108 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows this object to send messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

RPTACQIDXEN: Allows selected frequency or ADCs per X changes to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTSTATEEN: Allows changes in noise state or blocking of touch processing to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

RPTLVLEN: Allows noise level changes to trigger reporting messages to the host. Reporting is enabled if set to 1, and disabled if set to 0.

CALCFG1 Field

NOTE Writing this field will cause a recalibration to occur.

NOISEON: This bit can be set to 1 by the host to indicate that a noise source is activated for the product (for example, a charger is turned on). The object state-machine will change to the NOISY state if it was previously in the STABLE state

VNOISEON: Manually forces a change to the VERY_NOISY state for use during setup or development. If this bit is set to 1, the object state machine changes to the VERY_NOISY state if it was previously in the NOISY or STABLE state.

CFG1 Field

NOTE Use of this field is not recommended.

This field repeats bits from CALCFG1 field so that these bits can be changed without causing a touchscreen recalibration.

NOISEON: See [“CALCFG1 Field”](#).

VNOISEON: See [“CALCFG1 Field”](#).

CFG2 Field

DISBLKTCH: This bit can be set to 1 by the host to disable the blocking of touch processing when noise state promotion occurs. When a noise state transition occurs, it is often in response to a sudden escalation of measured noise, which may cause touch processing to perform ineffectively. The default behavior, when this bit is set to 0, is to block touch processing briefly as a safeguard in these circumstances. If this bit is set to 1, then no intervention with touch processing is made.

NOTCHNOISEON: Enables demotion of noise-level from either NOISY or VERY_NOISY states when clamped to one of these noise-levels by either the NOISEON or VNOISEON bits (CALCFG1 or CFG1 field) respectively. Consequently, this field alters the default behavior when noise-level promotion is enforced by configuration settings established by the host device.

Such a demotion can only occur during continuous touch, whereby noise-levels are deemed acceptably low. Essentially this bit allows the device to assume that noise is present until a touch is introduced for a sufficient time to determine its demotion to a quieter noise state.

If such a demotion occurs, then as soon as touch is removed, the noise-level will revert to the original state, as mandated by the configuration bits specified above.

CFG3 Field

NUMPRESETS: Specifies the number of frequency presets to use for touch measurement. Up to five preset frequencies are available, as defined in [“STABFREQ\[\], NOISFREQ\[\] and VNOIFREQ\[\] Fields”](#), [“STABFREQ\[\], NOISFREQ\[\] and VNOIFREQ\[\] Fields”](#), and [“STABFREQ\[\], NOISFREQ\[\] and VNOIFREQ\[\] Fields”](#). The default of 0 means that the maximum number of 5 presets will be used.

Range: 0 (5), 1 to 5

TCHHOFFEN: Enables Touch Hold-off (see [Section 5.12.1.4 “Background Scanning and Touch Hold-off Qualification”](#)). If this bit is set 1, the IIR filters are held off from being updated until touch status (in-touch or no-touch) is consistent over time. If a touch state transition occurs during a measurement cycle, all the noise measurements are rejected, and a new carousel collation begins.

This option improves filter quality during touch and no-touch transitions, but does so at the potential expense of frequency-hopping responsiveness.

NLGAIN Field

This field specifies the Noise Lines Gain to be used. Note that this is the gain for use when making noise line measurements; the gain for touch measurements is set by the Multiple Touch Touchscreen T100 object.

Range: 0 (reserved), 1 to 255

IIRCOEFF Field

This field specifies the coefficients applied to IIR filters that affect their response to changes in noise.

NLTD: Specifies the Coefficient for the IIR filtering of noise lines tallest deltas (NLTD) values. The range is 1 to 7, where a value of 1 is the fastest and 7 is the slowest. The default value of 0 means 5.

NLTD Range: 0 (5), 1 (fastest) to 7 (slowest)

MINNLTDIFF Field

This field specifies the minimum range (in 8-bit deltas) for the NLTD IIR filters below which the algorithm deems that noise measurements cannot yield a suitable candidate for frequency hopping. The minimum range is defined as the absolute extent between the noisiest and quietest filters associated with the presets currently available, as configured for the current noise state.

When the measured range is greater than or equal to this difference threshold, the frequency selection algorithm will make use of the NLTD IIR filters to determine a candidate preset for touch measurements.

This field (which is used in conjunction with the ["MINNLTDHOP Field"](#)) is used to prevent excessive frequency hopping under very low noise environments, or when there is no clear candidate, based on IIR filter analysis.

A value of 255 effectively means that the NLTD data can never yield a suitable candidate for frequency hopping.

Range: 0 to 255 (8-bit deltas)

MINNLTDHOP Field

This field specifies the minimum filter change (in 8-bit deltas) for NLTD IIR filters below which it is deemed that noise measurements do not indicate a suitable frequency hopping candidate (that is, less effected by noise).

This filter change is defined as the difference between the NLTD filter value of the currently selected acquisition index and that of the quietest alternative NLTD filter. If the difference in the two filters exceeds this specified minimum, then a suitable candidate for frequency hopping is deemed to have been found.

This field is only considered if the criterion defined by MINNLTDIFF is met first (see ["MINNLTDIFF Field"](#)).

Range: 0 to 255 (8-bit deltas)

HOPST Field

This field specifies the Hop Suspend Time. This is the time period from when the last frequency hop occurred to when a new frequency hop is allowed. The Hop Suspend Time is specified in units of 10 acquisition cycles. A value of zero effectively results in no hop timing restrictions.

Range: 0 to 255 (10 acquisition cycle increments)

STABCTRL, NOISCTRL and VNOICTRL Fields

These fields are used to configure specific operation settings for the appropriate noise states.

AUTO (NOISCTRL and VNOICTRL only): Controls the transitions into and out of the NOISY/VERY_NOISY state. Specifically, this bit selects whether transitions between the states can be made based on measured noise levels. If this bit is set to 1, the transitions are influenced by the detected noise levels. If this bit is set to 0, the detected noise levels are ignored. In the case of the VERY_NOISY state (VNOICTRL field), this bit effectively enables the use of the VERY_NOISY state.

When this bit is set to 1, the promotion from the STABLE to NOISY state is made using the following condition:

Detected noise \geq STABHINLTDTR

Similarly, the promotion from the NOISY to VERY_NOISY state is made using the following condition:

Detected noise \geq NOISHINLTDTR

The demotion from the VERY_NOISY to NOISY state is made using the following conditions:

Detected noise \leq VNOILONLTDTHR
for the last LOWNOISTCH cycles that had a touch detected

OR

No touch has been detected for NOTCH cycles

Similarly, the demotion from the NOISY to STABLE state, is made using the following conditions:

Detected noise \leq NOISLONLTDTHR
for the last LOWNOISTCH cycles that had a touch detected

OR

No touch has been detected for NOTCH cycles

Note: LOWNOISTCH and NOTCH in the above are in the NOISCNT and VNOICNT fields.

CAL: If this bit is set to 1, it forces a calibration on entry to the state.

STABFREQ[], NOISFREQ[] and VNOIFREQ[] Fields

These fields specify the set of five candidate frequencies that the noise suppression algorithm can select when operating in a particular state.

The number of candidate frequencies in use is determined by NUMPRESETS in “CFG3 Field”. If NUMPRESETS is less than 5, then the latter presets within the arrays are ignored.

Range: 0 to 255

STABTCHAPX[], NOISTCHAPX[] and VNOITCHAPX[] Fields

These fields specify the set of five candidate ADCs per X settings that the noise suppression algorithm can select when operating in the appropriate noise states when touch has been detected (see [Section 5.12.1 “Introduction”](#)).

The number of candidate ADCs per X settings in use is determined by NUMPRESETS in “CFG3 Field”. If NUMPRESETS is less than 5, then the latter presets within the arrays are ignored.

Range: 0 (255), 1 to 255

STABNOTCHAPX[], NOISNOTCHAPX[] and VNOINOTCHAPX[] Fields

These fields specify the set of five candidate ADCs per X settings that the noise suppression algorithm can select when operating in the appropriate noise states when no touch has been detected (see [Section 5.12.1 “Introduction”](#)).

The number of candidate no touch ADCs per X settings in use is determined by NUMPRESETS in “CFG3 Field”. If NUMPRESETS is less than 5, then the latter presets within the arrays are ignored.

Range: 0 (255), 1 to 255

NOISLONLTDTHR and VNOILONLTDTHR Fields

These fields define the noise threshold (in 8-bit deltas) below which the noise is identified as too low for the current state. When the noise level is less than VNOILONLTDTHR, the noise suppression algorithm enters NOISY state and when the noise level is less than NOISLONLTDTHR, the noise suppression algorithm enters STABLE state.

The range for these fields is 0 to 255

Range: 0 to 255 (8-bit deltas)

STABHINLTDTHR and NOISHINLTDTHR Fields

These fields define the noise threshold (in 8-bit deltas) above which the noise is identified as too high for the current state. When the noise level is greater or equal to STABHINLTDTHR, the noise suppression algorithm enters the NOISY state and when the noise level is greater or equal to NOISHINLTDTHR, the noise suppression algorithm enters the VERY_NOISY state.

The range for these fields is 0 to 255, where a value of 0 indicates that this threshold is disabled and therefore cannot impose a noise state promotion accordingly.

Range: 0 (no threshold applied), 1 to 255 (8-bit deltas)

NOISCNT and VNOICNT Fields

These fields provide real-time counters that are used in the NOISY and VERY_NOISY states.

NOTCH: Specifies the number of “no touch” cycles used to identify a low noise condition. This value is specified in units of 100 cycles, where a value of 0 means 1 (100 cycles).

NOTCH Range: 0 (1 = 100 cycles), 1 to 15 (number of cycles in units of 100 cycles)

LOWNOISTCH: Specifies the number of “low noise touched” cycles used to identify a low noise condition. This field is specified in units of 10 cycles.

LOWNOISTCH Range: 0 (do not wait for “low noise touched”),
1 to 15 (number of cycles in units of 10 cycles)

BLKNLTDTHR Field

This field specifies the Block NLTD Threshold. This is the threshold (in 8-bit deltas) at which measured noise is deemed excessively high, resulting in the blocking of touch processing. The default value of 0 indicates that no blocking threshold is applied.

Range: 0 (no blocking threshold applied), 1 to 255 (8-bit deltas)

5.12.3 CONFIGURATION CHECKS

The Self Capacitance Noise Suppression T108 object causes a configuration check to be performed in the following circumstances:

- When certain fields are changed

In addition, some fields will cause an automatic recalibration to be performed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3 “Command Processor T6 Object”](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

Note that while the object is disabled, potential configuration errors are allowed.

TABLE 5-41: CONFIGURATION CHECKS FOR SELF CAPACITANCE NOISE SUPPRESSION T108 (PROCG_NOISESUPSELFCAP_T108)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes ⁽¹⁾	None
CALCFG1	Yes	Yes	May cause a noise state transition, so filters may require resetting.
CFG1	Yes	No	May cause a noise state transition, so filters may require resetting.
CFG2	Yes	No	May cause a noise state transition, so filters may require resetting.
CFG3	Yes	No	May cause a noise state transition, so filters may require resetting.
NLGAIN	Yes	Yes	Affects noise acquisition measurements, so filters will require resetting
IIRCOEFF	Yes	No	Filters require resetting
MINNLTDIFF	No	No	None
MINNLTDHOP	No	No	None
HOPST	Yes	No	None
STABCTRL	Yes	No	May cause a noise state transition, so filters may require resetting.
STABFREQ[]	No	No	None
STABTCHAPX[]	No	No	None

TABLE 5-41: CONFIGURATION CHECKS FOR SELF CAPACITANCE NOISE SUPPRESSION T108 (PROCG_NOISESUPSELFCAP_T108) (CONTINUED)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
STABNOTCHAPX[]	No	No	None
STABHINLTDTHR	No	No	None
NOISCTRL	Yes	No	May cause a noise state transition, so filters may require resetting.
NOISFREQ[]	No	No	None
NOISTCHAPX[]	No	No	None
NOISNOTCHAPX[]	No	No	None
NOISLONLTDTHR	No	No	None
NOISHINLTDTHR	No	No	None
NOISCNT	No	No	None
VNOICTRL	Yes	No	May cause a noise state transition, so filters may require resetting.
VNOIFREQ[]	No	No	None
VNOITCHAPX[]	No	No	None
VNOINOTCHAPX[]	No	No	None
VNOILONLTDTHR	No	No	None
VNOICNT	No	No	None
BLKNLTDTHR	No	No	None

5.12.4 MESSAGES

The message data for the Self Capacitance Noise Suppression T108 object is shown in [Table 5-42](#).

TABLE 5-42: MESSAGE DATA FOR SELF CAPACITANCE NOISE SUPPRESSION T108 (PROCG_NOISESUPSELFCAP_T108)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS1	Reserved			LVLCHG	STATECHG	ACQIDXCHG	TCH	BLKTCH
2	STATUS2	ACQIDX			Reserved		STATE		
3	SUSHOPS	The number of suspended hops since last change							
4	PKNLTDLVL	The last measured NLTD noise level							
5	Reserved	Reserved							
6	PKNLTDDIFF	The peak measured filter range for NLTD based frequency hopping algorithm							
7	PKNLTDHOP	The peak measured quieter-hop filter change for NLTD based frequency hopping algorithm							

STATUS1 Field

This field reports the current status of the noise suppression algorithm.

BLKTCH: Indicates noise suppression algorithm has generated a request to block touch processing. This can occur under excessive measured noise-levels (as configured using the “[BLKNLTDTHR Field](#)”) or briefly when noise state promotion occurs (if DISBLKTCH is set to 0; see “[CFG2 Field](#)”).

TCH: Indicates that a touch is present. This bit is set to 1 if a touch is present, and is set to 0 if no touch is present.

ACQIDXCHG: Indicates that the noise suppression algorithm has changed the number of ADCs per X at least once since the previous message (that is, there has been a frequency hop). This bit is set to 1 if the number of ADCs per X has changed, and 0 otherwise.

The index of the frequency preset being used is reflected in the STATUS2 ACQIDX bit (see “[STATUS2 Field](#)”).

STATECHG: Indicates that the noise suppression algorithm has changed its state at least once since the previous message. This bit is set to 1 if the state has changed, and 0 otherwise.

The details of the state change are reported in the STATUS2 message field (see [“STATUS2 Field”](#)).

LVLCHG: Indicates that the measured noise level has changed at least once since the previous message. This bit is set to 1 if the measured noise level has changed, and 0 otherwise.

The peak measured NLTD noise level is reported in the PKNLTDLVL message field (see [“PKNLTDLVL Field”](#)).

STATUS2 Field

This field reports the current status of the noise suppression algorithm.

STATE: Indicates the current noise suppression algorithm state selected (see [Table 5-43](#)).

TABLE 5-43: NOISE SUPPRESSION ALGORITHM STATES

Value	State
0	STABLE
1	NOISY
2	VERY_NOISY

ACQINDEX: Indicates the index of the current ADCs per X settings and frequency settings in use (that is, an index into the ADCSPERX and frequency byte arrays between bytes 14 and 28, 34 and 48, and 54 and 68; see [Table 5-40](#)).

SUSHOPS Field

This field reports the number of suspended hops since the last change.

This field indicates the number of instances when a quieter preset was determined, but ignored because of a requirement to suspend frequency-hopping, as mandated by the HOPST configuration field (see [“HOPST Field”](#)).

This gives a useful indication of the variance in the measured noise between hops.

PKNLTDLVL Field

This field reports the peak measured NLTD noise level, since the last message was dispatched.

This noise level is the value of measured noise that is assessed against:

- STABHINLTHR and NOISHINLTHR, to invoke noise-state promotion
- NOISLONLTHR and VNOILONLTHR to invoke noise-state demotion
- BLKNLTDTHR to invoke blocking of touch processing under high noise levels

PKNLTDDIFF Field

This field reports the peak filter range limit for NLTD noise measurements, since the last message was dispatched.

This range is the largest measured difference between IIR filters, and is assessed against:

- MINNLTDDIFF to assess candidate presets and invoke frequency hopping.

PKNLTDHOP Field

This field reports the peak measured quieter IIR filter difference for the NLTD noise measurements, since the last message was dispatched.

This range is the largest measured difference between the IIR filter associated with the current presets being used for touch measurements and the IIR filter with the lowest noise level, and is assessed against:

- MINNLTDHOP to assess the candidate preset and invoke frequency hopping.

5.13 Symbol Gesture Processor T115 Object

5.13.1 INTRODUCTION

A Symbol Gesture Processor T115 object detects arbitrary shaped gestures by decomposing them into a series of ordinal strokes. These typically form alphabetic characters or symbols, such as a star shape, drawn by the user. This could be used by the host as an unlock mechanism, as a shortcut to trigger a certain application, or other similar uses.

The strokes that make up each symbol are defined in the Symbol Gesture Configuration T116 object (see [Section 6.16 "Symbol Gesture Configuration T116 Object"](#)). A message is triggered by the Symbol Gesture Processor T115 object if the shape drawn matches one of the stroke sequences held in the Symbol Gesture Configuration T116 object.

Each stroke is defined as a touch movement in a certain direction (Left, Right, Up or Down) of more than a defined number (MOVX/MOY) of pixels. Note that only one of MOVX or MOY needs to be exceeded to trigger the recognition of a stroke. For completeness, a release (removal of touch) is also considered to be a type of stroke.

The start positions of each stroke (including release strokes) of a detected symbol can be read through the Diagnostic Debug T37 object.

5.13.2 CONFIGURATION

**TABLE 5-44: CONFIGURATION FOR SYMBOL GESTURE PROCESSOR T115
(PROCI_SYMBOLGESTURE_T115)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	DBG RPTEN	Reserved				MULTTCHSUPEN	RPTEN	ENABLE
1	XMIN	Low X Bound LSByte							
2		Low X Bound MSByte							
3	XMAX	High X Bound LSByte							
4		High X Bound MSByte							
5	YMIN	Low Y Bound LSByte							
6		Low Y Bound MSByte							
7	YMAX	High Y Bound LSByte							
8		High Y Bound MSByte							
9	MOVX	MOVX LSByte							
10		MOVX MSByte							
11	MOVY	MOVY LSByte							
12		MOVY MSByte							
13	SYMTO	Symbol Timeout							
14	SYMTEMAX	Symbol Time Maximum							
15	EXTSTRCNTMAX	Extra Stroke Count Maximum							
16	LSTRCFG1	Reserved				DISLSTRDOWN	DISLSTRUP	DISLSTRRIGHT	DISLSTRLEFT
17	LSTRCNTTHR	LSTRCNTTHRY				LSTRCNTTHRX			
18	LSTRNUMTCH	Reserved	LSTRNUMTCHY			Reserved	LSTRNUMTCHX		
19	LSTRANGLE	Reserved	LSTRANGLEY			Reserved	LSTRANGLEX		

CTRL Field

ENABLE: Enables the use of this Symbol Gesture Processor T115 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

MULTTCHSUPEN: Enables Multi-touch Suppression. If this bit is set to 1, the symbol sequence is invalidated when there is more than one touch simultaneously detected on the touchscreen.

DBG RPTEN: Enables Debug Reporting. If this bit is set to 1, a debug report message is sent out whenever an individual stroke is detected on the screen. This feature is useful to see what strokes make up a symbol when troubleshooting or developing symbol stroke sequences. See [Section 5.13.4 "Messages"](#) for details of the debug output.

XMIN, YMIN, XMAX, YMAX Fields

These four fields define a bounding box within which all touch, move and release events must occur for a symbol capture sequence to be considered valid. The values for the XMIN, YMIN, XMAX, YMAX fields are specified in touchscreen pixels.

Range: 0 to 65535

MOVX and MOVY Fields

These fields define the X-direction Movement and Y-direction Movement. This is the amount of movement required for a new stroke gesture element to be detected when the movement is in the appropriate axis. The required movement is specified in touchscreen pixels, where the default value of 0 means Multiple Touch Touchscreen T100 X RANGE / 4 or Y RANGE / 4, as appropriate.

Range: 0 (T100 X RANGE/Y RANGE), 1 to 65535

SYMTO Field

This field specifies the Symbol Timeout for multi-stroke symbols.

Many symbols consist of multiple strokes separated by touch releases. For example, an uppercase “A” requires a touch release followed by a touch in order to draw the cross bar. This field therefore provides a timeout such that a subsequent stroke must start less than the timeout period after the previous release to be considered part of the same symbol. If another stroke does not begin before the timeout expires, then the sequence terminates and the search for a symbol match starts.

The Symbol Timeout is specified in units of 16 ms, where the default value of 0 means 30 (approximately 480 ms).

Range: 0 (30 = 480 ms), 1 to 255 (16 ms to 4 seconds)

SYMTIMEMAX Field**Long Name:**

This field specifies the Maximum Symbol Time. This is the maximum time (from a touchdown to release) the user is allowed to take to draw a symbol. If the touch is detected for more than the SYMTIMEMAX duration, then the current stroke sequence will be ignored.

Range: 0 (disable), 1 to 255 (32 ms – 8.16 seconds)

Units: 32ms

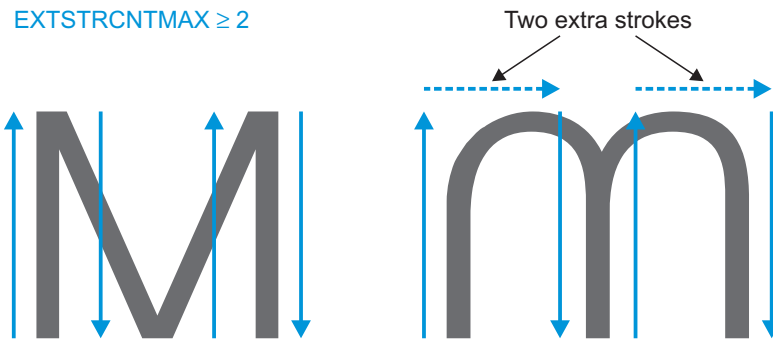
EXTSTRCNTMAX Field

This field specifies the Maximum Extra Stroke Count. Sometimes, users will add extra strokes (over and above the strokes explicitly defined in each symbol definition) to the way in which they draw symbols. This field sets a maximum number of extra strokes that can be detected before a stroke sequence is declared invalid as being too long or complex. This allows for some users’ writing styles, adding flicks or similar embellishments to the start or end of a valid stroke sequence.

For a non-rolling symbol, the extra strokes are allowed to be in between the defined strokes (non-rolling symbols are defined by the Symbol Gesture Configuration T116 ROLLSEQ; see “SYMDATA[] Fields”). For example, in Figure 5-20, an “M” (drawn as Up, Down, Up, Down) can be drawn as “m” (Up, Right, Down, Up, Right, Down) and still be recognized as an “M” if the extra stroke count is set to 2 or more. For rolling symbols, the extra strokes are only allowed at the beginning and end of a valid stroke sequence.

Range: 0 to 255

FIGURE 5-20: EFFECT OF EXTSTRCNTMAX FIELD



These two symbols will be considered equivalent (Up, Down, Up, Down) as the two extra strokes will be ignored.

LSTRCFG1 Field

DISLSTRLEFT, DISLSTRRIGHT, DISLSTRUP and DISLSTRDOWN: Disable the reporting of the long stroke in a particular direction. For example, setting DISLSTRLEFT to 1 disables the reporting of a long stroke in the left direction.

LSTRCNTTHR Field

This field allows for the easy configuration of a long stroke without the need for multiple SYMDATA[] stroke sequences. A long stroke in this context is a unidirectional stroke with a minimum size, such as a percentage of the touchscreen. It is often used as an unlock gesture by some users.

If long strokes are not used in an application, they may be disabled by setting the entire LSTRCNTTHR field to zero.

LSTRCNTTHRX and LSTRCNTTHRY: Configure the Long Stroke Count Thresholds for the X and Y axes. LSTRCNTTHRX defines the length of stroke that is required in the X axis and LSTRCNTTHRY defines the length of stroke that is required in the Y axis. The distance is specified as the number of MOVX or MOVY distances that the stroke must cover, to fulfill this requirement. The default value of 0 disables the long stroke in a particular direction.

LSTRCNTTHRX and LSTRCNTTHRY Range: 0 (disabled), 1 to 15 (number of MOVX/MOVY elements)

LSTRNUMTCH Field

This field specifies the number of touches required to trigger a long stroke in the X and Y axes. For example, if LSTRNUMTCHX is set to 1, then two simultaneous touches are required to trigger a LEFT/RIGHT long stroke on the X axis.

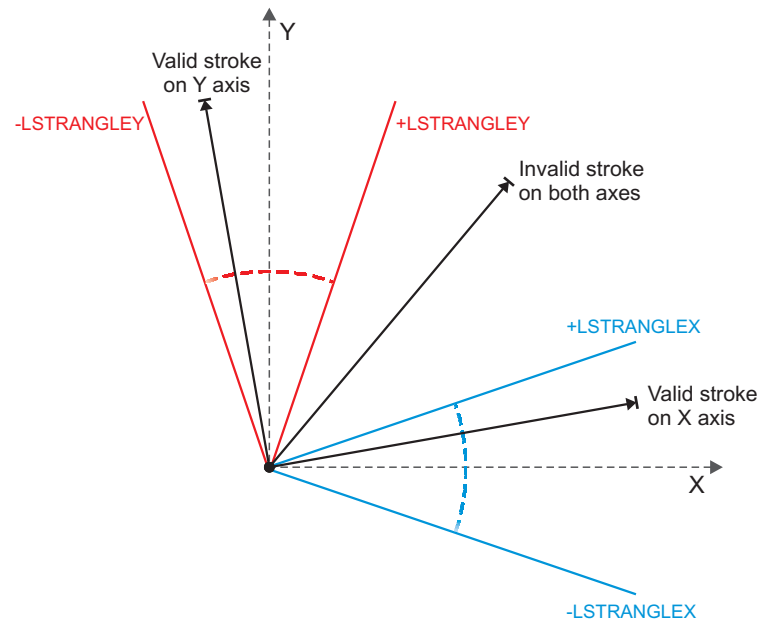
LSTRNUMTCHX and LSTRNUMTCHY: Determines the number of simultaneous touches required to trigger a Long Stroke on a particular axis. This value is specified as the number of touches minus 1. For example, if LSTRNUMTCH = 1, then two simultaneous touches are required.

LSTRNUMTCHX and LSTRNUMTCHY Range: 0 to 7 (number of touches minus 1)

LSTRANGLE Field

LSTRANGLEX and LSTRANGLEY: Configure the Long Stroke Angles for the X and Y axis directions. These controls specify the variance in angle that a long stroke is allowed to deviate from the ideal X/Y axis and still be considered valid. The angle is specified in units of 1/8 45°, where the default value of 0 means 8/8 (that is, the full 45° angle), which means no limits on the deviation allowed.

FIGURE 5-21: EFFECT OF LSTRANGLEX AND LSTRANGLEY



LSTRANGLEX and LSTRANGLEY Range: See [Table 5-45](#)

TABLE 5-45: LSTRANGLEX/LSTRANGLEY VALUES

Value	Angle in 1/8 of 45°
0	8/8
1	1/8
2	2/8
3	3/8
4	4/8
5	5/8
6	6/8
7	7/8

5.13.3 CONFIGURATION CHECKS

A Symbol Gesture Processor T115 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

TABLE 5-46: CONFIGURATION CHECKS FOR SYMBOL GESTURE PROCESSOR T115 (PROCI_SYMBOLGESTURE_T115)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	No	None
XMIN	No	No	None
XMAX	No	No	None

**TABLE 5-46: CONFIGURATION CHECKS FOR SYMBOL GESTURE PROCESSOR T115
(PROCI_SYMBOLGESTURE_T115) (CONTINUED)**

Field	Changing The Field Causes...		Effect of (Continued) Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
YMIN	No	No	None
YMAX	No	No	None
MOVX	No	No	None
MOVY	No	No	None
SYMTO	No	No	None
SYMTIMEMAX	No	No	None
EXTSTRCNTMAX	No	No	None
LSTRCFG1	No	No	None
LSTRCNTTHR	No	No	None
LSTRNUMTCH	Yes	No	Configuration error if LSTRNUMTCH > 4 (= maximum tracked touches minus 1)
LSTRANGLE	No	No	None

5.13.4 MESSAGES

The format of the message data for the Symbol Gesture Processor T115 object depends on the type of report.

Message Data for Normal Symbol Reports

**TABLE 5-47: MESSAGE DATA FOR SYMBOL GESTURE PROCESSOR T115
(PROCI_SYMBOLGESTURE_T115)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	RPTSYM	LSTR = 0	RPTCODE						
2	Reserved	Reserved							
3	DATA1	SYMDOFFSET							

Message Data for Normal Long Stroke Reports

**TABLE 5-48: MESSAGE DATA FOR SYMBOL GESTURE PROCESSOR T115
(PROCI_SYMBOLGESTURE_T115)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	RPTSYM	LSTR = 1	Reserved			STRTYPE			
2	Reserved	Reserved							
3	Reserved	Reserved							

Message Data for Debug Stroke Reports (CTRL_DBG RPTEN = 1)

**TABLE 5-49: MESSAGE DATA FOR SYMBOL GESTURE PROCESSOR T115
(PROCI_SYMBOLGESTURE_T115)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	RPTSYM	LSTR = 0	RPTCODE = 0						
2	DATA0	Reserved				DBGSTRTYPE			
3	Reserved	Reserved							

RPTSYM Field

This field reports the symbol data. The format of the data depends on the value of the reported LSTR bit.

RPTCODE: If LSTR = 0, reports a 7-bit symbol code, as defined by the Symbol Gesture Configuration T116 SYMDATA RPTCODE control (see “SYMDATA Fields”). If the CTRL_DBG RPTEN bit is set to 1, the RPTCODE control will be set to zero.

STRTYPE: If LSTR = 1, reports a Long Stroke as a 4-bit stroke type, as defined by the Symbol Gesture Configuration T116 SYMSTRSEQ field (see “SYMSTRSEQ[] Fields”).

LSTR: Determines the format is used to report the data. If this bit is set to 1, RPTCODE format is used. If this bit is set to 0, STRTYPE format is used.

DATA0 Field

This field is used only for debug stroke reports (that is, LSTR = 0 and DBGRPTEN = 1). For all other reports, this field is set to zero.

DBGSTRTYPE: reports the stroke type currently detected, as defined by the Symbol Gesture Configuration T116 SYMSTRSEQ field (see “SYMSTRSEQ[] Fields”). A message will be sent out as soon as a valid stroke is detected.

DATA1 Field

This field is used only for normal symbol reports (that is, LSTR = 0). For all other reports, this field is set to zero.

SYMDOFFSET: reports the offset in Symbol Gesture Configuration T116 SYMDATA for the symbol reported (see “SYMDATA[] Fields”). Note that different stroke combinations of the same symbol (with the same Report Code) can be defined in SYMDATA. SYMDOFFSET can therefore be used to identify which definition of the symbol triggered the match.

5.13.5 DEBUG DATA

The debug data modes for the Symbol Gesture Processor T115 object is shown in Table 5-50. The debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in Table 5-50.

The debug modes are described in the following sections.

TABLE 5-50: DIAGNOSTIC DEBUG COMMANDS

T6 Command	Name	Description
0x34	Symbol Gesture Mode	The Diagnostic Debug T37 object holds the start positions of each stroke (including release strokes) of a detected Symbol Gesture Processor T115 symbol. See Section 5.13.5.1 “Symbol Gesture Mode”.

5.13.5.1 Symbol Gesture Mode

Figure 5-22 shows the organization of the data in Symbol Gesture Mode. This reports the start positions of each stroke (including release strokes) of a detected Symbol Gesture Processor T115 symbol.

FIGURE 5-22: DIAGNOSTIC DEBUG T37 FIELDS – SYMBOL GESTURE MODE

Page	Data Index	Data[]
Page 0	0	Stroke count (that is the number of coordinate pairs)
	1 – 2	X coordinate LSByte for stroke 1 in sequence (LSByte, MSByte)
	3 – 4	Y coordinate LSByte for stroke 1 in sequence (LSByte, MSByte)
	...	
	77 – 78	X coordinate LSByte for stroke 20 in sequence (LSByte, MSByte)
	79 – 80	Y coordinate LSByte for stroke 20 in sequence (LSByte, MSByte)
	81 – 127	Reserved

Assumes page size = 128

5.14 Sensor Correction T121 Object

5.14.1 INTRODUCTION

A Sensor Correction T121 object allows adjustments to be made to mutual measurements from an associated touchscreen sensor. Normally sensors are expected to have uniform responses (that is, straight X and Y lines) and this object provides compensation for any irregularities that may be introduced by the sensor.

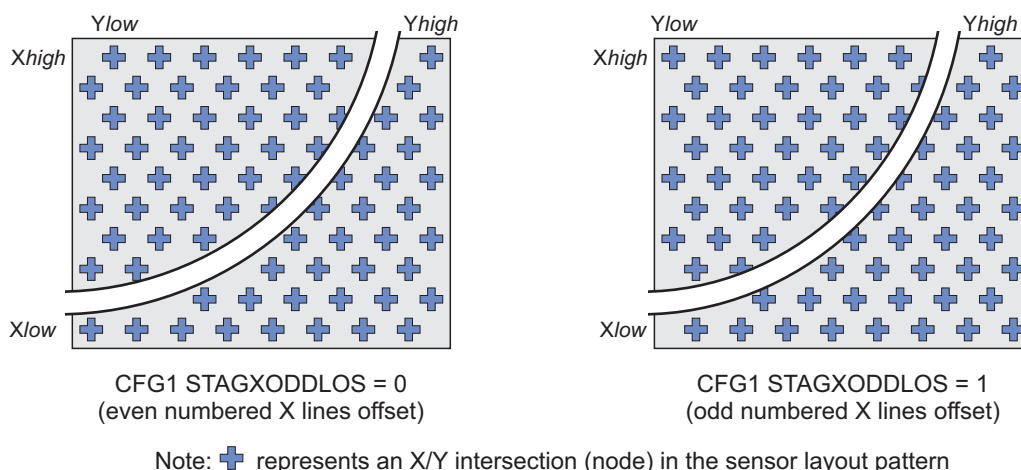
5.14.1.1 Staggered X Edge Correction

Staggered X edge correction corrects for potential irregularities in alternate delta measurements at the extreme Y edge nodes. This can be caused by a staggered X line sensor layout, such as on single layer caterpillar sensors.

Figure 5-23 gives an exaggerated illustration of the effective measurement node positions on the corner of a sensor that might exhibit this staggered X line irregularity. In this case, a staggered offset results on the *Ylow* edge in the measurement positions for even numbered X lines, with reference to the touchscreen XY origin. On the *Yhigh* edge, the staggered offset occurs on the inverse X line measurement positions (that is, on odd numbered X lines).

NOTE X and Y do not have to start at 0 and end at the maximum line on the sensor matrix. They apply to the touchscreen area only and so range from Multiple Touch Touchscreen T100 XORIGIN/YORIGIN origin to XSIZE/YSIZE – 1. For this reason the terms “low” and “high” are used here.

FIGURE 5-23: STAGGERED X EDGE CORRECTION



5.14.2 CONFIGURATION

**TABLE 5-51: CONFIGURATION FOR SENSOR CORRECTION T121
(PROCI_TSSENSORCORRECTION_T121)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved							ENABLE
1	CFG1	Reserved						STAGXODDLOS	STAGXYEDGEEN
2	YEDGESF	HI				LO			

CTRL Field

ENABLE: Enables the use of this Sensor Correction T121 object. The object is enabled if set to 1, and disabled if set to 0.

CFG1 Field

This field configures the sensor corrections. See [Section 5.14.1 “Introduction”](#) for details on the currently supported types of sensor corrections.

STAGXYEDGEEN: Enables Staggered X Edge Correction for Y lines (see [Section 5.14.1.1 “Staggered X Edge Correction”](#)). The sensor correction is performed if this bit is set to 1 and is not performed if this bit is set to 0.

STAGXODDLOS: Specifies whether odd or even X lines are offset with a staggered X sensor pattern (see [Figure 5-23](#)).

Set this bit to 1 if the staggered sensor layout has an odd numbered X line offset from the *Ylow* edge.

Set this bit to 0 if the staggered sensor layout has an even numbered X line offset from the *Ylow* edge.

YEDGESF Field

This field configures the Y edge scaling factors for the sensor corrections on Staggered X type single layer sensors.

LO: Specifies the *Ylow* scaling factor. This is the correction scaling factor to be applied to the edge corrections on the *Ylow* edge of the sensor. The scaling factor is specified in units of 1/8 of the total scaling factor applied, where the default value of 0 means 8 (scaling factor $8/8 = 1$).

LO Range: 0 (8), 1 to 15 (increments of 1/8 of total scaling factor applied)

HI: Specifies the *Yhigh* scaling factor. This is the correction scaling factor to be applied to the edge corrections on the *Yhigh* edge of the sensor. The scaling factor is specified in units of 1/8 of the total scaling factor applied, where the default value of 0 means 8 (scaling factor $8/8 = 1$).

HI Range: 0 (8), 1 to 15 (increments of 1/8 of total scaling factor applied)

6.0 SUPPORT OBJECTS

6.1 Introduction

Support objects provide additional functionality on the device. [Table 6-1](#) lists the support objects on the mXT144U.

TABLE 6-1: SUPPORT OBJECTS

Object	Description
Communications Configuration T18 (SPT_COMMSCONFIG_T18)	Configures additional communications behavior for the device. See Section 6.2 “Communications Configuration T18 Object” .
Self Test T25 (SPT_SELFTEST_T25)	Configures and performs self-test routines to find faults on a touch sensor. See Section 6.3 “Self Test T25 Object” .
User Data T38 (SPT_USERDATA_T38)	Provides a data storage area for user data. See Section 6.4 “User Data T38 Object” .
Message Count T44 (SPT_MESSAGECOUNT_T44)	Provides a count of pending messages. See Section 6.5 “Message Count T44 Object” .
CTE Configuration T46 (SPT_CTECONFIG_T46)	Controls the capacitive touch engine for the device. See Section 6.6 “CTE Configuration T46 Object” .
Timer T61 (SPT_TIMER_T61)	Provides control of a timer. See Section 6.7 “Timer T61 Object” .
Serial Data Command T68 (SPT_SERIALDATACOMMAND_T68)	Provides an interface for the host driver to deliver various data sets to the device. See Section 6.8 “Serial Data Command T68 Object” .
Dynamic Configuration Controller T70 (SPT_DYNAMICCONFIGURATIONCONTROLLER_T70)	Allows rules to be defined that respond to system events. See Section 6.9 “Dynamic Configuration Controller T70 Object” .
Dynamic Configuration Container T71 (SPT_DYNAMICCONFIGURATIONCONTAINER_T71)	Allows the storage of user configuration on the device that can be selected at runtime based on rules defined in the Dynamic Configuration Controller T70 object. See Section 6.10 “Dynamic Configuration Container T71 Object” .
Auxiliary Touch Configuration T104 (SPT_AUXTOUCHCONFIG_T104)	Allows the setting of self capacitance gain and thresholds for a particular measurement to generate auxiliary touch data for use by other objects. See Section 6.11 “Auxiliary Touch Configuration T104 Object” .
Self Capacitance Global Configuration T109 (SPT_SELFCAPGLOBALCONFIG_T109)	Provides configuration for a self capacitance measurements employed on the device. See Section 6.12 “Self Capacitance Global Configuration T109 Object” .
Self Capacitance Tuning Parameters T110 (SPT_SELFCAPTUNINGPARAMS_T110)	Provides configuration space for a generic set of settings for self capacitance measurements. See Section 6.13 “Self Capacitance Tuning Parameters T110 Object” .
Self Capacitance Configuration T111 (SPT_SELFCAPCONFIG_T111)	Provides configuration for self capacitance measurements employed on the device. See Section 6.14 “Self Capacitance Configuration T111 Object” .
Self Capacitance Measurement Configuration T113 (SPT_SELFCAPMEASURECONFIG_T113)	Configures self capacitance measurements to generate data for use by other objects. See Section 6.15 “Self Capacitance Measurement Configuration T113 Object” .
Symbol Gesture Configuration T116 (SPT_SYMBOLGESTURECONFIG_T116)	Stores configuration data that defines the symbols to be detected by the Symbol Gesture Processor T115 object. See Section 6.16 “Symbol Gesture Configuration T116 Object” .
Low Power Idle Configuration T126 (SPT_SELFCAPIIDLECONFIG_T126)	Configures the overall behavior of the low power self capacitance features. See Section 6.17 “Low Power Idle Configuration T126 Object” .

6.2 Communications Configuration T18 Object

6.2.1 INTRODUCTION

The Communications Configuration T18 object specifies additional communications behavior for the device.

6.2.2 CONFIGURATION

**TABLE 6-2: CONFIGURATION FOR COMMUNICATIONS CONFIGURATION T18
(SPT_COMMSCONFIG_T18)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved	RETRIGEN	Reserved		HISPEEDSPI	MODE	Reserved	
1	COMMAND	CHG line command code							

CTRL Field

MODE: Selects the $\overline{\text{CHG}}$ line mode. If this bit is set to 0 (the default), the $\overline{\text{CHG}}$ line operates in Mode 0. If this bit is set to 1, the $\overline{\text{CHG}}$ line operates in Mode 1. Refer to the *mXT144U 1.0 Data Sheet* for more information on the $\overline{\text{CHG}}$ line modes.

HISPEEDSPI: If this bit is set to 1, debug data is sent over High Speed SPI. If this bit is set to 0 (default), debug data is sent over Normal Speed SPI (the normal speed for Microchip touchscreen products).

RETRIGEN: Enables $\overline{\text{CHG}}$ line host retriggering mode. This provides extra edge-based interrupts to the host on the CHG line. If this mode is enabled, the CHG line is deasserted once every acquisition cycle for 50 μs and then re-asserted. This creates edges on the CHG line for the host in case it missed the CHG line being asserted. CHG line host retriggering mode is enabled if this bit is set to 1 and disabled if set to 0. Note that if command code 2 or 3 is set in the COMMAND field, then the command will override the RETRIGEN mode setting.

A minimum acquisition time of 10 ms is required for host retriggering to operate. Power Configuration T7 IDLEACQINT and ACTVACQINT must therefore be set to a minimum of 10 ms (see “IDLEACQINT and ACTVACQINT Fields” and “IDLEACQINT and ACTVACQINT Fields”).

COMMAND Field

The COMMAND field provides direct control over the $\overline{\text{CHG}}$ line. This overrides the operation of the $\overline{\text{CHG}}$ line controlled by the MODE bit in the CTRL field. Entering one of the command codes listed in Table 6-3 alters the state of the CHG line.

TABLE 6-3: COMMAND CODES

Command	Description
0	No command (default)
1	Return $\overline{\text{CHG}}$ line to normal operation, as determined by the MODE bit of the CTRL field
2	Force the $\overline{\text{CHG}}$ line high (inactive)
3	Force the CHG line low (active)

6.3 Self Test T25 Object

6.3.1 INTRODUCTION

The Self Test T25 object runs self-test routines in the device to find faults in the sense lines and electrodes. The Self Test T25 object runs a series of test sequences. As soon as the first failure is found, the test run stops and the object reports a message. See [Section 6.3.4 “Messages”](#) for details of the test messages.

The following tests can be run:

- Analog power test. This tests that AVdd power is present.
- Pin fault test. This tests the sense pins (X0 to X11 and Y0 to Y11) on the device. This allows low-resistance shorts (line-to-line, line-to-power and line-to-GND) to be detected, as well as resistive line-to-line shorts. This test also tests for shorts on the driven shield line. The pin fault tests are configured using the PINDWELLUS and PINTHR fields.
- Signal limit tests. These test the signals from each of the touch objects on the device. These tests are run per touch object and not for the entire touchscreen matrix. The user must specify the expected minimum and maximum level for each touch object under test. The signal limit tests are configured using the UPSIGLIM, LOSIGLIM and SIGRANGELIM fields.

To run a test, the CMD field is set to the code of the test to be run. The Self Test T25 object then immediately runs the test once and then stops. At the end of the test, the CMD field is cleared. If reporting is enabled (see the “CTRL Field”), the Self Test T25 object also sends a report message with the result of the test (see [Section 6.3.4 “Messages”](#)).

The tests can be called at the following times:

- For pin fault testing: when the sense lines and driven shield line are not in use.
- For signal limit testing: at any time after acquisition when the signals are stable (for example, not during calibration). A signal test should not be performed when Optimal Integration is being used in the Shieldless T56 object (see [Section 5.6 “Shieldless T56 Object”](#)). Doing so could result in the test failing as the signal level will be lower than when Optimal Integration is not being used. Optimal Integration should be disabled before the signal limit test is run.
- For all other tests: at any time.

If an error is found, the calibration command in the Command Processor T6 object should be used once the error has been cleared. Note that if the Analog Power Test fails, it will also be necessary to perform a power cycle on the product otherwise the device may still report a pin fault issue.

In addition to the above tests, the device runs an initial pin fault test:

- The initial pin fault test is run whenever the device is reset. The initial pin fault test runs a simple test to check whether any high voltage X drive lines are shorted to ground, another X line or any Y line. This detects shorts that would cause damage to the device if a high voltage is run through the X line.

If there is no problem and the test succeeds, the device continues to its acquisition and communications routines as normal. If the test fails, a message is sent but the device does not enter its acquisition routines. This allows the user to reset the device in the event of any problems during development. Note that the initial pin fault test is run, and the pass or failure message sent, *regardless of whether Self Test T25 is enabled or reporting is enabled*.

Note that because the initial pin fault test is run irrespective of whether the Self Test T25 object is enabled or not, it is essential to choose a suitable value for the PINDWELLUS field, even if the Self Test T25 object is disabled (see [“PINDWELLUS Field”](#)).

6.3.2 CONFIGURATION

**TABLE 6-4: CONFIGURATION FOR SELF TEST T25
(SPT_SELFTEST_T25)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved						RPTEN	ENABLE
1	CMD	Test code of test to run							
2 – 5	UPSIGLIM[0]	Higher signal limit LSByte for touch object 0							
		Higher signal limit MSByte for touch object 0							
	LOSIGLIM[0]	Lower signal limit LSByte for touch object 0							
		Lower signal limit MSByte for touch object 0							
		...							

**TABLE 6-4: CONFIGURATION FOR SELF TEST T25
(SPT_SELFTEST_T25)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$(4n-2) - (4n+1)$	UPSIGLIM[n-1]	Higher signal limit LSByte for touch object $n-1$							
		Higher signal limit MSByte for touch object $n-1$							
	LOSIGLIM[n-1]	Lower signal limit LSByte for touch object $n-1$							
		Lower signal limit MSByte for touch object $n-1$							
$(4n+2)$	PINDWELLUS	Pin fault test pin dwell time							
$(4n+3) - (4n+4)$	SIGRANGELIM[0]	Signal range limit LSByte for touch object 0							
		Signal range limit MSByte for touch object 0							
		...							
$(6n+1) - (6n+2)$	SIGRANGELIM[n-1]	Signal range limit LSByte for touch object $n-1$							
		Signal range limit MSByte for touch object $n-1$							
$(6n+3)$	PINTHR	Pin fault test threshold							

Note: n = number of touch objects, assigned in the following order:
 All Multiple Touch Touchscreen T100 objects
 All Key Array T15 objects

CTRL Field

ENABLE: Enables the object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

CMD Field

This field is used to send test commands to the device. Valid test commands are listed in [Table 6-5](#).

TABLE 6-5: TEST COMMANDS

Command	Description
0x00	The CMD field is set to 0x00 after test completed
0x01	Test for AVdd power present
0x12	Run the pin fault test
0x17	Run the signal limit test
0xFE	Run all the tests

Writing 0x01 to the CMD field causes the device to perform an immediate check for the presence of the AVdd power line. Note that this test is also automatically run every 200 ms if the object is enabled.

Writing 0x12 to the CMD field causes the device to run a pin fault test.

Writing 0x17 to the CMD field causes the device to run a signal limit test.

Writing 0xFE to the CMD field causes the device to run all the tests in the order above.

UPSIGLIM[] and LOSIGLIM[] Fields

These 16-bit fields specify the higher and lower signal limits for the signal tests. UPSIGLIM specifies the upper limit and LOSIGLIM specifies the lower limit. UPSIGLIM must be greater than LOSIGLIM.

When the test is run, the signals must fall between LOSIGLIM and UPSIGLIM for the test to pass. Any signal values outside this range will generate a signal limit error in the test. The limits are specified on a per touch object basis.

The values chosen for UPSIGLIM and LOSIGLIM should differ depending on whether the unit is a design prototype or a production item.

For design prototypes, UPSIGLIM and LOSIGLIM should each be set to a value between 7116 and 9616.

For production use, LOSIGLIM should never be lower than 1116 and UPSIGLIM should never be higher than 14616. The exact values, however, are determined by analyzing the data from actual sensor samples. Using the full range can cause sensors that have too much signal to pass the test.

Range: See [Table 6-6](#)

TABLE 6-6: RECOMMENDED VALUE FOR UPSIGLIM[] AND LOSIGLIM[]

Touch Object	Range		Recommended Design Values		Recommended Production Values	
	LOSIGLIM	UPSIGLIM	LOSIGLIM	UPSIGLIM	LOSIGLIM	UPSIGLIM
Key Array T15	1116	14616	7116	9616	Determined on a case-by-case basis	
Multiple Touch Touchscreen T100	1116	14616	7116	9616	Determined on a case-by-case basis	

Note: UPSIGLIM must be \geq LOSIGLIM

PINDWELLUS Field

This field configures the pin dwell time (in μ s) for the pin fault test, where 0 means use a special case of 4 x Acquisition Configuration T8 CHRGTIME (in μ s). The value for this field is determined using the Microchip Sensor Design Calculator (contact your Microchip representative for more information).

PINDWELLUS is also used by the initial pin fault test on power-up, irrespective of whether the Self Test T25 object is enabled or not. The value of this field should be set so as to allow enough time for the X lines to slew to the correct level before testing. The PINDWELLUS field should always be set to correct value for each design, regardless of whether Self Test T25 feature is used or not by the application.

Range: 0 (automatic value in clock cycles), 1 to 254 (time in μ s)

SIGRANGELIM[] Fields

This field specifies the signal range limit for the signal tests. When the test is run, the difference between the largest and smallest signal must be less than the value specified in SIGRANGELIM for the test to pass.

Range: 1 to 13500

PINTHR Field

This field specifies the Pin Thresholds used to detect shorts in the pin fault tests. Increasing the value of this threshold increases the resistance that will be detected as a short.

An exact value of PINTHR can be selected using the following formula:

$$PINTHR = \text{floor} \left(\left(\frac{R_{THR}}{R_P + R_{THR}} - 0.5 \right) \times 512 \right) + 1$$

Where $R_P = 13 \times 10^3$ and R_{THR} is the resistance below which it is desired that a short is guaranteed to be detected.

The default value of 225 equates to a threshold of 200 k Ω and this the recommended level. Attempting to detect short resistances larger than 200 k Ω is not recommended unless advised by Microchip.

In order to assist in tuning of this threshold, tuning data is provided by the Diagnostic Debug T37 and Command Processor T6 objects (see [Section 2.2 "Diagnostic Debug T37 Object"](#) and [Section 3.3 "Command Processor T6 Object"](#)). The tuning output provides minimum and maximum pin values detected during the last run of the pin fault test. Pin values greater than or equal to PINTHR result in a pass.

Range: 0 (225 = 200 k Ω), 1 to 255

6.3.3 CONFIGURATION CHECKS

The Self Test T25 object causes a configuration check to be performed in the following circumstances:

- When the object is enabled (that is, the ENABLE bit is set on in the CTRL field)
- When certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)), and the device halts until the error has been corrected. To fix the error, the object settings should be checked to verify that they are all within their allowed limits, as stated in the field descriptions.

**TABLE 6-7: CONFIGURATION CHECKS FOR SELF TEST T25
(SPT_SELFTEST_T25)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	No	None
CMD	No	No	None
UPSIGLIM[]	Yes	No	Error if UPSIGLIM[n] is below LOSIGLIM[n]
LOSIGLIM[]	Yes	No	Error if LOSIGLIM is above UPSIGLIM[n]
PINDWELLUS	No	No	None
SIGRANGELIM[]	No	No	None
PINTHR	No	No	None

Note 1: If the ENABLE bit is toggled on or off.

6.3.4 MESSAGES

The Self Test T25 object reports the test results in its message data. The message data for the Self Test T25 object is shown in [Table 6-8](#).

**TABLE 6-8: MESSAGE DATA FOR SELF TEST T25
(SPT_SELFTEST_T25)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Result code							
2 – 6	INFO	Result data							

STATUS Field

This field contains a result code that indicates the success or failure of the test. Valid codes are given in [Table 6-9](#).

TABLE 6-9: RESULT CODES

Code	Test Result
0xFE	All tests passed.
0xFD	The test code supplied in the CMD field is not associated with a valid test.
0x01	AVdd is not present. This failure is reported to the host every 200 ms.
0x12	The test failed because of a pin fault. The INFO fields indicate the first pin fault that was detected (see Table 6-10). Note that if the initial pin fault test fails, then the Self Test T25 object will generate a message with this result code on reset.
0x17	The test failed because of a signal limit fault.

INFO Field

This field contains the result data of the test. The actual data depends on which test was run, as detailed below. Note that if a test does not generate data, the INFO field consists solely of reserved bytes.

INFO Field – Analog Power Fault

The INFO field consists of reserved bytes only.

NOTE If the Analog Power Test fails, it will be necessary to perform a power cycle on the product otherwise the device may still report a pin fault issue.

INFO Field – Pin Fault

If the result was a pin fault, the INFO field has the data format shown in [Table 6-10](#).

TABLE 6-10: TEST RESULT DATA FOR A PIN FAULT

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
2	SEQ_NUM	Test sequence number							
3	X_PIN	Failing pin indication							
4	Y_PIN	Failing pin indication							

SEQ_NUM Field

The test sequence number of the test in which a fault is found. The sequence numbers and their meanings are listed in [Table 6-11](#).

TABLE 6-11: SEQUENCE NUMBERS FOR PIN FAULT TEST

Code	Test Result
0x01	Driven Ground This test detects shorts to a power rail (that is, they fail high). All pins are driven low to Ground. Note that the detection of X/Y shorts is dependent on the supply voltage levels.
0x02	Driven High This test detects shorts to Ground (that is, they fail low). This test can detect resistive shorts, but only up to ~200 kΩ. All pins are pulled high.
0x03	Walking 1 This test is similar to test sequence 01, but uses the pull-up resistors to detect resistive shorts (but only up to ~200 kΩ). All the pins are set to zero. Then, starting with the first pin, each pin in turn is pulled high (set to 1), leaving all the other pins set to zero. The effect of this operation is that a 1 bit walks along the pins.
0x04	Walking 0 This test is similar to test sequence 02, but uses the pull-up resistors to detect resistive shorts (but only up to ~200 kΩ). All the pins are set to 1. Then, starting with the first pin, each pin in turn is cleared (set to 0), leaving all the other pins set to 1. The effect of this operation is that a "0" bit "walks" along the pins.
0x07	Initial High Voltage This test detects an initial high voltage pin fault. Specifically, it detects shorts between high voltage drive lines and GND, X or Y lines that would damage the device if a high voltage acquisition were performed. All pins are driven low and then the high voltage drive lines are all pulled high one after another.

X_PIN and Y_PIN Fields

These fields indicate which pins have failed. A non zero value in X_PIN indicates the X line number plus 1 of a failed X sense pin. A non zero value in Y_PIN indicates the Y line number plus 1 of a failed Y sense pin. For example, if X5 has failed, X_PIN will read 6. Similarly, if Y3 has failed, Y_PIN will read 4.

If both fields are zero, the driven shield has failed.

If more than one pin fails, the test reports the first failed pin detected.

See [Table 6-12](#) for a summary.

TABLE 6-12: PIN FAILURE RESULTS

X_PIN Value	Y_PIN Value	Pin Failure
0	Non zero	A Y sense pin has failed. Y_PIN will be set to the number of the pin plus 1. For example, if Y3 has failed, Y_PIN will read 4.
Non zero	0	An X sense pin has failed. X_PIN will be set to the number of the pin plus 1. For example, if X5 has failed, X_PIN will read 6.

INFO Field – Signal Limit Error

If the result was a signal limit error, the INFO field has the data format shown in [Table 6-13](#).

TABLE 6-13: TEST RESULT DATA FOR A SIGNAL LIMIT ERROR

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
2	TYPE_NUM	Touch object type number							
3	TYPE_INSTANCE	Touch object type instance							

TYPE_NUM Field

The type number of the touch object for which a signal limit error was found (see [Section 1.5 “Object Table”](#)).

TYPE_INSTANCE Field

The instance number of the touch object for which a signal limit error was found.

6.3.5 DIAGNOSTIC DEBUG DATA

The diagnostic debug data modes for the Self Test T25 object is shown in [Table 6-14](#). The diagnostic debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in [Table 6-14](#).

The diagnostic debug modes are described in the following sections.

TABLE 6-14: DIAGNOSTIC DEBUG COMMANDS

T6 Command	Name	Description
0x35	Self Test Mode	The Diagnostic Debug T37 object holds the tuning data for the Self Test T25 pin fault tests. See Section 6.3.5.1 “Self Test Mode” .

6.3.5.1 Self Test Mode

[Figure 6-1](#) shows the format of the data in Self Test Mode. This reports the tuning data reported by the Self Test T25 object for the pin fault test.

FIGURE 6-1: DIAGNOSTIC DEBUG T37 FIELDS – SELF TEST MODE

Page	Data Index	Data[]
Page 0	0	Minimum measured pin value for a pin that is expected to be pulled low
	1	Maximum measured pin value for a pin that is expected to be pulled low
	2	Minimum measured pin value for a pin that is expected to be pulled high
	3	Maximum measured pin value for a pin that is expected to be pulled high
	4 – 127	Reserved

Assumes page size = 128

6.4 User Data T38 Object

6.4.1 INTRODUCTION

The User Data T38 object provides a small area on the device for the user to store their own custom data, such as the user's own version information. The size of this area varies from device to device. On the mXT144U the size of this area is 64 bytes.

Any data stored in this object will be backed up to the non-volatile memory when the Command Processor T6 object processes a backup command (see [Section 3.3 "Command Processor T6 Object"](#)).

NOTE The data is not included in the checksum reported by the Command Processor T6 after a backup has been processed.

6.4.2 CONFIGURATION

TABLE 6-15: CONFIGURATION FOR SPT_USERDATA_T38

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 – 63	DATA[]	User data							

DATA[] Field

This field contains the user's data.

6.5 Message Count T44 Object

6.5.1 INTRODUCTION

This object contains a count of the number of pending messages. This enables the host to use a direct memory access (DMA) transfer to read the messages from the Message Processor T5 object. See [Section 3.2.1 “Introduction”](#) for more information.

6.5.2 CONFIGURATION

**TABLE 6-16: CONFIGURATION FOR MESSAGE COUNT T44
(SPT_MESSAGECOUNT_T44)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	COUNT	Message count							

COUNT Field

This field specifies the number of pending messages that are ready to send.

6.6 CTE Configuration T46 Object

6.6.1 INTRODUCTION

The CTE Configuration T46 object controls how the capacitive touch engine (CTE) in the device samples the mutual capacitive touchscreen signals.

The X lines are sampled one at a time, with at least 2 analog-to-digital conversions (ADCs) performed on each X line in turn. The number of ADCs per X line is specified by the ACTVSYNCSPERX and IDLESYNCSPERX fields. The minimum ADCs per X is saturated to 2. Therefore, if less than 2 ADCs per X is specified, then 2 will be used.

The number of ADCs per X is a nominal value only. Other objects can suggest other values for the ADCs per X setting under specific circumstances, such as when external noise is detected. The mXT144U will use the highest required setting for ADCs per X, as determined by the various objects. The objects that can suggest a maximum ADCs per X settings include:

- Glove Detection T78 for use with glove touches
- Noise Suppression T72 for use when external noise has been detected

The number of pulses for each ADC can also be configured. The amount of signal measured depends on the number of pulses per ADC. The more pulses per ADC, the higher the amount of charge that is measured. The number of pulses per ADC is determined by the PULSESPPERADC field (see “[PULSESPPERADC Field](#)”).

Therefore, the total number of pulses per X line is equivalent to:

$$\text{PULSESPPERADC} \times (\text{ACTVSYNCSPERX or IDLESYNCSPERX})$$

6.6.2 CONFIGURATION

TABLE 6-17: CONFIGURATION FOR CTE CONFIGURATION T46 (SPT_CTECONFIG_T46)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved		INTMODE		Reserved			
1	MODE	Sensor mode							
2	IDLESYNCSPERX	Number of ADCs per X when idle							
3	ACTVSYNCSPERX	Number of ADCs per X when active							
4	Reserved	Reserved							
5	PULSESPPERADC	Number of pulses for each ADC							
6	Reserved	Reserved							
7	Reserved	Reserved							
8	Reserved	Reserved							
9	Reserved	Reserved							
10	Reserved	Reserved							
11	INRUSHCFG	Reserved					YRESISTOR		

CTRL Field

INTMODE: Configures the Integration Mode.

NOTE This control is for use only during design, under the guidance of Microchip field engineers, to help tune the product for characterization. It should not be used in production. This control should be left at its default value of 0 unless advised otherwise by Microchip.

INTMODE Range: See [Table 6-18](#)

TABLE 6-18: INTMODE VALUES

Value	Description
0	Normal integration (default)
1	Positive-only integration
2	Negative-only integration
3	Reserved

MODE Field

This field specifies the touchscreen sensor (XY) mode to be used.

Range: See [Table 6-19](#)

TABLE 6-19: MODE VALUES

Code	Mode
0	12 X by 12 Y

IDLESYNCSPERX Field

This field sets the number of ADCs per X when idle. The range for this field is 1 to 255, where a value of 0 means 8. See [Section 6.6.1 “Introduction”](#) for information on how this field helps determine the number of ADCs per X.

Range: 0 (8), 1 to 255 (number of ADCs)

ACTVSYNCSPERX Field

This field sets the number of ADCs per X when active. The range for this field is 1 to 255, where a value of 0 means 8. See [Section 6.6.1 “Introduction”](#) for information on how this field helps determine the number of ADCs per X.

Range: 0 (8), 1 to 255 (number of ADCs)

PULSESPERADC Field

This field determines the number of pulses for each ADC conversion. The range for this field is 0 to 3, which represents 1 to 4 pulses.

Range: 0 to 3 (number of pulses minus 1)

INRUSHCFG Field

This field controls the selection of the Inrush Current Resistors for mutual touch measurements.

YRESISTOR: Selects the Y line inrush resistors for all the Y lines.

YRESISTOR Range: See [Table 6-20](#).

TABLE 6-20: YRESISTOR VALUES

YRESISTOR Setting	Resistor Value
0	50 Ω (default)
1	71 Ω
2	117 Ω
3	180 Ω
4	276 Ω
5	424 Ω
6	651 Ω
7	1 k Ω

6.6.3 CONFIGURATION CHECKS

The CTE Configuration T46 object causes a configuration check to be performed in the following circumstances:

- When certain fields are changed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)), and the device halts until the error has been corrected. To fix the error, the object settings should be checked to verify that they are all within their allowed limits, as stated in the field descriptions.

**TABLE 6-21: CONFIGURATION CHECKS FOR CTE CONFIGURATION T46
(SPT_CTECONFIG_T46)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	No	No	None
MODE	Yes	No	None
IDLESYNCSPERX	Yes	No	None
ACTVSYNCSPERX	Yes	No	None
PULSESPERADC	Yes	Yes	Error if out of range
INRUSHCFG	No	Yes	None

6.6.4 MESSAGES

The message data for the CTE Configuration T46 object is shown in [Table 6-22](#).

**TABLE 6-22: MESSAGE DATA FOR CTE CONFIGURATION T46
(SPT_CTECONFIG_T46)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved							CHKERR

STATUS Field

CHKERR: If set, this error implies that the device has not been correctly calibrated in the factory.

6.7 Timer T61 Object

6.7.1 INTRODUCTION

A Timer T61 object provides a timer that can be set up to trigger either a single message after a delay or a stream of regular messages. When the timer expires, a message is sent in the appropriate manner. The timer has two modes, set by the MODE field:

- **Single trigger mode:** Triggers a single message after a specified period.
- **Repeat trigger mode:** Triggers repeat messages at a specified interval.

There are 6 instances of the Timer T61 object on the mXT144U.

6.7.2 CONFIGURATION

**TABLE 6-23: CONFIGURATION FOR TIMER T61
(SPT_TIMER_T61)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved						RPTEN	ENABLE
1	CMD	Timer command							
2	MODE	Reserved							MODE
3	PERIOD	Timer period LSByte							
4		Timer period MSByte							

CTRL Field

ENABLE: Enables or disables the use of the Timer T61 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Enables or disables the Timer T61 object to report messages. Reporting is enabled if set to 1, and disabled if set to 0.

CMD Field

This field allows command to be given to the timer. See [Table 6-24](#) for a list of available commands.

TABLE 6-24: TIMER COMMANDS

Command	Description
1	Start the timer
2	Stop the timer
3	Force the timer to send a message

This field is cleared to zero once the command has been processed. Note that the host should write a command to this field only if the field is zero (that is, the previous command has been processed), otherwise the previous command will be overwritten.

MODE Field

MODE: Configures the timer mode. If the mode is set to zero, the timer operates in single trigger mode. If the mode is set to 1, the timer operates in repeat trigger mode.

Range: 0 (single trigger mode), 1 (repeat trigger mode)

PERIOD Field

This field specifies the timer period in 1 ms, where a value of 0 means 1000 ms.

Range: 0 (1000 ms), 1 to 65535 (period in ms)

6.7.3 MESSAGES

The message data for the Timer T61 object is shown in [Table 6-25](#).

TABLE 6-25: MESSAGE DATA FOR TIMER T61
(SPT_TIMER_T61)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	ELAPSED	START	STOP	FORCERPT	Reserved			RUNNING
2	CNTVAL	Current timer count value LSByte							
3		Current timer count value MSByte							

STATUS Field

RUNNING: The timer is running.

FORCEREPT: The timer has been forced to report via the CMD configuration field.

STOP: The timer has stopped.

START: The timer has started to run.

ELAPSED: The specified period in the PERIOD configuration field has elapsed.

CNTVAL Field

This field reports the current timer count value.

6.8 Serial Data Command T68 Object

6.8.1 INTRODUCTION

The Serial Data Command T68 object provides an interface for the host driver to deliver various data sets to the device, differentiated by a data type field.

The Serial Data Command T68 object supports the data types listed in [Table 6-26](#).

TABLE 6-26: DATA TYPE CODES

Code	Description
0x0005	Product Data Store (PDS)

6.8.1.1 PDS Data Type

This data type can be used to store product-specific data during production. This can be retrieved later using the Diagnostic Debug T37 object (see [Section 2.2 “Diagnostic Debug T37 Object”](#)). The format of this data is determined by the user. The PDS data can contain a maximum of 60 bytes.

6.8.2 CONFIGURATION

TABLE 6-27: CONFIGURATION FOR SPT_SERIALDATACOMMAND_T68

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved						RPTEN	ENABLE
1 – 2	Reserved	Reserved							
3 – 4	DATATYPE	Data type LSByte							
		Data type MSByte							
5	LENGTH	Length of active data in array							
6	DATA[0]	Data byte 0							
...									
6+ <i>n</i> –1	DATA[<i>n</i> –1]	Data byte <i>n</i> –1							
6+ <i>n</i>	CMD	Reserved					COMMAND		
6+ <i>n</i> +1 – 6+ <i>n</i> +2	Reserved	Reserved							

Note: n is the data size (64 in mXT144U)

CTRL Field

ENABLE: Enables or disables the use of the Serial Data Command T68 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Enables or disables the Serial Data Command T68 object to report messages. Reporting is enabled if set to 1, and disabled if set to 0.

DATATYPE Field

This field specifies the type of the data and the actions to be executed upon reception.

LENGTH Field

This field specifies the length of the data to be processed within the data field. Note that a length of zero is legal since this may make sense when handling some data types. A length greater than the size of the data array will result in an error message and the state machine will reset.

NOTE Due to the way the CRC24 is calculated, all but the last buffer should contain an even number of bytes. It is recommended that where multiple buffers are required, all but the last buffer should be the maximum size.

DATA Fields

These fields contain the input data.

CMD Field

COMMAND: This field is used by the host to indicate that the data is ready to be processed. The LENGTH and the DATA fields must be written before the command is written to the Serial Data Command T68 object. After the host writes the command, the device will detect and clear the command to zero before processing the data.

Once the data is processed a message indicating the successful processing is sent to the host. The host must wait for this message before sending the next command.

TABLE 6-28: SERIAL DATA COMMANDS

Command	Value	Description
NONE	0x00	No command
START	0x01	Start of the data set
CONTINUE	0x02	Continuation of the data set which does not fit in a single DATA[] buffer
END	0x03	End of the data set

All transactions are of the following form:

One START command

Zero or more CONTINUE commands

One END command

No other sequence is supported.

The host driver should calculate the size of the DATA array and the address of the CMD field by using the size and address of the object from the Information Block at the start of the device's memory map.

DATA[] size = object size – 9

CMD address = object address + 6 + DATA[] size

6.8.3 MESSAGES

The message data for the Serial Data Command T68 object is shown in [Table 6-29](#).

**TABLE 6-29: MESSAGE DATA FOR SERIAL DATA COMMAND T68
(SPT_SERIALDATACOMMAND_T68)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved				Error code			
2 – 4	CHECKSUM	Checksum bits 0– 7							
		Checksum bits 8 – 15							
		Checksum bits 16 – 23							

STATUS Field

This field reports the status of the Serial Data Command T68 object, as shown in [Table 6-30](#).

TABLE 6-30: ERROR CODES

Value	Description
0x00	Success/No error
0x01	Command supplied in CMD.COMMAND is out of sequence
0x02	Supplied DATATYPE value is not supported
0x03	Supplied LENGTH value exceeds length of DATA[] array
0x04	More bytes supplied than can be accommodated by this data type
0x05	Data content is invalid
0x0F	The action could not be completed due to an error outside of this object

CHECKSUM Field

This field reports a 24-bit CRC calculated from all data bytes received since the START command.

6.8.4 DIAGNOSTIC DEBUG DATA

Diagnostic debug data for the following data modes can be viewed using the Diagnostic Debug T37 object:

- Product Data Store

See [Section 2.2 “Diagnostic Debug T37 Object”](#) for more details.

6.9 Dynamic Configuration Controller T70 Object

6.9.1 INTRODUCTION

A Dynamic Configuration Controller T70 object provides a flexible way for the host to change configurations during run-time in response to a trigger. The configurations are defined for a specific event and when that event is triggered, the associated configuration settings are changed. There are 20 instances of the Dynamic Configuration Controller T70 object on the MXT144U.

Events can take two forms:

- Double edge event

A double edge event is an event with an on and off state (such as a touch) and can therefore be considered to have a rising edge and a falling edge. The rising edge is the positive state change at the start of the event and the falling edge is the negative state change at the end of the event.

For example, a noise detection event typically has two edges: the first (rising) edge is a change from a “no noise” to a “noise detected” state, and the second (falling) edge is the change back from a “noise detected” to a “no noise” to state.

Host defined events have the behavior of double edge events.

- Single edge event

A single edge event is an event with a positive state change only.

For example, a calibration command is a single edge event.

A Dynamic Configuration Controller T70 object can be configured to respond to the following:

- A single edge event
- A single edge (rising or falling) of a double edge event
- Both edges of a double edge event

The ENUMID of the EVENT field is used to specify which event to trigger on.

Alternative sets of configuration settings can be read from either the Non-volatile memory (NVM) or from the Dynamic Configuration Container T71 object (see [Section 6.10 “Dynamic Configuration Container T71 Object”](#)), as selected by the ALTSRC bit of the CTRL field. If the settings are to be read from the Dynamic Configuration Container T71 object, the SRCOFFSET field specifies the offset to use.

Note that, in the case of the Command Processor T6, the only byte that can be loaded with configuration data is the CALIBRATE field. That is, If the OBJTYPE is 6 (Command Processor T6), then DSTOFFSET must be 2 and LENGTH must be 0 (1), otherwise a configuration error is generated. All other objects in the chip, including the Dynamic Configuration Controller T70 itself, can be loaded with new configuration data.

The BACKUPNV Field of the Command Processor T6 object supports commands that help avoid the issue of a Dynamic Configuration Controller T70 object modifying settings before they are backed up. The host can also restore a configuration from the NVM using the Command Processor T6 object. The restore function does not require a reset. See [Section 3.3 “Command Processor T6 Object”](#) for more details.

To set up and use a Dynamic Configuration Controller T70 object:

1. Stop the Dynamic Configuration Controller T70 object using the BACKUPNV field in the Command Processor T6 object (See [Section 3.3.2 “Configuration”](#)).
2. Write the intended settings to the device and back up the configuration to NVM.

[Figure 6-2](#) shows an example of using a Dynamic Configuration Controller T70 object to set up a double edge trigger for a host event (EVENT HOSTDEF = 1) that loads alternative Multiple Touch Touchscreen T100 settings.

In this case, CTRL ALTSRC is set to 0 so that the configuration data is read from the Dynamic Configuration Container T71 object on the rising edge (event start) and from NVM on the falling edge (event end). The configuration data consists of 25 bytes of the Multiple Touch Touchscreen T100 settings, starting from byte 28 (GAIN).

FIGURE 6-2: EXAMPLE HOST EVENT – LOADING ALTERNATIVE MULTIPLE TOUCH TOUCHSCREEN T100 SETTINGS

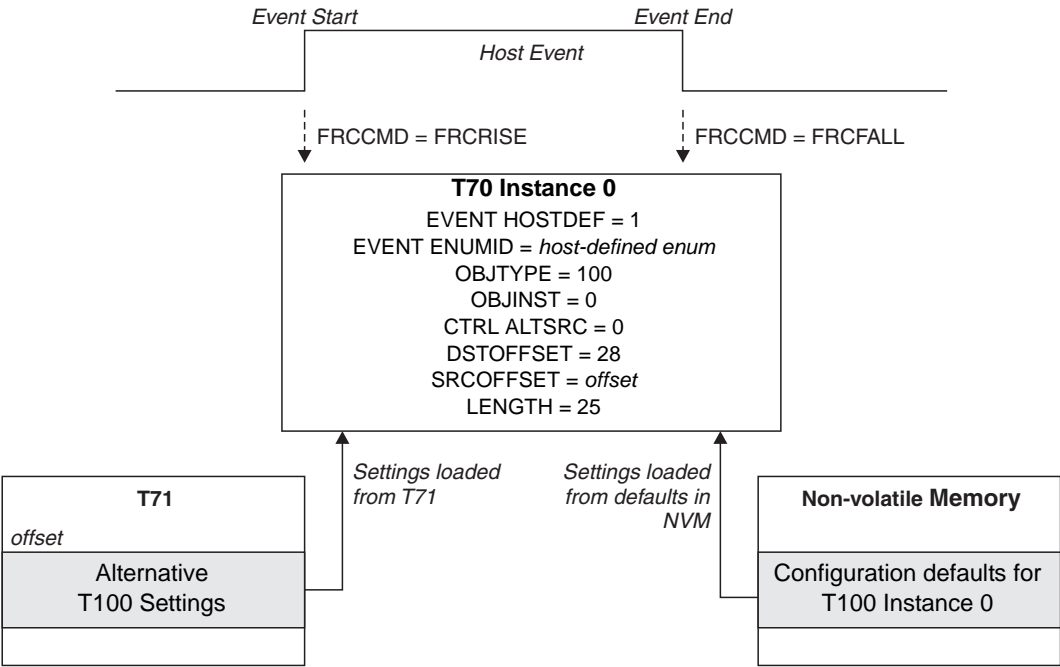
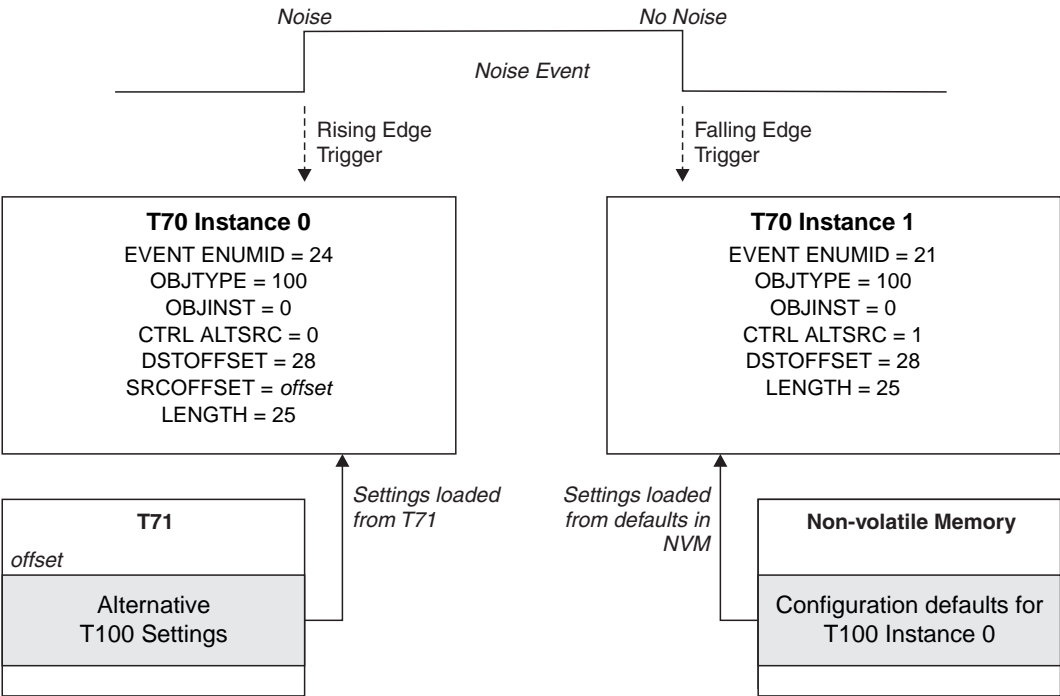


Figure 6-3 shows an example of using two separate Dynamic Configuration Controller T70 object instances to set up two triggers for noise events: one on the rising edge (EVENT ENUMID = 24) on the transition from the STABLE to NOISY state, and one on the falling edge (EVENT ENUMID = 21) on the transition from the NOISY to STABLE state.

FIGURE 6-3: EXAMPLE SINGLE EDGE EVENTS – LOADING MULTIPLE TOUCH TOUCHSCREEN T100 SETTINGS IN RESPONSE TO NOISE



In this case, alternative settings are loaded for Multiple Touch Touchscreen T100 at the start of the noise event and NVM settings reloaded at the end of the noise event. As with [Figure 6-2](#), the configuration data consists of 25 bytes of the Multiple Touch Touchscreen T100 settings, starting from byte 28 (GAIN).

6.9.2 CONFIGURATION

**TABLE 6-31: CONFIGURATION FOR DYNAMIC CONFIGURATION CONTROLLER T70
(SPT_DYNAMICCONFIGURATIONCONTROLLER_T70)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved			FRCCMD		ALTSRC	RPTEN	ENABLE
1	EVENT	ENUMID (LSByte)							
2		HOSTDEF	ENUMID (MSbits)						
3	OBJTYPE	Object type element							
4	Reserved	Reserved							
5	OBJINST	Object instance							
6	DSTOFFSET	Destination offset							
7	SRCOFFSET	Source offset LSByte							
8		Source offset MSByte							
9	LENGTH	Length offset							

CTRL Field

ENABLE: Enables or disables the use of the Dynamic Configuration Controller T70 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Enables or disables the Dynamic Configuration Controller T70 object to report messages. Reporting is enabled if set to 1, and disabled when set to 0.

ALTSRC: This bit allows the host to select the location of the source data (see [Table 6-32](#)). The source data is selected from either the Dynamic Configuration Container T71 object with an associated offset, as defined in the SRCOFFSET field, or from the corresponding backed-up area in NVM.

TABLE 6-32: ALTSRC BIT – CONFIGURATION DATA SOURCES

Event Type	ALTSRC = 1	ALTSRC = 0
Single Edge Triggered Event	Configuration data read from NVM.	Configuration data read from Dynamic Configuration Container T71
Double Edge Triggered Event	Configuration data read from NVM on rising edge and from Dynamic Configuration Container T71 on falling edge	Configuration data read from Dynamic Configuration Container T71 on rising edge and from NVM on falling edge

FRCCMD: The FRCCMD bits are used to allow the host to trigger the event configured in the EVENT Field. The FRCRise and FRCFall commands should be used accordingly to trigger on the corresponding edge of the configured event.

NOTE If more than one instance of the Dynamic Configuration Controller T70 are configured to respond to the same event, using the FRCCMD bits to force an event to be triggered on one instance will force all configured instances to respond as well.

[Table 6-33](#) gives the values for the FRCCMD bits.

TABLE 6-33: FRCCMD COMMAND CODES

FRCCMD	Command	Description
00	None	No trigger
01	FRCRise	Triggers the event on a rising edge.
10	FRCFall	Triggers the event on a falling edge.
11	Reserved	—

EVENT Field

HOSTDEF: This field allows the host to specify the clarity of the event. A device defined enumeration is specified if set to 0, and a host defined enumeration is specified if set to 1. This field must be used along with the FRCCMD to trigger the host defined events. Host defined events have the behavior of double-edged events.

ENUMID: This field configures the event enumeration ID that is used to trigger an event for a particular instance of the Dynamic Configuration Controller T70 object. [Table 6-34](#) lists the standard ENUMIDs available on the mXT144U. All other values have an undefined behavior unless the HOSTDEF bit is set to 1, in which case the ENUMID is a host-defined value.

NOTE When an event that is outside the event list in [Table 6-34](#) is specified without using the HOSTDEF field, then undefined behavior may occur.

TABLE 6-34: EVENT ENUMERATIONS

ENUMID	Description	Event Type
0	None No action.	—
1	User Calibration Command Calibration is requested by user through the Command Processor T6 object. See also event enumerations 2 and 41	Single Edge
2	User or System Calibration Command A calibration has occurred. The calibration may be invoked by the system or by a user request through the Command Processor T6 object. See also event enumerations 1 and 41	Single Edge
10	Touch Detect (High) Multiple Touch Touchscreen T100 (instance 0) goes from being not touched to being touched.	Single Edge
11	Touch Detect Release (Low) Multiple Touch Touchscreen T100 (instance 0) goes from being touched to not touched.	Single Edge
12	Touch Detect Change (High or Low) Multiple Touch Touchscreen T100 (instance 0) either goes from being not touched to touched, or from being touched to not touched.	Double Edge
13	Timer 0 Elapsed Instance 0 of the Timer T61 object has elapsed. See also event enumerations 14 and 28 to 31	Single Edge
14	Timer 1 Elapsed Instance 1 of the Timer T61 object has elapsed. See also event enumerations 13 and 28 to 31	Single Edge
18	Touch Suppression On (High) Touch Suppression T42 (instance 0) has started touchscreen suppression.	Single Edge
19	Touch Suppression Off (Low) Touch Suppression T42 (instance 0) has stopped touchscreen suppression.	Single Edge
20	Touch Suppression Change (High or Low) Touch Suppression T42 (instance 0) has either started the touchscreen suppression or stopped touchscreen suppression.	Double Edge
21	Noise Suppression STABLE State, Dual X Off Noise Suppression T72 has changed to the STABLE state with Dual X disabled.	Single Edge
22	Noise Suppression STABLE State, Dual X On Noise Suppression T72 has changed to the STABLE state with Dual X enabled.	Single Edge
23	Noise Suppression NOISY State, Dual X Off Noise Suppression T72 has changed to the NOISY state with Dual X disabled.	Single Edge

TABLE 6-34: EVENT ENUMERATIONS (CONTINUED)

ENUMID	Description	Event Type
24	Noise Suppression NOISY State, Dual X On Noise Suppression T72 has changed to the NOISY state with Dual X enabled.	Single Edge
25	Noise Suppression VERY_NOISY State, Dual X Off Noise Suppression T72 has changed to the VERY_NOISY state with Dual X disabled.	Single Edge
26	Noise Suppression VERY_NOISY state, Dual X On Noise Suppression T72 has changed to the VERY_NOISY state with Dual X enabled.	Single Edge
27	Acquisition Antitouch Calibration Request Acquisition Configuration T8 has requested a calibration due to an antitouch timeout.	Single Edge
28	Timer 2 Elapsed Instance 2 of the Timer T61 object has elapsed. See also event enumerations 13, 14, and 29 to 31	Single Edge
29	Timer 3 Elapsed Instance 3 of the Timer T61 object has elapsed. See also event enumerations 13, 14, 28, 30 and 31	Single Edge
30	Timer 4 Elapsed Instance 4 of the Timer T61 object has elapsed. See also event enumerations 13, 14, 28, 29 and 31	Single Edge
31	Timer 5 Elapsed Instance 5 of the Timer T61 object has elapsed. See also event enumerations 13, 14, and 28 to 30	Single Edge
41	System Calibration Command A system calibration has occurred (that is, a calibration that has been invoked by the system not the user). See also event enumerations 1 and 2.	Single Edge
111	Moisture Processing Touch Detect (High) Retransmission Compensation T80 moisture processing has detected a touch.	Single Edge
112	Moisture Processing Touch Release (Low) Retransmission Compensation T80 moisture processing has detected a touch release.	Single Edge
113	Moisture Processing Touch Detect Change (High or Low) Retransmission Compensation T80 moisture processing has detected a touch or a touch release.	Double Edge
114	Moisture Processing Moisture Detect (High) Retransmission Compensation T80 moisture processing has detected the presence of moisture.	Single Edge
115	Moisture Processing Moisture Detect Stop (Low) Retransmission Compensation T80 moisture processing no longer detects the presence of moisture.	Single Edge
116	Moisture Processing Moisture Detect Change (High or Low) Retransmission Compensation T80 moisture processing has changed to detecting the presence of moisture or no longer detects moisture.	Double Edge
117	Moisture Processing Touch Detect in Moisture (High) Retransmission Compensation T80 moisture processing has detected a touch in moisture.	Single Edge
118	Moisture Processing Touch Release in Moisture (Low) Retransmission Compensation T80 moisture processing has detected a touch release in moisture.	Single Edge

TABLE 6-34: EVENT ENUMERATIONS (CONTINUED)

ENUMID	Description	Event Type
119	Moisture Processing Touch in Moisture Detect Change (High or Low) Retransmission Compensation T80 moisture processing has detected a touch or a touch release in moisture.	Double Edge
120	Moisture Processing Degraded Mode (High) Retransmission Compensation T80 moisture processing has changed from normal mode to degraded mode.	Single Edge
121	Moisture Processing Degraded Mode (Low) Retransmission Compensation T80 moisture processing has changed from degraded mode to normal mode.	Single Edge
122	Moisture Processing Degraded Mode Change (High or Low) Retransmission Compensation T80 moisture processing has changed either from normal mode to degraded mode or from degraded mode to normal mode.	Double Edge
132	Retransmission Compensation (High) Retransmission compensation is actively reconstructing touches.	Single Edge
133	Retransmission Compensation (Low) Retransmission compensation is no longer actively reconstructing touches.	Single Edge
134	Retransmission Compensation (High or Low) Retransmission compensation has changed to either actively reconstructing touches or to no longer actively reconstructing touches.	Double Edge
159	Lens Bending T65 Excessive Force Detect (High) – Instance 0 The force measured in Lens Bending T65 instance 0 has grown above the active excessive force threshold.	Single Edge
160	Lens Bending T65 Excessive Force Release (Low) – Instance 0 Force measured in Lens Bending T65 instance 0 has dropped below the active excessive force threshold.	Single Edge
161	Lens Bending T65 Excessive Force Change (High or Low) – Instance 0 Force measured in Lens Bending T65 instance 0 has grown above or has dropped below the active excessive force threshold.	Double Edge
162	Lens Bending T65 Excessive Force Detect (High) – Instance 1 Force measured in Lens Bending T65 instance 1 has grown above the active excessive force threshold.	Single Edge
163	Lens Bending T65 Excessive Force Release (Low) – Instance 1 Force measured in Lens Bending T65 instance 1 has dropped below the active excessive force threshold.	Single Edge
164	Lens Bending T65 Excessive Force Change (High or Low) – Instance 1 Force measured in Lens Bending T65 instance 1 has grown above or has dropped below the active excessive force threshold.	Double Edge
165	Lens Bending T65 Excessive Force Detect (High) – Instance 2 The force measured in Lens Bending T65 instance 2 has grown above the active excessive force threshold.	Single Edge
166	Lens Bending T65 Excessive Force Release (Low) – Instance 2 Force measured in Lens Bending T65 instance 2 has dropped below the active excessive force threshold.	Single Edge
167	Lens Bending T65 Excessive Force Change (High or Low) – Instance 2 Force measured in Lens Bending T65 instance 2 has grown above or has dropped below the active excessive force threshold.	Double Edge
All other values	Reserved; undefined behavior may occur	–

OBJTYPE Field

This field specifies the Object Type of the object for which the configuration settings should be replaced when a specified event is triggered. This is the number after the “T” suffix at the end of the object’s name.

Range: Depends on the objects available (see [Section 1.6.2 “Object Instances”](#))

OBJINST FIELD

This field specifies the Object Instance for which the configuration settings should be replaced when a specified event is triggered.

Range: Depends on the number of instances for the object (see [Section 1.6.2 “Object Instances”](#))

DSTOFFSET

This field specifies the starting Destination Offset of the first field in the configuration area for the object for which the settings should be replaced. In other words, this is the first byte for which the settings are to be replaced. For example, to start at the GAIN field of the Multiple Touch Touchscreen T100 object, DSTOFFSET would be 28.

Range: 0 to 255

SRCOFFSET Field

This field specifies the starting offset from which to copy the source data in the Dynamic Configuration Container T71 object.

Range: 0 to Dynamic Configuration Container T71 size minus 1

LENGTH Field

This field specifies the length of the data to be written minus 1 (that is, length as an offset to the last byte).

Range: 0 to 255 (1 to 256 bytes)

6.9.3 CONFIGURATION CHECKS

A Dynamic Configuration Controller T70 object causes a configuration check in the following circumstances:

- When the object is enabled
- If the object is enabled, when certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, a setting is outside of its allowed range). This is signalled to the host and the device is halted until the error has been corrected. To fix an error, check that all object settings are within their allowed limits, as stated in the field descriptions.

Note that a configuration error is generated in the following situation:

- If the OBJTYPE is equal to 6 (Command Processor T6) and either DSTOFFSET is not equal to 2 or LENGTH is not equal to 0 (1). That is, if the destination object is the Command Processor T6, and the destination is not the CALIBRATE field (only), a configuration error results.

TABLE 6-35: CONFIGURATION CHECKS FOR DYNAMIC CONFIGURATION CONTROLLER T70 (SPT_DYNAMICCONFIGURATIONCONTROLLER_T70)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes	No	None
EVENT	Yes	No	None
OBJTYPE	Yes	No	Error if specified object type is not supported.
OBJINST	Yes	No	Error if specified object instance is out of range.

**TABLE 6-35: CONFIGURATION CHECKS FOR DYNAMIC CONFIGURATION CONTROLLER T70
(SPT_DYNAMICCONFIGURATIONCONTROLLER_T70)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
DSTOFFSET	Yes	No	Error if specified destination offset is out of range of the specific object configuration area, or OBJTYPE = 6 and DSTOFFSET \neq 2.
SRCOFFSET	Yes	No	Error if specified address is out of range as defined in the Dynamic Configuration Container T71 object.
LENGTH	Yes	No	Error if specified LENGTH with DSTOFFSET is out of range, or LENGTH with SRCOFFSET is out of range, or OBJTYPE = 6 and LENGTH \neq 0.

6.9.4 MESSAGES

The message data for a Dynamic Configuration Controller T70 object is shown in [Table 6-36](#).

**TABLE 6-36: MESSAGE DATA FOR DYNAMIC CONFIGURATION CONTROLLER T70
(SPT_DYNAMICCONFIGURATIONCONTROLLER_T70)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved						EVENTEDGE	HANDLED

STATUS Field

HANDLED: Indicates that an event has occurred and has been handled.

EVENTEDGE: Indicates a single-edge triggered event or the rising edge of a double-edge triggered event when set to 1, or indicates a falling edge of a double-edge triggered event when set to 0.

6.10 Dynamic Configuration Container T71 Object

6.10.1 INTRODUCTION

The Dynamic Configuration Container T71 object provides an area on the device for the user to store their own sets of configuration data that can be copied to other objects during run-time. The configurations are copied based on different events, as defined in the Dynamic Configuration Controller T70 object.

6.10.2 CONFIGURATION

TABLE 6-37: CONFIGURATION FOR DYNAMIC CONFIGURATION CONTAINER T71
(SPT_DYNAMICCONFIGURATIONCONTAINER_T71)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 – (n-1)	DATA[]	Configuration Data							

Note: n is the number of bytes for the Dynamic Configuration Container T71 object (n = 200 in mXT144U)

DATA[] Field

This field contains the configuration data. See [Section 6.9 “Dynamic Configuration Controller T70 Object”](#) for details on how this data is used.

6.11 Auxiliary Touch Configuration T104 Object

6.11.1 INTRODUCTION

An Auxiliary Touch Configuration T104 object allows the setting of gain and thresholds for self capacitance measurements to generate auxiliary touch data for use by other objects.

Specifically the Auxiliary Touch Configuration T104 object generates self capacitance touch data, which is then used by the Retransmission Compensation T80 object to help with retransmission errors, water immunity and gloved touches.

See [Appendix A “Measurement Processing on mXT144U”](#) for more details on how the Auxiliary Touch Configuration T104 object operates with other objects.

6.11.2 CONFIGURATION

TABLE 6-38: CONFIGURATION FOR AUXILIARY TOUCH CONFIGURATION T104 (SPT_AUXTOUCHCONFIG_T104)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved							ENABLE
1	XGAIN	X axis gain							
2	XTCHTHR	X axis touch threshold							
3	XTCHHYST	X axis touch hysteresis							
4	XINTTHR	X axis internal threshold							
5	XINTHYST	X axis internal hysteresis							
6	YGAIN	Y axis gain							
7	YTCHTHR	Y axis touch threshold							
8	YTCHHYST	Y axis touch hysteresis							
9	YINTTHR	Y axis internal threshold							
10	YINTHYST	Y axis internal hysteresis							

CTRL Field

ENABLE: Enables or disables the use of this Auxiliary Touch Configuration T104 object. The object is enabled if set to 1, and disabled if set to 0.

The MEASALLOW field in the Acquisition Configuration T8 object must be set appropriately to allow self capacitance touches in order for the Auxiliary Touch Configuration T104 to operate correctly.

NOTE A tune should be performed using the Self Capacitance Global Configuration T109 object after enabling the Auxiliary Touch Configuration T104 object (see [“CMD Field”](#)).

XGAIN Field

This field sets the X Axis Gain. This is the gain of the analog circuits in front of the analog to digital converter (ADC) for X axis self capacitance measurements.

For more information on choosing a suitable gain value, refer to *maXTouch U Series Tuning Guide*.

The default gain mode should be sufficient for most applications, although using Gain \times 2 mode may help improve moisture immunity performance. To enable Gain \times 2 mode, and thus double the gain value, set XGAIN bit 7 to 1.

Range: 0 to 29, 0 to 58 (Gain \times 2 mode)

XTCHTHR Field

This field specifies the X Axis Touch Threshold in 8-bit deltas. This is the threshold above which X axis nodes are marked as above threshold or in detect.

Range: 0 to 255 (threshold in 8-bit deltas)

Typical: 80

XTCHHYST Field

This field specifies the X Axis Touch Hysteresis in 8-bit deltas. This is used to specify a difference in threshold which would cause nodes in detect (based upon XTCHTHR) to go out of detect.

XTCHYST is capped internally to $(XTCHTHR \times 0.75)$. A hysteresis greater than 75% of the X Axis Touch Threshold is therefore not possible. Note that, under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the X Axis Touch Threshold (XTCHTHR), as this can cause potential problems during touch lift off, double taps, and noise handling.

A value of 0 means there is no hysteresis.

Range: 0 (no hysteresis), 1 to $XTCHTHR \times 0.75$ (hysteresis in 8-bit deltas)

Typical: 20

XINTTHR Field

This field specifies the X Axis Internal Threshold in 8-bit deltas. This is the threshold above which X axis nodes are marked as above the internal threshold or in detect at a lower threshold.

Range: 0 to 255 (threshold in 8-bit deltas)

Typical: 20

XINTHYST Field

This field specifies the X Axis Internal Hysteresis in 8-bit deltas. This is used to specify a difference in threshold which would cause nodes in detect (based upon XINTTHR) to go out of detect.

XINTHYST is capped internally to $(XINTTHR \times 0.75)$. A hysteresis greater than 75% of the X Axis Internal Threshold is therefore not possible. Note that, under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the X Axis Internal Threshold (XINTTHR), as this can cause potential problems during touch lift off, double taps, and noise handling.

A value of 0 means there is no hysteresis.

Range: 0 (no hysteresis), 1 to $XINTTHR \times 0.75$ (hysteresis in 8-bit deltas)

Typical: 5

YGAIN Field

This field sets the Y Axis Gain. This is the gain of the analog circuits in front of the analog to digital converter (ADC) for Y axis self capacitance measurements.

For more information on choosing a suitable gain value, refer to *maXTouch U Series Tuning Guide*.

The default gain mode should be sufficient for most applications, although using Gain $\times 2$ mode may help improve moisture immunity performance. To enable Gain $\times 2$ mode, and thus double the gain value, set YGAIN bit 7 to 1.

Range: 0 to 29, 0 to 58 (Gain $\times 2$ mode)

YTCHTHR Field

This field specifies the Y Axis Touch Threshold in 8-bit deltas. This is the threshold above which Y axis nodes are marked as above threshold or in detect.

Range: 0 to 255 (threshold in 8-bit deltas)

Typical: 80

YTCHYST Field

This field specifies the Y Axis Touch Hysteresis in 8-bit deltas. This is used to specify a difference in threshold which would cause nodes in detect (based upon YTCHTHR) to go out of detect.

YTCHYST is capped internally to $(YTCHTHR \times 0.75)$. A hysteresis greater than 75% of the Y Axis Touch Threshold is therefore not possible. Note that, under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the Y Axis Touch Threshold (YTCHTHR), as this can cause potential problems during touch lift off, double taps, and noise handling.

A value of 0 means there is no hysteresis.

Range: 0 (no hysteresis), 1 to $YTCHTHR \times 0.75$ (hysteresis in 8-bit deltas)

Typical: 20

YINTTHR Field

This field specifies the Y Axis Internal Threshold in 8-bit deltas. This is the threshold above which Y axis nodes are marked as above the internal threshold or in detect at a lower threshold.

Range: 0 to 255 (threshold in 8-bit deltas)

Typical: 20

YINTHYST Field

This field specifies the Y axis Internal Hysteresis in 8-bit deltas. This is used to specify a difference in threshold which would cause nodes in detect (based upon YINTTHR) to go out of detect.

YINTHYST is capped internally to $(YINTTHR \times 0.75)$. A hysteresis greater than 75% of the Y Axis Internal Threshold is therefore not possible. Note that, under normal circumstances, it is not recommended to set the hysteresis to more than 25% of the Y Axis Internal Threshold (YINTTHR), as this can cause potential problems during touch lift off, double taps, and noise handling.

A value of 0 means there is no hysteresis.

Range: 0 (no hysteresis), 1 to $YINTTHR \times 0.75$ (hysteresis in 8-bit deltas)

Typical: 5

6.11.3 CONFIGURATION CHECKS

An Auxiliary Touch Configuration T104 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed

In addition, some fields may cause an automatic recalibration to be performed.

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 "Messages"](#)). The Auxiliary Touch Configuration T104 halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

TABLE 6-39: CONFIGURATION CHECKS FOR AUXILIARY TOUCH CONFIGURATION T104 (SPT_AUXTOUCHCONFIG_T104)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes	Yes ⁽¹⁾	None
XGAIN	Yes	Yes	None
XTCHTHR	Yes	No	None
XTCHHYST	Yes	No	None
XINTTHR	Yes	No	None
XINTHYST	Yes	No	None
YGAIN	Yes	Yes	None
YTCHTHR	Yes	No	None
YTCHHYST	Yes	No	None
YINTTHR	Yes	No	None
YINTHYST	Yes	No	None

Note 1: If the ENABLE bit is toggled on or off.

6.12 Self Capacitance Global Configuration T109 Object

6.12.1 INTRODUCTION

The Self Capacitance Global Configuration T109 object provides configuration for self capacitance measurements employed on the device.

6.12.2 CONFIGURATION

TABLE 6-40: CONFIGURATION FOR SELF CAPACITANCE GLOBAL CONFIGURATION T109 (SPT_SELFCAPGLOBALCONFIG_T109)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
0	CTRL	DISMEASY	DISMEASX	Reserved			SNGLENDEN	RPTEN	Reserved	
1	Reserved	Reserved								
2	CMDONRESET	Command performed on reset								
3	CMD	Command								
4	Reserved	Reserved								
5	Reserved	Reserved								
6	Reserved	Reserved								
7	Reserved	Reserved								
8	TUNECFG	Reserved					EXTRATUNITER			

CTRL Field

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

SNGLENDEN: Enables Single Ended Measurement.

NOTE This control is for use only during design, under the guidance of Microchip field engineers, to help tune the product for characterization. It should not be used in production. This control should be left at its default value of 0, unless advised otherwise by Microchip.

DISMEASX and DISMEASY: Disable Self Capacitance Measurements on X or Y axis. If a bit is set to 1, all self capacitance measurements on the corresponding axis are disabled. If a bit is set to 0, all self capacitance measurements on the corresponding axis can be made. It is not recommended to disable measurements on both axes (that is DISMEASX = 1 and DISMEASY = 1); one axis should always remain enabled.

Note that the ability to disable self capacitance measurements on the X or Y axis depends on other configuration settings:

- Acquisition Configuration T8: MEASALLOW (see "[MEASALLOW Field](#)")
- Auxiliary Touch Configuration T104: CTRL ENABLE (see "[CTRL Field](#)")
- Self Capacitance Measurement Configuration T113: CTRL ENABLE and CTRL ALTAXISEN (see "[CTRL Field](#)")

In particular, setting DISMEASX and DISMEASY should be done in conjunction with the Self Capacitance Measurement Configuration T113 CTRL ALTAXISEN control.

CMDONRESET Field

This field schedules the given command on reset. See [Table 6-41](#) for command values.

This field should be set to 5, even if the remaining bytes in this object are not configured, such as in a mutual capacitance only system.

CMD Field

This field schedules the given command.

The command field is cleared after the command has been actioned.

Range: See [Table 6-41](#)

TABLE 6-41: COMMANDS

Value	Command	Description
0	None	None.
1	Tune	Tunes the current self capacitance tuning parameters. The new settings are then used and the system performs a calibration. Self capacitance touch must be enabled for a tune to work (see Section 3.5 "Acquisition Configuration T8 Object" and Section 6.11 "Auxiliary Touch Configuration T104 Object"). In addition, appropriate measurements must be enabled for any associated parameters to be tuned (for example, in the Acquisition Configuration T8 and Self Capacitance Measurement Configuration T113 objects).
2	Store To NVM	Stores the current self capacitance tuning parameters into NVM (non volatile memory). Note that this area of NVM is outside the configuration backup region.
3	Apply From NVM	Applies the current self capacitance tuning parameters from NVM and causes a system calibration. Note that this area of NVM is outside the configuration backup region.
4	Store To Configuration RAM	Stores the current self capacitance tuning parameters into the Self Capacitance Tuning Parameters T110 object.
5	Apply From Configuration RAM	Applies the current self capacitance tuning parameters from the Self Capacitance Tuning Parameters T110 object and causes a system calibration.
6	Apply From NVM and Store To Configuration RAM	Applies the current self capacitance tuning parameters from NVM. Note that this area of NVM is outside the configuration backup region. Stores the current self capacitance tuning parameters into the Self Capacitance Tuning Parameters T110 object. Causes a system calibration.
7	Refresh Configuration RAM	Refreshes the Self Capacitance Tuning Parameters T110 object from NVM backup. Note that this area of NVM is inside the configuration backup region.

Note: NVM (non volatile memory) is outside the configuration backup region.

TUNECFG Field

This field provides additional tuning configurations.

EXTRATUNITER: Configures Extra Tuning Iterations. The range for this control is 0 to 7, where 0 means that no extra tuning iterations are required.

Range: 0 (disabled; no extra tuning), 1 to 7

6.12.3 CONFIGURATION CHECKS

TABLE 6-42: CONFIGURATION CHECKS FOR SELF CAPACITANCE GLOBAL CONFIGURATION T109 (SPT_SELFCAPGLOBALCONFIG_T109)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	No	Yes	None
CMDONRESET	No	No	None
CMD	No	No	None
TUNECFG	No	No	None

6.12.4 MESSAGES

TABLE 6-43: MESSAGE DATA FOR SELF CAPACITANCE GLOBAL CONFIGURATION T109 (SPT_SELFCAPGLOBALCONFIG_T109)

Byte	Field							Bit 1	Bit 0
0	CMD	Command processed							
1	ERRORCODE	Error code							

CMD Field

This field reports the command that was processed and has caused this message (see “CMDONRESET Field” and “CMD Field”).

ERRORCODE Field

This field reports the success/error code. An error code of 0 means that there is no error (success). Otherwise, a non-zero indicates the error resulting from the command that caused this message (see Table 6-44).

TABLE 6-44: ERROR CODES

Command	Error Codes
1 – Tune	0 – Tuning successful
	Non zero – Tuning failed with error code
2 – Store To NVM	0x20 – write error
3 – Apply From NVM	0x08 – invalid CRC
	0x10 – read error
6 – Apply From NVM and Store To Configuration RAM	0x08 – invalid CRC
	0x10 – read error

6.13 Self Capacitance Tuning Parameters T110 Object

6.13.1 INTRODUCTION

The Self Capacitance Tuning Parameters T110 object provides configuration space for a generic set of settings for self capacitance measurements employed on the device. It has a size equivalent to 2 × the number of Y lines.

NOTE This object is for use only under the guidance of Microchip field engineers. This object should be left with its default values unless advised otherwise by Microchip.

6.13.2 CONFIGURATION

TABLE 6-45: CONFIGURATION FOR SELF CAPACITANCE TUNING PARAMETERS T110
(SPT_SELFCAPTUNINGPARAMS_T110)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	PARAMS[0]	Setting 0 LSByte							
1		Setting 0 MSByte							
...									
(2 × <i>n</i>)−2	PARAMS[<i>n</i> −1]	Setting <i>n</i> −1 LSByte							
(2 × <i>n</i>)−1		Setting <i>n</i> −1 MSByte							

Note: n = Number of lines stored for this instance. Unused bytes will be reserved.

PARAMS[] Fields

These fields specify the generic settings.

Range: 0 to 65535

6.14 Self Capacitance Configuration T111 Object

6.14.1 INTRODUCTION

The Self Capacitance Configuration T111 object provides configuration for self capacitance measurements (see [Appendix A “Measurement Processing on mXT144U”](#)).

On the mXT144U there are 2 instances of the Self Capacitance Configuration T111 object (see [Table 6-46](#) for details).

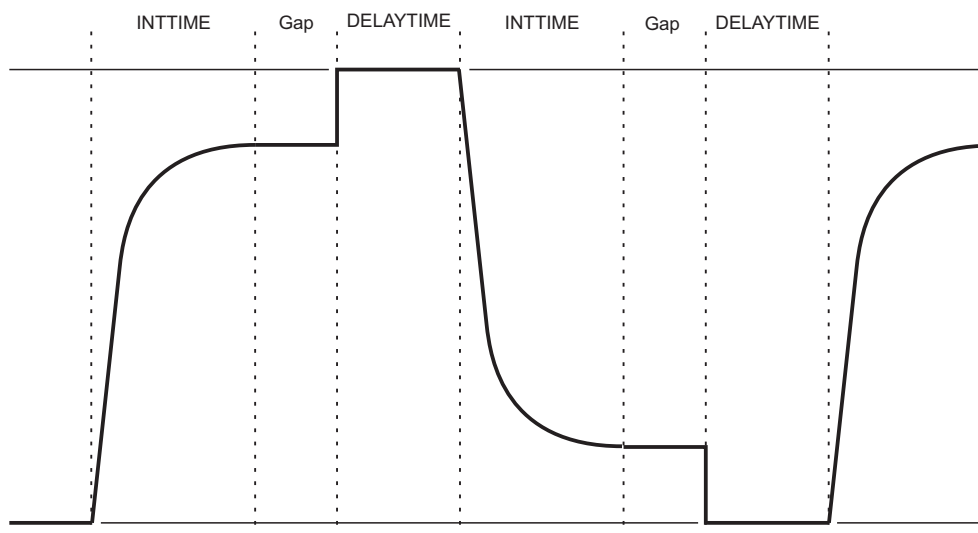
TABLE 6-46: SELF CAPACITANCE CONFIGURATION T111 INSTANCES

Instance	Purpose
Instance 0	Used to configure self capacitance touch measurements. Settings for this instance use a subset of the configuration fields (see Table 6-47).
Instance 1	Used to configure self capacitance single-ended measurements. Settings for this instance use a subset of the configuration fields (see Table 6-48).

Self capacitance measurements are required for single touch measurements when self capacitance measurements are selected as the primary measurement type. Self capacitance data is also used to augment the mutual capacitance data to help cope with retransmission and moisture issues (see [Section 5.10 “Retransmission Compensation T80 Object”](#)).

The pulse for a self capacitance measurement is outlined in [Figure 6-4](#).

FIGURE 6-4: A SELF CAPACITANCE MEASUREMENT PULSE



NOTE: The pulse shape is shown for guidance only. The shape may vary between devices, although the timing principles are the same.

Note that, in [Figure 6-4](#), Gap is the extra time provided by the associated Self Capacitance Noise Suppression T108 object.

See [Appendix A “Measurement Processing on mXT144U”](#) for more details on how the Self Capacitance Configuration T111 object operates with other objects.

The matrix axes are sampled one at a time (depending on the hardware there may be more than one burst per axis), with at least 2 analog-to-digital conversions (ADCs) performed on each axis in turn. For each axis measurement, all the matrix sense lines are burst together for both axes, although only one axis is measured.

The number of ADCs per X line is specified by the ACTVSYNCSPERL and IDLESYNCSPERL fields. The minimum ADCs per line is saturated to 2. Therefore, if less than 2 ADCs per line is specified, then 2 will be used.

The number of ADCs per line is a nominal value only. Other objects can suggest other values for the ADCs per line setting under specific circumstances, such as when external noise is detected. The mXT144U will use the highest required setting for ADCs per line, as determined by the various objects. The objects that can suggest a maximum ADCs per line settings include:

- Self Capacitance Noise Suppression T108 for use when external noise has been detected

6.14.2 CONFIGURATION

6.14.2.1 Instance 0 – Self Capacitance Touch Measurements

TABLE 6-47: CONFIGURATION FOR SELF CAPACITANCE CONFIGURATION T111 (SPT_SELFPCAPCONFIG_T111)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
0	CTRL	DISDCCORR	Reserved				LFCOMPEN	Reserved		
1	DBGCTRL	Reserved			DCEN		RAWDATAEN	SIGSEN	REFSEN	DELTASEN
2	INTTIME	Integration time								
3	DELAYTIME	Delay time								
4	IDLESYNCSPERL	Number of ADCs per sensor line when idle								
5	ACTVSYNCSPERL	Number of ADCs per sensor line when active								
6	DRIFT	Drift interval								
7	DRIFTST	Drift suspend time								
8	Reserved	Reserved								
9	CALRECSTR	Calibration recovery strength								
10	Reserved	Reserved								
11	Reserved	Reserved								
12	Reserved	Reserved								
13	Reserved	Reserved								
14	Reserved	Reserved								
15	INRUSHCFG	Reserved	XRESISTOR			Reserved	YRESISTOR			
16	ALTINTTIMEX	Alternative X axis measurement integration time								
17	ALTDELAYTIMEX	Alternative X axis measurement delay time								
18	DCDRIFT	DC drift interval								
19	Reserved	Reserved								
20	DCFILTER	Reserved		SLEWACC	SLEWEN	IIRCOEF				IIREN
21	DCCALRECSTR	DC Calibration recovery strength								
22	DCCALERRRATIO	DC Calibration recovery error ratio								
23	Reserved	Reserved								
24	Reserved	Reserved								
25	Reserved	Reserved								
26	DCGAINSF	DC gain scaling factor								
27	DCTHRX	X axis DC Self capacitance threshold								
28	DCTHRY	Y axis DC Self capacitance threshold								
29	DCIDLESLEWMIN	Idle Mode Minimum DC Slew Rate								
30	DCPROCTHRX	DC Level Processing Threshold for X								
31	DCPROCTHRY	DC Level Processing Threshold for Y								

6.14.2.2 Instance 1 – Self Capacitance Single-ended Measurements

TABLE 6-48: CONFIGURATION FOR SELF CAPACITANCE CONFIGURATION T111 (SPT_SELFPCAPCONFIG_T111)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved							
1	DBGCTRL	Reserved			DCEN	RAWDATAEN	SIGSEN	REFSEN	DELTASEN
2	INTTIME	Integration time							
3	DELAYTIME	Delay time							
4	IDLESYNCSERL	Number of ADCs per sensor line when idle							
5	ACTVSYNCSERL	Number of ADCs per sensor line when active							

**TABLE 6-48: CONFIGURATION FOR SELF CAPACITANCE CONFIGURATION T111
(SPT_SELFCAPCONFIG_T111)**

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6	Reserved	Reserved							
7	Reserved	Reserved							
8	Reserved	Reserved							
9	Reserved	Reserved							
10	Reserved	Reserved							
11	Reserved	Reserved							
12	Reserved	Reserved							
13	Reserved	Reserved							
14	Reserved	Reserved							
15	INRUSHCFG	Reserved	XRESISTOR			Reserved	YRESISTOR		
16	Reserved	Reserved							
17	Reserved	Reserved							
18	Reserved	Reserved							
19	Reserved	Reserved							
20	Reserved	Reserved							
21	Reserved	Reserved							
22	Reserved	Reserved							
23	Reserved	Reserved							
24	Reserved	Reserved							
25	Reserved	Reserved							
26	Reserved	Reserved							
27	Reserved	Reserved							
28	Reserved	Reserved							
29	Reserved	Reserved							
30	Reserved	Reserved							
31	Reserved	Reserved							

CTRL Field

LFCOMPEN: (Touch data instance Enables Low Frequency Compensation.

If this bit is set to 1, low frequency compensation is performed on the measurement. In this case, the Self Capacitance Global Configuration T109 tuning procedure will perform 3 phases of tuning for the self capacitance differential (assuming that there is enough storage space within Self Capacitance Tuning Parameters T110). Note the following:

- The number of ADCs per line should be set to a multiple of 3 (see “IDLESYNCSPEL Field” and “ACTVSYNCSPEL Field”), but the device will constrain this internally if the ADCs per line is not set appropriately.
- An additional sample is added per scan, so the amount of pulses seen is:

$$((\text{ADCs_per_line} / 3) \times 3) + 1$$

If this bit is set to 0, low frequency compensation is not applied to the measurement and the tuning procedure will run as normal. In this case, the Self Capacitance Global Configuration T109 tuning procedure will perform only a single phase of tuning.

DISDCORR: (Touch data instance only) If this bit is set to 0, the self capacitance single-ended measurement data is used to correct the DC level of the self capacitance touch data in the case when it appears that the self capacitance touch DC level is incorrectly too high and might therefore create false touches. If this bit is set to 1, this DC correction is disabled. Disabling DC Correction should only be used in cases where the self capacitance single-ended data is unreliable for this purpose.

DBGCTRL Field

This field allows different debug data types to be enabled or disabled for use with the hardware debug interface. Note that the Command Processor T6 DEBUGCTRL2 DBGOBJMODEEN bit must be set to 1 for this to work (see [“DEBUGCTRL2 Field”](#)).

DELTAEN: If this bit is set to 1, self capacitance deltas are output by the hardware debug interface.

REFSEN: If this bit is set to 1, self capacitance references are output by the hardware debug interface.

SIGSEN: If this bit is set to 1, self capacitance signals are output by the hardware debug interface.

RAWDATAEN: If this bit is set to 1, raw data values are output by the hardware debug interface.

DCEN: If this bit is set to 1, DC debug data is output by the hardware debug interface.

INTTIME Field

This field specifies the Integration Time in 24 MHz clock cycles (41.67 ns), where the default value of 0 means 36 (1.5 μ s).

Range: 0 (36), 1 to 255

DELAYTIME Field

This field specifies the Delay Time in 24 MHz clock cycles (41.67 ns). This is the length of additional time that a pulse is extended by after the integration time has completed. The default value of 0 means 36 (1.5 μ s).

Range: 0 (36), 1 to 255

IDLESYNCSPERL Field

This field specifies the number of ADCs per sensor line in idle mode (see [Section 6.14.1 “Introduction”](#) for more information).

The minimum ADCs per line is saturated to 2; so if less than 2 ADCs per X is specified, a value of 2 will be used. The default value of 0 means 8.

Range: 0 (8), 1 to 255

ACTVSYNCSPERL Field

This field specifies the number of ADCs per sensor line in active mode (see [Section 6.14.1 “Introduction”](#) for more information).

The minimum ADCs per line is saturated to 2; so if less than 2 ADCs per X is specified, a value of 2 will be used. The default value of 0 means 8.

Range: 0 (8), 1 to 255

DRIFT Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

The default behavior of this field is to provide the interval (specified in 200 ms increments) between drift operations for self capacitance based references. A drift operation samples the deltas after a cycle, and "moves" the references (per node) towards those deltas. This allows for small variations in operating conditions, such as temperature changes. The amount to drift is specific to the measurement type, but typically a drift "moves" the references by 4 to 8 16-bit delta counts.

The range is 0 to 255 in 200 ms increments, where the default value of 0 means 5 (1 second). A value of 255 (disabled) is not recommended for this field.

Range: 0 (5), 1 to 254, 255 (disabled)

DRIFTST Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field specifies the Drift Suspend Time. This is the duration for which drift is suspended after any touches have been removed. See the mutual capacitance handling of the [“DRIFTST Field”](#) for more information on the Drift Suspend Time.

DRIFTST is configured in units of 200 ms, where a value of zero means 10 (2 seconds).

Range: 0 (10), 1 to 255

CALRECSTR Field

This field configures the Calibration Recovery Strength.

The Calibration Recovery process causes an automatic calibration when the system decides that a calibrated in touch has been removed from the sensor. This field controls how aggressive that process is for self-capacitance touch data. Low values will increase the probability of a system calibration; high values will decrease this probability. A value of 255 disables this feature, and the default value of 0 means 30.

Note that the tuning data can be output through the Diagnostic Debug T37 object using the Command Processor T6 DIAGNOSTIC command (see [Section 2.2 “Diagnostic Debug T37 Object”](#) and [“DIAGNOSTIC Field”](#)). Note that this must be enabled using the Acquisition Configuration T8 CFG CALRECTUN control (see [“CFG Field”](#)).

Range: 0 (30), 1 to 254, 255 (disabled)

INRUSHCFG Field

YRESISTOR/XRESISTOR: Controls the selection of the inrush current resistors for the measurement type associated with the object instance. The value specified controls the selection for all X or Y line measurements (as appropriate). The value specified controls the selection for all X or Y lines (as appropriate).

YRESISTOR/XRESISTOR Range: See [Table 6-49](#).

TABLE 6-49: YRESISTOR AND XRESISTOR VALUES

YRESISTOR/XRESISTOR Setting	Resistor Value
0	50Ω (default)
1	71Ω
2	117Ω
3	180Ω
4	276Ω
5	424Ω
6	651Ω
7	1KΩ

ALTINTTIMEX Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field specifies the Alternative X Axis Measurement Integration Time in 24 MHz clock cycles (41.67 ns). This allows different integration times to be specified for the self capacitance touch X and Y measurements. Note that this field does not apply to self capacitance single-ended measurements.

If this field is set to zero, the INTTIME setting is used for both X and Y measurements (see [“INTTIME Field”](#)).

If this field is set to a non-zero value then the specified value is used for X measurements, and the INTTIME field setting is used for the Y measurement.

Note that, if ALTINTTIMEX is being used, it is important to make sure that the DELAYTIME and ALTDELAYTIMEX are adjusted such that the following condition is met (see [“DELAYTIME Field”](#) and [“ALTDELAYTIMEX Field”](#)):

$$\text{INTTIME} + \text{DELAYTIME} = \text{ALTINTTIMEX} + \text{ALTDELAYTIMEX}$$

Otherwise external noise handling performance will be impacted.

Range: See [Table 6-50](#)

TABLE 6-50: ALTINTTIMEX VALUES

Value	X Measurements	Y Measurements
0	The INTTIME setting is used for both X and Y measurements	
1 to 255	ALTINTTIMEX specifies an alternative integration time for X measurements	The INTTIME setting is used for Y measurements.

ALTDELAYTIMEX Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field specifies the Alternative X Axis Measurement Delay Time in 24 MHz clock cycles (41.67 ns). This allows different delay times to be specified for the self capacitance touch X and Y measurements. Note that this field does not apply to self capacitance single-ended measurements.

If this field is set to zero, the DELAYTIME setting is used for both X and Y measurements (see “[DELAYTIME Field](#)”).

If this field is set to a non-zero value then the specified value is used for X measurements, and the DELAYTIME field setting is used for the Y measurement.

Note that, if an alternative integration time is being used (see “[ALTINTTIMEX Field](#)”), ALTDELAYTIME value must be set such that the two burst frequencies match.

Range: 0 (DELAYTIME used for both X and Y measurements),
1 to 255

DCDRIFT Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field controls the DC Drift Interval for DC correction. The drift intervals for other measurement type instances are stored in other objects.

Signals can drift because of changes in the nature of the components and materials over time and temperature. It is crucial that such drift is compensated for, otherwise false detections and sensitivity shifts can occur.

Drift compensation is performed by adjusting the reference level. The rate of adjustment is performed slowly, otherwise the incorrect DC levels may introduce erroneous touch detections and calibrations. The DCDRIFT field is configured in increments of 200 ms, where the default value of 0 means 5 (1 second). A value of 255 disables drift.

Range: 0 (5), 1 to 254, 255 (disabled)

DCFILTER Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field provides the DC Filter controls.

IIREN: Enables the IIR Filter. The filter is enabled if this bit is set to 1, and disabled if it is set to 0.

IIRCOEF: Specifies the DC IIR Filter Coefficient. The units are 2s exponent of filter coefficient.

IIRCOEF Range: 0 to 7

IIRCOEF Typical: 1

SLEWEN: Enables the Slew Rate Filter. The slew rate filter is enabled if this bit is set to 1, and disabled if it is set to 0.

SLEWACC: Enables the Slew Rate Filter Accelerator. The accelerator is enabled if this bit is set to 1, and disabled if it is set to 0.

DCCALRECSTR Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field configures the DC Calibration Recovery Strength.

The DC Calibration Recovery process causes an automatic calibration when the system decides that the overall touch level has become negative. This field controls how aggressive that process is for the self capacitance DC level. Low values will increase the probability of a system calibration; high values will decrease this probability. A value of 255 disables this feature. The default value of 0 means 30.

Note that the tuning data can be output through the Diagnostic Debug T37 object using the Command Processor T6 DIAGNOSTIC command (see [Section 2.2 “Diagnostic Debug T37 Object”](#) and [“DIAGNOSTIC Field”](#)). Note that this must be enabled using the Acquisition Configuration T8 CFG CALRECTUN control (see [“CFG Field”](#)).

NOTE If DC calibration is enabled (that is, not equal to 255), then the self capacitance touch measurement must be allowed using Acquisition Configuration T8 MEASALLOW (see [“MEASALLOW Field”](#)).

Range: 0 (30), 1 to 254, 255 (disabled)

DCCALERRRATIO Field

NOTE This field is used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field configures the DC Calibration Error Ratio. This is the proportion (in units of 1/256) of the difference between the DC level estimates for the single-ended and auxiliary touch measurements. This difference is compared against the calibration recovery strength in order to trigger a calibration.

Range: 0 to 255

DCGAINSF Field

This field configures the DC Gain Scaling Factor. This is the scaling factor used in gain calculations associated with the system-provided DC level. The value is used to scale the magnitude of the input DC level so that it better matches the estimates based upon other self capacitance touch measurements.

A smaller value should be used if the DC level exceeds the estimate from other self capacitance measurements; a larger value should be used if the DC level is less than the estimate from other self capacitance measurements. It may be beneficial to re-tune this value if the gain of the self capacitance measurements is changed.

The value for this field is a signed 8-bit number, where -128 represents a scaling factor of 50%, 0 represents a scaling factor of 100% and +127 represents a scaling factor of 150%.

Range: -128 to +127

DCTHRX Field

NOTE This field is used only by the Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field configures the Self Capacitance DC Threshold for X axis measurements against which the DC estimate is compared for leaving the low power idle state.

Range: 0 to 255

DCTHRY Field

NOTE This field is used only by the Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field configures the Self Capacitance DC Threshold for Y axis measurements against which the DC estimate is compared for leaving the low power idle state.

Range: 0 to 255

DCIDLESLEWMIN Field

NOTE This field is used only by the Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

This field configures a minimum slew rate applied to the DC estimate for the measurement controlled by this Self Capacitance Configuration T111 instance. The slew rate can reduce to one count per acquisition cycle in quiet conditions and thus will increase first touch latency greatly when the idle default measurement is SELF PROX (self capacitance single ended; see "MEASIDLEDEF Field"). The slew rate must allow enough change in the estimate to allow the DC threshold to be reached. A value of approximately (3 × DCTHRX) or (3 × DCTHRY), as appropriate, should result in a first touch latency very similar to that when the default idle mode is MUTUALTCH or SELFTHCH.

Range: 0 (1), 1 to 255

DCPROCTHRX and DCPROCTHRY Fields

NOTE These fields are used only by Self Capacitance Configuration T111 instance 0 for self capacitance touch data.

These fields set the DC Level Processing Threshold for the X and Y lines. This is the threshold used for processing DC level estimates for the X and Y lines (for example, for use by the Self Capacitance Measurement Configuration T113 algorithms). If the threshold is greater than or equal to DCPROCTHRX/DCPROCTHRY, the DC level processing is done. The threshold is specified in 8-bit delta values, where the default value of 0 effectively means always perform the processing.

Range: 0 to 255

6.14.3 CONFIGURATION CHECKS

TABLE 6-51: CONFIGURATION CHECKS FOR SELF CAPACITANCE CONFIGURATION T111 (SPT_SELFCONFIG_T111)

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	No	Yes	None
DBGCTRL	No	No	None
INTTIME	No	Yes	None
DELAYTIME	No	Yes	None
IDLESYNCSPERL	No	No	None
ACTVSYNCSPERL	No	No	None
DRIFT	No	No	None
DRIFTST	No	No	None
CALRECSTR	No	No	None
INRUSHCFG	No	Yes	None
ALTINTTIMEX	No	Yes	None
ALTDELAYTIMEX	No	Yes	None
DCDRIFT	No	No	None
DCFILTER	No	No	None
DCCALRECSTR	No	No	None
DCCALERRRATIO	No	No	None
DCGAINSF	No	No	None
DCTHRX	No	No	None
DCTHRY	No	No	None

**TABLE 6-51: CONFIGURATION CHECKS FOR SELF CAPACITANCE CONFIGURATION T111
(SPT_SELFCAPCONFIG_T111)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
DCIDLESLEWMIN	No	No	None
DCPROCTHRX	No	No	None
DCPROCTHRY	No	No	None

6.15 Self Capacitance Measurement Configuration T113 Object

6.15.1 INTRODUCTION

A Self Capacitance Measurement Configuration T113 object is used to set the gain for a particular self capacitance single-ended measurement to generate data for use by other objects.

The current use of this object on the mXT144U is to make it possible for the Acquisition Configuration T8 object to trigger a recalibration when an anti-touch is detected. To this end, the Self Capacitance Measurement Configuration T113 object computes the self capacitance single-ended measurement, and the Acquisition Configuration T8 object then triggers the recalibration.

Note that the SELFPROX bit in the Acquisition Configuration T8 MEASALLOW field must also be set for the self capacitance single-ended measurements to be used (see [“MEASALLOW Field”](#)).

6.15.2 CONFIGURATION

TABLE 6-52: CONFIGURATION FOR SELF CAPACITANCE MEASUREMENT CONFIGURATION T113 (SPT_SELFPCAPMEASURECONFIG_T113)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	Reserved					ALTAXISEN	Reserved	ENABLE
1	GAINX	Gain for the X axis							
2	GAINY	Gain for the Y axis							

CTRL Field

ENABLE: Enables the use of this Self Capacitance Measurement Configuration T113 object. The object is enabled if set to 1, and disabled if set to 0.

NOTE A tune should be performed using the Self Capacitance Global Configuration T109 object after enabling the Self Capacitance Measurement Configuration T113 object (see [“CMD Field”](#)).

ALTAXISEN: Specifies on which axis of electrodes to perform the single-ended self capacitance measurement.

If this bit is set to 0, the measurement is performed on the *X* axis (that is, the axis that has the larger number of lines).

If this bit is set to 1, the measurement is performed on the *Y* axis (that is, the axis that has the smaller number of lines).

Note that setting ALTAXISEN should be done in conjunction with the Self Capacitance Global Configuration T109 CTRL DISMEASX and DISMEASY controls if they are set.

GAINX and GAINY Fields

These fields set the X or Y axis gain, as appropriate, for the self capacitance single-ended measurement.

Range: 0 to 29

6.15.3 CONFIGURATION CHECKS

A Self Capacitance Measurement Configuration T113 object causes a configuration check to be performed in the following circumstances:

- If the object is enabled, when certain fields are changed

A configuration check may determine that a configuration error has occurred (for example, if a setting is set outside of its allowed range or a conflict has occurred between two settings). This is signaled to the host (see [Section 3.3.3 “Messages”](#)). The device halts until the error has been corrected. To fix the error, check that all the object settings are within their allowed limits, as stated in the field descriptions.

**TABLE 6-53: CONFIGURATION CHECKS FOR SELF CAPACITANCE MEASUREMENT
CONFIGURATION T113
(SPT_SELFCAPMEASURECONFIG_T113)**

Field	Changing The Field Causes...		Effect of Configuration Checks On Field
	Configuration Check	Automatic Recalibration	
CTRL	Yes ⁽¹⁾	Yes	None
GAINX	Yes	Yes	None
GAINY	Yes	Yes	None

Note 1: If the ENABLE bit is toggled on or off.

6.16 Symbol Gesture Configuration T116 Object

6.16.1 INTRODUCTION

The Symbol Gesture Configuration T116 object stores the configuration data that defines the symbols to be detected by the Symbol Gesture Processor T115 object (see [Section 5.13 “Symbol Gesture Processor T115 Object”](#)). Each symbol is defined as series of strokes (touch movement) in certain directions (Left, Right, Up or Down). For completeness, a release (removal of touch) is also considered to be a type of stroke.

Note that the Symbol Gesture Configuration T116 object defines the stroke sequences only; all other aspects of drawing the strokes are configured by the Symbol Gesture Processor T115 object, such as how much movement is allowed or how long a symbol should take to draw.

6.16.2 CONFIGURATION

TABLE 6-54: CONFIGURATION FOR SYMBOL GESTURE CONFIGURATION T116 (SPT_SYMBOLGESTURECONFIG_T116)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved							
1 – <i>m</i>	SYMDATA[]	Symbol Data							

Note: *m* is the length of the symbol data block (254 on mXT144U)

SYMDATA[] Fields

These bytes contain the symbol stroke data. SYMDATA[] contains one or more symbol definition structures, where each symbol definition has the structure shown in [Table 6-55](#). The number of symbol definitions is limited only by the total number of bytes they occupy, as they must fit within the SYMDATA[] block size (254 on mXT144U).

TABLE 6-55: SYMBOL DEFINITION STRUCTURE

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	SYMCFG1	DISTMINY		DISTMINX		SYMSTRCNT			
1	SYMCFG2	ROLLSEQ	Reserved			ILLFIRSTSTR			
2	SYMCFG3	Reserved	ASPECTRATIOY			Reserved	ASPECTRATIOX		
3	SYMRPTCFG	DISRPT	RPTCODE						
4	SYMSTRSEQ[0]	Symbol Stroke Sequence[1]				Symbol Stroke Sequence[0]			
		...							
4 + (n−1)	SYMSTRSEQ[n−1]	Symbol Stroke Sequence[n+1]				Symbol Stroke Sequence[n]			

Note: $n = (\text{SYMCFG1 SYMSTRCNT} + 1) / 2$

SYMCFG1 Field

SYMSTRCNT: Defines the number of strokes which make up the symbol gesture. This in turn determines the number of bytes of memory which the symbol uses: $(\text{SYMSTRCNT} + 1) / 2 + 4$ bytes.

SYMSTRCNT Range: 0 to 15

DISTMINX/DISTMINY: Specify the minimum distance between the start and end position of a symbol relative to the symbol's size in the respective axis (X/Y). If the distance is smaller than DISTMINX/DISTMINY, the symbol is considered invalid.

DISTMINX/DISTMINY Range: See [Table 6-56](#)

TABLE 6-56: DISTMINX/DISTMINY VALUES

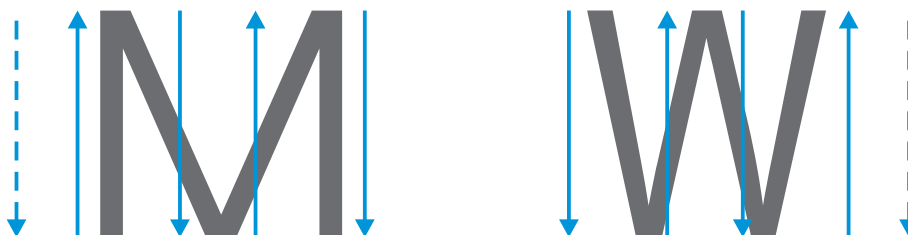
POSDISTMIN	Pass Criterion
0	No constraint
1	Distance \geq size / 2
2	Distance \geq size / 4
3	Distance \geq size / 8

Note: Distance is the difference between Start and End positions in the X/Y axis.

SYMCFG2 Field

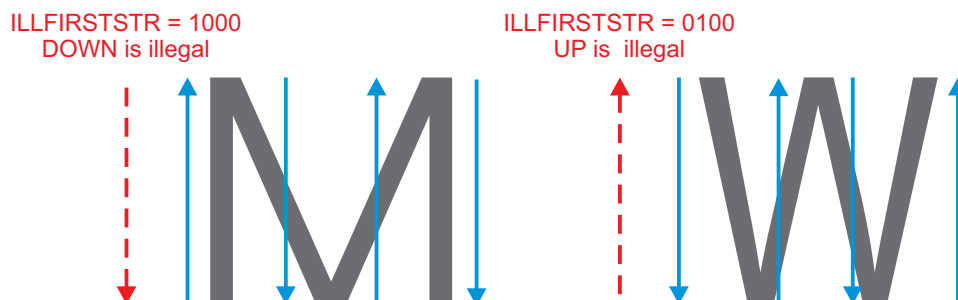
ILLFIRSTSTR: Specifies the Illegal First Strokes (leading strokes) that are to be ignored. Some symbol matches can be excluded if the first movement is in a particular direction (for example, to distinguish between “M” and “W”).

FIGURE 6-5: WITHOUT ILLFIRSTSTR



These two symbols are ambiguous.

FIGURE 6-6: WITH ILLFIRSTSTR



These two symbols can be distinguished

ILLFIRSTSTR is specified as a bit field. If any of the strokes in the bit field occur as the first stroke in the SYMSTRSEQ[] data, the sequence matching is aborted. See also [Table 6-60](#).

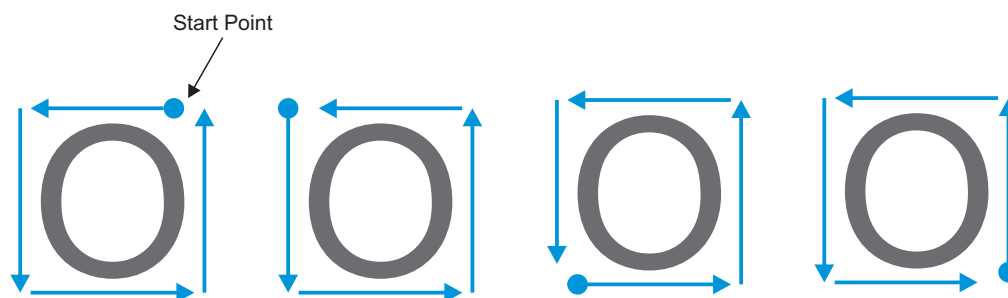
ILLFIRSTSTR Range: See [Table 6-57](#)

TABLE 6-57: ILLFIRSTSTR – BIT FIELD FORMAT

Bit 3	Bit 2	Bit 1	Bit 0
DOWN	UP	RIGHT	LEFT

ROLLSEQ: Specifies the symbol consists of a rolling stroke sequence. Some symbols can be drawn in varying rotations, which will mean that there are multiple valid start points for the symbol. For example, a five-point star or an “O” might be drawn starting at any point. In this case, setting this bit to 1 indicates that the symbol has a rolling stroke sequence. If the symbol has a fixed (unique) sequence, then this bit must be set to 0 (default).

FIGURE 6-7: ROLLSEQ = 1



This symbol can be drawn in any rotation

ROLLSEQ Range: 0 = Unique symbol orientation
1 = Multiple possible symbol orientations

SYMCFG3 Field

ASPECTRATIOX/ASPECTRATIOY: Constrain the Aspect Ratio of the symbol on X or Y axis.

Some symbols that are declared with strokes in only one axis can be expanded along the other axis by using this control. For example, a symbol "M" is declared as UP, DOWN, UP, DOWN.

ASPECTRATIOX/ASPECTRATIOYRange: See [Table 6-58](#) and [Table 6-59](#)

TABLE 6-58: ASPECTRATIOX

ASPECTRATIOX	Pass criteria: Axis to constrain \geq reference axis
0	No constraint
1	$X \geq 1/4 Y$
2	$X \geq 1/2 Y$
3	$X \geq 3/4 Y$
4	$X \geq Y$
5	$X \geq 5/4 Y$
6	$X \geq 6/4 Y$
7	$X \geq 7/4 Y$

Note: X = symbol length in the X-axis; Y = symbol length in the Y-axis

TABLE 6-59: ASPECTRATIOY

ASPECTRATIOY	Pass criteria: Axis to constrain \geq reference axis
0	No constraint
1	$Y \geq 1/4 X$
2	$Y \geq 1/2 X$
3	$Y \geq 3/4 X$
4	$YX \geq X$
5	$Y \geq 5/4 X$
6	$Y \geq 6/4 X$
7	$Y \geq 7/4 X$

Note: X = symbol length in the X-axis; Y = symbol length in the Y-axis

SYMRPTCFG Control

RPTCODE: Specifies the Reported Code for the symbol (for example, the ASCII code for the character detected). This code is sent in the Symbol Gesture Processor T115 message data (see [Section 5.13.4 “Messages”](#)).

RPTCODE Range: 0 to 127

DISRPT: Disable Reporting. If this bit is set to 1, reporting of this symbol is disabled. Note that if DISRPT is set to 1, this does not stop this symbol from being matched to the detected strokes. This feature can be used to suppress certain symbols by declaring them in SYMDATA but with DISRPT set to 1.

SYMSTRSEQ[] Fields

These fields specify the Symbol Stroke Sequence. SYMSTRSEQ contains encoded stroke data. The data is packed as two 4-bit stroke types per byte (see [Table 6-60](#) for the available stroke types).

Range: See [Table 6-60](#)

TABLE 6-60: SYMBOL STROKE TYPES

Value	Stroke Type
0	Null
2	Release
3	Left
4	Right
5	Up
6	Down

Note: There is no code for a simple “touch”, so a double tap must be coded as a Release–Release sequence (code 2).

Tips for configuring symbols:

- Put the simpler subset symbols *after* the more complex symbols in SYMDATA[]
The recognition algorithm searches through SYMDATA[] in sequence, and stops once a symbol match is found. Therefore a simple symbol will match first and prevent a more complex symbol from being matched.
For example, if a “C” is declared before an “O” in SYMDATA, an “O” drawn with [Left, Down, Right, Up] will be reported as a “C” drawn with [Left, Down, Right]. In this case it satisfies the “C” criteria and stops further symbol searches.
Example SYMDATA declaration sequences are:
 - O declared before C:**
O = [Left, Down, Right, Up, Rel]
C = [Left, Down, Right, Rel]
 - W declared before V:**
W = [Down, Up, Down, Up, Rel]
V = [Down, Up, Rel]
- Declare additional stroke combinations for the same symbol
It is helpful to include alternative combinations for the same symbol to allow differences in the way in which they may be drawn.
For example, an “S” can be declared with the following sequences:
[Left, Down, Right, Down, Left, Rel]
[Left, Down, Left, Rel]
This allows “S” is drawn quickly as a [Left, Down, Left] sequence.
Note that the same RPTCODE can be used for both symbols as there is no check for repeated RPTCODEs.
Similarly, the “Z” symbol can have two sequences defined:
[Right, Down, Right]
[Right, Left, Right]
so that a short “z” can be recognized.

6.17 Low Power Idle Configuration T126 Object

6.17.1 INTRODUCTION

The Low Power Idle Configuration T126 object configures the overall behavior of the low power features.

The low-power-idle features are an extension to the single-ended self capacitance measurement type, so the SELFPROX control must be set in Acquisition Configuration T8 MEASALLOW and MEASIDLEDEF (see “[MEASALLOW Field](#)” and “[MEASIDLEDEF Field](#)”).

This low power idle system runs two IIR filters on the measurement:

- A slow filter to achieve a form of reference value
- A fast filter to form a lightly-filtered signal value.

The difference between these filtered values is compared with a threshold value to determine whether or not the screen is touched.

High values for the slow filter coefficient will limit the ability of the system to cope with thermal changes.

When Low Power Idle Configuration T126 is enabled, the device can be set to use autonomous mode. In this mode, the parts of the device not involved in performing the low power idle measurements are completely powered off to save more power.

6.17.2 CONFIGURATION

TABLE 6-61: CONFIGURATION FOR LOW POWER IDLE CONFIGURATION T126 (SPT_SELFCAPIIDLECONFIG_T126)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	CTRL	DBGEN	Reserved	VMODEN	AUTOEN	RPTTCHEN	RPTAUTOEN	RPTEN	ENABLE
1	Reserved	Reserved							
2	GAIN	Measurement Gain							
3	THRESHOLD	Touch Threshold							
4	SYNCSPEL	Number of Sync Groups per Line							
5	DRIFTCOEF	Drift Filter Coefficient							
6	SIGFILTCOEF	Signal Filter Coefficient							
7	TUNPARAM	Tuning parameter LSByte							
8	Reserved	Reserved							

CTRL Field

ENABLE: Enables the use of this Low Power Idle Configuration T126 object. The object is enabled if set to 1, and disabled if set to 0.

RPTEN: Allows the object to send status messages to the host through the Message Processor T5 object. Reporting is enabled if set to 1, and disabled if set to 0.

RPTAUTOEN: Enables the reporting of messages to the host when the device is in autonomous mode. Reporting in autonomous mode is enabled if set to 1, and disabled if set to 0. Note that RPTEN must also be set for reporting to work.

RPTTCHEN: Enables message reporting when the device wakes up because of a touch being detected. Reporting is enabled if set to 1, and disabled if set to 0.

AUTOEN: Enables Autonomous Mode low power idle scan.

VMODEN: Enables voltage modulation on the integration level of low power idle scans.

DBGEN: Enables debug data output. Debug data is output can be viewed by the Diagnostic Debug T37 object (see [Section 6.17.4 “Diagnostic Debug Data”](#)) or by the hardware debug interface. Note that to view the data on the hardware debug interface, the Command Processor T6 DEBUGCTRL2 DBGOBJMODEEN bit must also be set to 1 (see “[DEBUGCTRL2 Field](#)”).

GAIN Field

This field sets the measurement gain to be used for the low power idle measurements.

Range: 0 to 255

THRESHOLD Field

This field sets the detect threshold (in delta values) to use for the low power idle measurements. A value of 255 disables thresholding.

Range: 0 to 254, 255 (disabled)

SYNCSPERL Field

This field specifies the number of ADCs per sensor line for the low power idle measurements (see [Section 6.14.1 "Introduction"](#) for more information).

The default value of 0 means use the value specified by Self Capacitance Configuration T111 Instance 1 IDLESYNCSPERL (that is, IDLESYNCSPERL for single-ended self capacitance measurements).

Range: 0 (use Self Capacitance Configuration T111 Instance 1 IDLESYNCSPERL), 1 to 255

DRIFTCOEF Field

This field sets the Drift Filter Coefficient. This is the coefficient for the filter that drifts the reference level of the low power idle measurements.

Recommended Range: 0 to 31

Typical: 4 to 9

SIGFILTCOEF Field

This field sets the Signal Filter Coefficient. The signal filter mitigates the effects of noise on the measurement to reduce the probability of false threshold breaches. The SIGFILTCOEF field sets the coefficient of the signal filter on the low power idle measurements.

Note that high SIGFILTCOEF values may lead to an unacceptable touch latency.

Recommended Range: 0 to 31

Typical: 0 to 1

TUNPARAM Field

This field sets the Tuning Parameters. Specifically, this field sets the value for the charge compensation to be used for the low power idle measurements. This is used for tuning the self capacitance mechanism.

Range: 0 to 255

Typical: 0 to 3

In addition, some fields may cause an automatic recalibration to be performed.

NOTE A configuration error will result if T126 is enabled and T113:CTRL:ALTAXISEN is configured to select the X axis.

6.17.3 MESSAGES

The message data for the Low Power Idle Configuration T126 object is shown in [Table 6-62](#).

TABLE 6-62: MESSAGE DATA FOR LOW POWER IDLE CONFIGURATION T126 (SPT_SELFCAPIDLECONFIG_T126)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	STATUS	Reserved						TYPE	
2	DELTA	Delta value LSByte							
3		Delta value MSByte							

STATUS Field

This field reports the current state of the Low Power Idle Configuration T126 object.

TYPE: Indicates the Message type (see [Table 6-63](#)).

TABLE 6-63: MESSAGE TYPE

Value	Meaning
0	Reserved for Command Processor T6 Report All
1	Positive threshold breached (touch detect)
2	Negative threshold breached (antitouch detect)
3	Autonomous mode started
4	Autonomous mode ended (other than touch or antitouch detect)
All other values	Reserved

6.17.4 DIAGNOSTIC DEBUG DATA

The diagnostic debug data modes for the Low Power Idle Configuration T126 object is shown in [Table 6-64](#). The diagnostic debug data can be viewed in the Diagnostic Debug T37 object by using the appropriate Command Processor T6 command, also listed in [Table 6-64](#).

The diagnostic debug modes are described in the following sections.

TABLE 6-64: DIAGNOSTIC DEBUG COMMANDS

T6 Command	Name	Description
0x3B	Low Power mode	The Diagnostic Debug T37 object holds debug data for the Low Power Mode. See Section 6.17.4.1 “Low Power Mode” .

6.17.4.1 Low Power Mode

[Figure](#) shows the organization of the data in the pages for the Low Power mode.

Diagnostic Debug T37 Data – Low Power Mode

Page	Data Index	Data[]
Page 0	0 – 1	Low power idle delta
	2 – 3	Low power idle reference level
	4 – 5	Raw low power idle signal
	6	Status: 0 = device in idle mode 1 = device in active mode
	7 – 8	Raw (unfiltered) low power signal
	9 – 127	Reserved

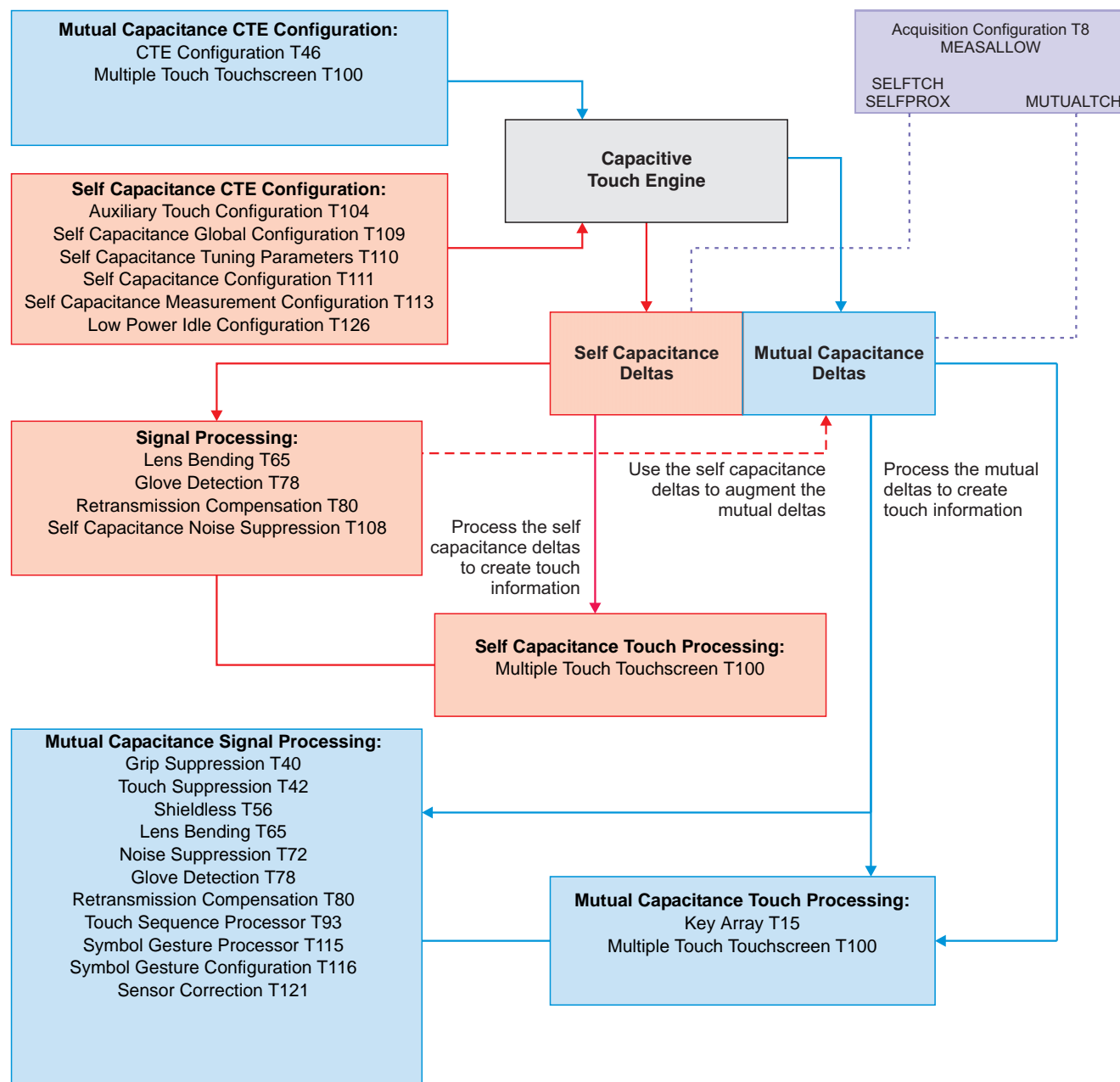
Assumes page size = 128

APPENDIX A: MEASUREMENT PROCESSING ON MXT144U

A.1 Measurement Processing Objects

Figure A-1 shows the different measurement types on the mXT144U and the objects used to configure and process these.

FIGURE A-1: MEASUREMENT PROCESSING



A.2 Touch Detection

The mXT144U allows for both mutual and self capacitance measurements, with the self capacitance measurements being used to augment the mutual capacitance measurements to produce reliable touch information.

When self capacitance measurements are enabled, touch classification is achieved using both mutual and self capacitance touch data. This has the advantage that both types of measurement systems can work together to detect touches under a wide variety of circumstances.

The system may be configured for different types of default measurements in idle and active modes. For example, the device may be configured for Mutual Capacitance Touch Default Idle and Self Capacitance Touch Default Active (or any other combination determined by Acquisition Configuration T8; see [Section 3.5.2 "Configuration"](#)). Note that other types of scans (such as other types of self capacitance scans) may also be made depending on configuration.

Mutual capacitance touch data is used wherever possible to classify touches as this has greater granularity than self capacitance measurements and provides positional information on touches. For this reason, multiple touches can only be determined by mutual capacitance touch data. In Self Capacitance Touch Default mode, if the self capacitance touch processing detects multiple touches, touchscreen processing is skipped until mutual capacitance touch data is available.

Self capacitance measurements and P2P mutual capacitance measurements allow for the detection of single touches in extreme cases, such as single thick glove touches, when mutual capacitance touch detection alone may miss touches.

A.3 Touch Classification

A.3.1 FINGER TOUCHES

Finger touches are classified when the following conditions are met:

- Mutual capacitance touch delta \geq Multiple Touch Touchscreen T100 TCHTHR,
or
mutual capacitance touch deltas $<$ Multiple Touch Touchscreen T100 TCHTHR
but has been compensated by Retransmission Compensation T80
and (if Self Capacitance Touch Default Active mode)
- Self capacitance touch deltas \geq Auxiliary Touch Configuration T104 XTCHTHR/YTCHTHR

A.3.2 GLOVE TOUCHES

Normal thin glove touches are classified using mutual capacitance data when the following conditions are met:

- Mutual capacitance touch deltas \geq Multiple Touch Touchscreen T100 INTTHR
and
mutual capacitance touch deltas $<$ Multiple Touch Touchscreen T100 TCHTHR (out of detect)
- and
- The conditions set in Glove Detection T78 are met for glove touches
and (if Self Capacitance Touch Default Active mode)
- Self capacitance touch deltas \geq Auxiliary Touch Configuration T104 XINTTHR/YINTTHR

Note that multiple touch glove touches are supported (mutual measurements only).

Thick glove touches are classified using self capacitance data when the following conditions are met:

- Mutual capacitance touch delta $<$ Multiple Touch Touchscreen T100 INTTHR
- and
- Self capacitance touch deltas \geq Glove Detection T78 SCTGLOVETHR
and meet Glove Detection T78 criteria

A.3.3 SUPPRESSED LARGE TOUCHES

Suppressed large touches are classified when the following conditions are met:

- Touch Suppression T42 has detected a large touch for suppression
and
- Touch Suppression T42 CFG SUPTCHRPTEN bit is set to 1 to allow the reporting of suppressed large touches

APPENDIX B: CHECKSUM CALCULATION

B.1 24-bit CRC

B.1.1 INTRODUCTION

The device uses a 24-bit cyclic redundancy check (CRC) in the following places:

- To check the integrity of the main Information Block for the device
- To check the integrity of the Information Block T254 object
- To check the integrity of the configuration settings held in the non-volatile memory

These CRC checksums allow the host to be confident that the memory map layout has been read over the communications bus correctly.

NOTE The C code in this appendix uses the type declarations `uint8_t`, `uint16_t` and `uint32_t` for 8-bit, 16-bit and 32-bit unsigned integers respectively.

B.1.2 24-BIT CRC ALGORITHM

Each checksum is generated by running an algorithm which takes two new bytes of data and combines them with the current checksum to produce a new checksum value.

The sample code below shows how to calculate the checksum for each 16-bit word in a byte stream.

```
uint32_t crc24(uint32_t crc, uint8_t firstbyte, uint8_t secondbyte)
{
    static const uint32_t crcpoly = 0x80001B;
    uint32_t result;
    uint16_t data_word;

    data_word = (uint16_t)firstbyte | (uint16_t)((uint16_t)secondbyte << 8u);
    result = (uint32_t)data_word ^ (uint32_t)(crc << 1u);

    if(result & 0x1000000) // If bit 25 is set
    {
        result ^= crcpoly; // XOR result with crcpoly
    }

    return result;
}
```

The checksum routine is called iteratively to calculate the CRC two bytes (16-bit word) at a time. This means that two bytes must be read from the device before performing each iteration of the checksum. Therefore, if an odd number of bytes is read from the device, the host's checksum code should add a zero byte to the end of the byte stream to make the sequence even. For example, if the following stream of 7 bytes is received from the device:

byte1 — byte2 — byte3 — byte4 — byte5 — byte6 — byte7

The checksum could be calculated as follows:

```
uint32_t CRC = 0;
CRC = crc24(CRC, byte1, byte2)
CRC = crc24(CRC, byte3, byte4)
CRC = crc24(CRC, byte5, byte6)
CRC = crc24(CRC, byte7, 0) /* <- zero added for the last checksum */

CRC = CRC & 0x0FFFFFFF; /* <- mask the 32-bit result to 24-bit */
```

An alternative method of calling the CRC calculation function is to use a loop, as shown in [Appendix B.1.3 "Information Block Checksum"](#).

[Table B-1](#) shows an example block of 32 bytes and the CRCs these generate. The bytes consist of 31 data bytes plus an additional zero byte to even up the count. You can use the data in this table to verify the validity of any coded CRC routine.

TABLE B-1: EXAMPLE CRC CALCULATIONS FOR 24-BIT CRC

Byte Pairs		CRC Calculation Result
First Byte	Second Byte	
0x00	0xFF	0x00FF00 (Intermediate CRC – partial calculation)
0x11	0xEE	0x011011 (Intermediate CRC – partial calculation)
0x22	0xDD	0x02FD00 (Intermediate CRC – partial calculation)
0x33	0xCC	0x053633 (Intermediate CRC – partial calculation)
0x44	0xBB	0x0AD722 (Intermediate CRC – partial calculation)
0x55	0xAA	0x150411 (Intermediate CRC – partial calculation)
0x66	0x99	0x2A9144 (Intermediate CRC – partial calculation)
0x77	0x88	0x55AAFF (Intermediate CRC – partial calculation)
0x88	0x77	0xAB2276 (Intermediate CRC – partial calculation)
0x99	0x66	0xD6226E (Intermediate CRC – partial calculation)
0xAA	0x55	0x2C116D (Intermediate CRC – partial calculation)
0xBB	0x44	0x586661 (Intermediate CRC – partial calculation)
0xCC	0x33	0xB0FF0E (Intermediate CRC – partial calculation)
0xDD	0x22	0xE1DCDA (Intermediate CRC – partial calculation)
0xEE	0x11	0x43A841 (Intermediate CRC – partial calculation)
0xFF	0x00	0x87507D (Expected CRC – final calculation)

B.1.3 INFORMATION BLOCK CHECKSUM

The checksum for the Information Block should be calculated by the host on start-up when the Information Block is first read. This should be compared to the checksum value at the end of the Information Block. If there is a mismatch, an error has occurred.

The following code shows how to use the example `crc24()` function given in [Appendix B.1.2 “24-bit CRC Algorithm”](#) to calculate the CRC checksum for the Information Block.

```
uint32_t crc = 0;                                /* Calculated CRC */
uint16_t crc_area_size;                          /* Size of data for CRC calculation */
uint8_t *mem;                                    /* Data buffer */
uint8_t i;
uint8_t status;

/* 7 bytes of version data, 6 * NUM_OF_OBJECTS bytes of object table. */
crc_area_size = ID_INFORMATION_SIZE +
    info_block->info_id.num_declared_objects *
    OBJECT_TABLE_ELEMENT_SIZE;

mem = (uint8_t *) malloc(crc_area_size);
if (mem == NULL)
{
    /* Handle error */
}

/*
 * Read the data using a function written for this purpose
 * Here, it is assumed that the function is named read_mem()
 * and that it returns a status code with the value READ_MEM_OK.
 * It is assumed to have the following prototype:
 * uint8_t read_mem( uint16_t Address, uint8_t ByteCount, uint8_t *Data )
 */
status = read_mem(0, crc_area_size, mem);

if (status != READ_MEM_OK)
```

```

{
    /* Handle error */
}

/*
 * Call the CRC function crc24() iteratively to calculate the CRC,
 * passing it two bytes at a time.
 */
i = 0;
while (i < (crc_area_size - 1))
{
    crc = crc24(crc, *(mem + i), *(mem + i + 1));
    i += 2;
}

/*
 * Call the crc24() with the final byte,
 * plus an extra 0 value byte to make the sequence even.
 */
crc = crc24(crc, *(mem + i), 0);

free(mem);

/* Final result */
crc = (crc & 0x00FFFFFF); /* <- mask the 32-bit result to 24-bit */

```

B.1.4 CONFIGURATION CHECKSUM

The configuration checksum checks the integrity of the memory map when the device non-volatile memory is written to. Typically this is when the device is initially set in the factory or during subsequent firmware upgrades. The host should perform a CRC on the entire contents of the memory map from the address of the Dynamic Configuration Container T71 object onwards. This CRC should then be compared with the expected checksum.

B.2 8-bit CRC

B.2.1 INTRODUCTION

An 8-bit CRC can be used in the following places:

- During communications with the host to check that data has been transmitted over the communications bus correctly (see [Section B.2.3 “Communications Checksum Mode”](#)).
- To check the integrity of data in messages from the Message Processor T5 object when receiving messages (see [Section B.2.4 “Reading the CRC for the Message Processor T5 Data”](#)).
- To check the integrity of data output in messages

B.2.2 8-BIT CRC ALGORITHM

The sample code below shows how to calculate the 8-bit checksum.

```

uint8_t crc8(unsigned char crc, unsigned char data)
{
    static const uint8_t crcpoly = 0x8C;
    uint8_t index;
    uint8_t fb;
    index = 8;

    do
    {
        fb = (crc ^ data) & 0x01;
        data >>= 1;
        crc >>= 1;
        if (fb)
            crc ^= crcpoly;
    } while (--index);

    return crc;
}

```

}

B.2.3 COMMUNICATIONS CHECKSUM MODE

B.2.3.1 Introduction

In communications checksum mode an 8-bit CRC is added to all data transmissions. The CRC is sent at the end of the data as the last byte before the STOP condition. The CRC is calculated on all the bytes sent, including the address bytes.

To indicate that a checksum is to be included, the most significant bit of the MSByte of the address must be set to 1. In the following examples, therefore, the start address of 0x1234 is sent as 0x9234.

The following sections give examples of I²C write and read operations.

B.2.3.2 I²C Write Example

This example sets the address pointer to 0x1234. The address is sent to the device with the most significant bit of the MSByte set to 1 (that is, 0x9234). This indicates I²C communications checksum mode. The example then writes five bytes to the device: four data bytes plus a checksum byte (see [Figure B-1](#)).

FIGURE B-1: EXAMPLE I²C WRITE WITH CHECKSUM

Address		Data		Data		Data		Data		CRC	
LSByte	MSByte										
START	SLA-W	0x34	0x92	0x96	0x9B	0xA0	0xA5	0x7A	STOP		

The example I²C command in [Figure B-1](#) writes the four bytes to contiguous addresses:

0x96 to address 0x1234

0x9B to address 0x1235

0xA0 to address 0x1236

0xA5 to address 0x1237

The I²C command sends a checksum (in this case, 0x7A) as the last byte before the STOP condition. The device compares this checksum with its own checksum calculated from the data sent. If the two checksums do not match, the Command Processor T6 object in the device generates a COMSERR message (see [Section 3.3.3 "Messages"](#)).

NOTE The order of messages is not guaranteed so messages may be out of order. Other intervening messages may be received before this one.

[Table B-2](#) shows the sequence of the bytes in this example and the intermediate calculations of the CRC.

TABLE B-2: CRC CALCULATIONS FOR EXAMPLE I²C WRITE

Bytes Sent by Host		CRC Calculations in the Device	
0x34	Address LSByte	0xDF	Intermediate CRCs (partial calculations)
0x92	Address MSByte	0xBB	
0x96	Data	0xDE	
0x9B		0x79	
0xA0		0xCB	
0xA5		0x7A*	Expected CRC (final calculation)
0x7A*	Actual CRC		

* These two values should match

B.2.3.3 I²C Read Example

A checksum can be added to I²C reads of the Message Processor T5 object. This contains a checksum as its last byte when I²C communications checksum mode is enabled in the Message Processor T5 object.

NOTE No other I²C reads can have a checksum, only reads of the Message Processor T5 object.

The example in [Figure B-2](#) reads the message data from the Message Processor T5 object in the device. It is assumed in this example that the address of the Message Processor T5 object is 0x1234. The address must be sent to the device with most significant bit of the MSByte set to 1 (that is, 0x9234). This indicates I²C communications checksum mode.

FIGURE B-2: EXAMPLE I²C READ WITH CHECKSUM

Set Address Pointer

Address					
		LSByte	MSByte	CRC	
START	SLA-W	0x34	0x92	0xBB	STOP

Read Data

		Report ID	Data	Data	Data	Data	Data	Data	Data	Data	CRC	
START	SLA-R	0x01	0x9B	0xA0	0xA5	0xAA	0xAF	0xB4	0xB9	0xCC	0x04	STOP

The example consists of two operations. The first operation is an I²C write of the address to the device. The second operation is an I²C read of the message data.

[Table B-3](#) shows the sequence of the initial write bytes with the intermediate calculations of the CRC.

TABLE B-3: CRC CALCULATIONS FOR EXAMPLE I²C WRITE OF ADDRESS

Bytes Sent by Host		CRC Calculations in the Device	
0x34	Address LSByte	0xDF	Intermediate CRC (partial calculation)
0x92	Address MSByte	0xBB*	Expected CRC (final calculation)
0xBB*	Actual CRC		

* These two values should match

An extra byte that holds the checksum (0xBB in this case) of the start address of the Message Processor T5 object is sent as the last byte before the STOP condition. This prevents a COMSERR message being generated by the Command Processor T6 object.

[Table B-4](#) gives the sequence of the read bytes with the intermediate calculations of the CRC.

TABLE B-4: CRC CALCULATIONS FOR EXAMPLE I²C READ OF MESSAGE DATA

Bytes Sent by Device		CRC Calculations in Host	
0x01	Report ID	0x5E	Intermediate CRCs (partial calculations)
0x9B	Data	0xF5	
0xA0		0xE4	
0xA5		0x18	
0xAA		0x8E	
0xAF		0x7D	
0xB4		0x56	
0xB9		0xA8	
0xCC		0x04*	
0x04*	Actual CRC		Expected CRC (final calculation)

* These two values should match

B.2.4 READING THE CRC FOR THE MESSAGE PROCESSOR T5 DATA

The following code shows how to use the example `crc8()` function given in [Section B.2.2 “8-bit CRC Algorithm”](#) to calculate the CRC checksum for the data in the Message Processor T5 object.

```
uint8_t crc = 0;
uint8_t data_in;

for(i=0; i<MESSAGEPROCESSOR_SIZE; i++)
{
    data_in = read_byte();
    crc = crc8(crc, data_in);
}

if(crc == 0)
{
    /* CRC is OK - do something appropriate */
    crc_pass();
}
else
{
    /* CRC failed - handle error */
    crc_fail();
}
```

APPENDIX C: BOOTLOADING PROCEDURE

The maXTouch devices have a common bootloading procedure that allows them to be forced into a special application update mode of operation.

The bootloader ID for the mXT144U is 0x88. The mXT144U bootloader uses extended ID mode.

The I²C address for bootloader mode on the mXT144U is 0x26.

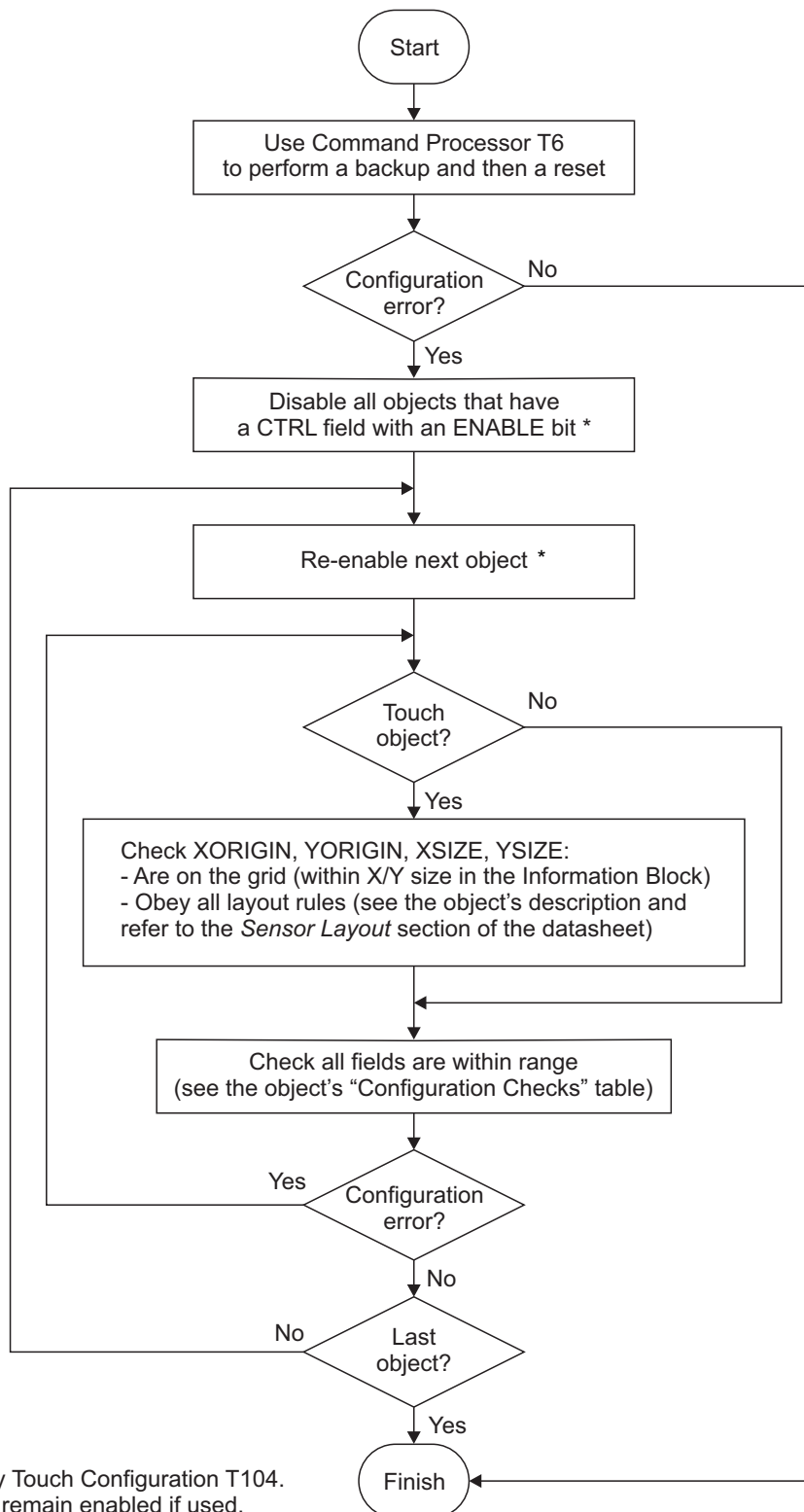
For more information on the bootloading procedure, refer to the following application note:

- Application Note: MXTAN0216 – *Bootloading Procedure for maXTouch Devices*

APPENDIX D: LOCATING CONFIGURATION ERRORS

Figure D-1 shows the algorithm for locating configuration errors. See [Section 1.7 "Configuration Checks"](#) for more information on the configuration checks performed by the device.

FIGURE D-1: ALGORITHM FOR LOCATING CONFIGURATION ERRORS



APPENDIX E: FUTURE INFORMATION BLOCK T254 OBJECT

E.1 Introduction

Currently the Object Table entries are held within the Information Block for the device. The current format, however, limits the object type codes to 8 bits. It is anticipated that at some time in the future new objects will be issued with 16-bit type codes (that is, codes above 255). The Information Block T254 object will therefore be added to the protocol to accommodate Object Table elements with 16-bit type codes. This means that if the Information Block T254 object is present on a device, it must be checked for any further Object Table entries.

NOTE It is important for future compatibility that any *current* driver code is written to allow for the presence of the Information Block T254 object when it is eventually used. The host driver code must parse the Object Table, as at present, and also process the Information Block T254 object if it is found to be present on the device.

This appendix gives the format of the future Information Block T254 object so that the driver code can be written accordingly.

E.2 Information Block T254

This object acts as an extension to the Information Block at the start of the device's memory map and holds additional Object Table entries. Specifically, it holds entries for those objects that have 16-bit type codes (that is, type codes above 255).

TABLE 6-65: CONFIGURATION FOR INFORMATION BLOCK T254 (GEN_INFOBLOCK16BIT_T254)

Byte	Field	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 – 6	Object Table	Object Table element 1 (7 bytes)							
7 – 13	Object Table	Object Table element 2 (7 bytes)							
...									
$m*7 - (m*7)+6$	Object Table	Last Object Table element (7 bytes)							
$(m+1)*7 - (m+1)*7+2$	Checksum	24-bit checksum (3 bytes)							

Object Table Element Fields

These 7-byte fields hold the additional elements in the Object Table. The format of each element is similar to that of the Object Table elements held directly in the main Object Table for the device (see [Section 1.5 “Object Table”](#)), except that the type code occupies two bytes instead of one. The format of each element is given in [Table E-1](#).

TABLE E-1: OBJECT TABLE ELEMENT FORMAT

Byte	Description
0	Type code LSByte
1	Type code MSByte
2	Start position LSByte
3	Start position MSByte
4	Size – 1
5	Instances – 1
6	Number of Report IDs per instance

Type code (2 bytes): This 16-bit field holds the unique type code for the object to identify it. This is the number after the “_T” suffix at the end of the object's internal name as given in the object descriptions. For example, the type code for the Command Processor T6 is 6 (from GEN_COMMANDPROCESSOR_T6).

Start position (2 bytes): This 16-bit field holds the start location of the object in the memory map.

Size – 1 (1 byte): This field holds the size (minus 1) of the object in the memory map. This is stored as Size-1, so it is effectively the offset to the end of the object.

Instances – 1 (1 byte): This holds the number of instances (minus 1) of the object in the memory map. The number of instances can be calculated by adding 1 to this number (see [Section 1.6.2 “Object Instances”](#)). The different instances of an object are arranged consecutively in the memory map.

Number of Report IDs per instance (1 byte): This field holds the number of report IDs for each instance of the object. See [Section 1.5.6 “Report IDs”](#) for more information on report IDs.

Checksum Field

A checksum of the contents of the Information Block T254 object. This allows the host to check that the Object Table elements stored in the Information Block T254 object have been read correctly over the communications interface. See [Appendix B “Checksum Calculation”](#) for details on calculating the checksum.

APPENDIX F: ASSOCIATED DOCUMENTS

NOTE Some of the documents listed below are available under NDA only.
--

The following documents are available by contacting your Microchip representative:

Product Documentation

- *mXT144U 1.0 Data Sheet*
- *maXTouch U Series Tuning Guide* (distributed with Microchip approval only)
- Application Note: MXTAN0213 – *Interfacing with maXTouch Touchscreen Controllers*

Touchscreen design and PCB/FPCB layout guidelines

- Application Note: QTAN0054 – *Getting Started with maXTouch Touchscreen Designs*
- Application Note: MXTAN0208 – *Design Guide for PCB Layouts for maXTouch Touch Controllers*
- Application Note: QTAN0080 – *Touchscreens Sensor Design Guide*

Configuring the device

- Application Note: QTAN0059 – *Using the maXTouch Self Test Feature*

Miscellaneous

- Application Note: QTAN0050 – *Using the maXTouch Debug Port*
- Application Note: QTAN0058 – *Rejecting Unintentional Touches with the maXTouch Touchscreen Controllers*
- Application Note: QTAN0061 – *maXTouch Sensitivity Effects for Mobile Devices*
- Application Note: MXTAN0216 – *Bootloading Procedure for maXTouch Devices*

Tools

- *maXTouch Studio User Guide* (distributed as on-line help with maXTouch Studio)

Device drivers

- Application note: QTAN0055 – *Writing Object-based Host code for maXTouch Devices*
- Application note: MXT0201 – *Object Server Configuration File Format*

APPENDIX G: REVISION HISTORY

Revision CX (August 2016)

Final Atmel edition for firmware revision 1.0.AA – Atmel Release version

Revision A (November 2017)

Reformatted edition for firmware revision 1.0.AB – Microchip Release version

This revision incorporates the following updates:

- Updated to Microchip protocol guide format:
 - “[To Our Valued Customers](#)” added
 - Associated Documents moved to [Appendix F “Associated Documents”](#)
 - Known Issues removed. Now in own document
 - Revision History moved to this appendix
 - Index added
 - “[The Microchip Web Site](#)”, “[Customer Change Notification Service](#)” and “[Customer Support](#)” sections added
 - Back cover updated
 - References to Atmel Corporation removed or changed to Microchip Technology Inc, where appropriate
 - New documentation number assigned
- [Section 1.5.6 “Report IDs”](#) added (missing in error)
- Message Processor T5 object documentation added (was marked Internal in error)
- Power Configuration T7:
 - FINEIDLEACQINT renamed to IDLEACQINTFINE
 - FINEACTVACQINT renamed to ACTVACQINTFINE
 - Figures updated to include IDLEACQINTFINE and ACTVACQINTFINE
- Acquisition Configuration T8:
 - MEASIDLEDEF: [Table 3-10](#) updated with additional comments
- Key Array T15:
 - XORIGIN, YORIGIN, XSIZE and YSIZE Fields: Note concerning sharing of lines with other touch objects updated. Ranges added for all four fields
- Communications Configuration T18:
 - CTRL HISPEEDSPI added (missing in error)
- Self Test T25:
 - Pin Fault Test results: [Table 6-10](#) updated to mention driven shield. X_PIN and Y_PIN updated to mention driven shield failure
- Diagnostic Debug T37:
 - Product Data Store Mode description moved here from Serial Data Command T68
- Touch Suppression T42:
 - SUPSTRENGTH: Range corrected to show 255 disables Edge Touch Suppression and Large Object Touch Suppression
 - Configuration checks: APPRTHR corrected to “No” configuration check
- CTE Configuration T46:
 - *Pulse Synchronization* section removed (included in error)
 - SYNCDELAY and ADCSPERSYNC fields removed (included in error)
- Shieldless T56:
 - Configuration checks updated to show correct footnote for OPTINT
 - STATUS message field: Note added for UNCAL bit
- Lens Bending T65:
 - FORCEDI: Note added. Range values corrected: 0 = 1 (default), values 1 and 2 are not capped to 2
 - ATCHRATIO: ATCHRATIO and DSRATIO updated to state capping behavior

- Serial Data Command T68:
 - Product Data Store Mode description moved to Diagnostic Debug T37
- Dynamic Configuration Controller T70:
 - Event Enumeration IDs 132 to 134 added (missing in error).
- Noise Suppression T72:
 - Object renamed (was maXCharger T72)
 - [Figure 5-15](#) updated to show MINTHRFRAC in formula instead of NLTHRFRAC
- Multiple Touch Touchscreen T100:
 - TCHAUX: Example usage updated for clarity
 - NUMTCH: Text concerning Touch ID allocation added
 - XORIGIN, YORIGIN, XSIZE and YSIZE Fields: Note concerning sharing of lines with other touch objects updated. Ranges updated for all four fields
 - MOVHYSTI and MOVHYSTN: Additional text concerning implementing a jitter filter added
- Auxiliary Touch Configuration T104:
 - XGAIN and YGAIN maximum corrected to 29 (was 44 in error)
- Self Capacitance Noise Suppression T108:
 - Object renamed (was maXCharger T108)
 - [Table 5-41](#) table modified (Configuration Check and Automatic Recalibration columns were wrong way round)
- Self Capacitance Configuration T111:
 - *Pulse Synchronization* section removed (included in error)
 - DBGCTRL SIGSEN bit now documented (was reserved)
 - ALTDLAYTIMEX range corrected to 0 to 255
 - SYNCDELAY and ADCSPERSYNC field descriptions removed (included in error)
 - DCPROCTHRX and DCPROXTHRY fields added (missing in error)
- Self Capacitance Measurement Configuration T113:
 - XGAIN and YGAIN maximum corrected to 29 (was 44 in error)
- Symbol Gesture Processor T115:
 - LSTRNUMTCHX/LSTRNUMTCHY range corrected to 0 to 7
- Low Power Idle Configuration T126:
 - DRIFTCOEF and SIGFILTCOEF: documented ranges changed to a recommended range of 0 to 31, as this is the effective range, and typical values updated
- [Appendix A “Measurement Processing on mXT144U”](#):
 - [Appendix A.2 “Touch Detection”](#): Description updated
 - [Appendix A.3 “Touch Classification”](#): Classification criteria updated
- Other minor modifications to aid clarity

INDEX

A

Acquisition Configuration T8 object.....	29
calibration recovery process	29
calibration recovery tuning data mode	38
configuration checks	37
configuration controls.....	32, 38
ATCHCALST field	34
ATCHCALSTHR field.....	34
ATCHFRCCALRATIO field	34
ATCHFRCCALTHR field.....	34
CFG field.....	36
CHRGTIME field	32
DRIFTST field	33
MEASACTVDEF field	36
MEASALLOW field.....	35
MEASIDLEDEF field	35
REFMODE field	36
TCHAUTOCAL field	33
TCHDRIFT field	32
diagnostic debug data	38
mutual capacitance recovery process.....	30
self capacitance measurements	
calibration and touch recovery process	31, 201
Active to idle timeout	25
ADCs, number of	
CTE Configuration T46 object.....	173, 174
Noise Suppression T72 object	114
Self Capacitance Configuration T111 object.....	197, 200
Antitouch calibration.....	37
Autonomous mode	212
Auxiliary Touch Configuration T104 object	190
configuration checks	192
configuration controls.....	190
CTRL field	190
XGAIN field	190
XINTHYST field.....	191
XINTTHR field.....	191
XTCHHYST field	190
XTCHTHR field	190
YGAIN field	191
YINTHYST field.....	192
YINTTHR field.....	192
YTCHHYST field	191
YTCHTHR field	191

B

Background scanning and touch hold-off qualification	
Noise Suppression T72 object	113
Self Capacitance Noise Suppression T108 object	145
Byte order.....	11

C

Calibration recovery process.....	29
Calibration recovery tuning data mode	38
CHG line.....	20
host retrigging mode	164
mode selection	164
Classes of objects	8
Command Processor T6 object.....	22
configuration controls.....	22
BACKUPNV field.....	22
CALIBRATE field	22
DEBUGCTRL field	23
DEBUGCTRL2 field	23

DIAGNOSTIC field.....	23
REPORTALL field.....	22
RESET field	22
message data	24
CHECKSUM field	24
STATUS field	24
Communications Configuration T18 object.....	164
configuration controls	164
COMMAND field	164
CTRL field.....	164
Configuration and tuning.....	11
Configuration checks	11
Configuration values	10
CTE Configuration T46 object	173
ADCs per X.....	174
configuration checks	174
configuration controls	173
ACTVSYNCSPERX field	173, 174
CTRL field.....	173
IDLESYNCSERX field.....	173, 174
INRUSHCFG field.....	174
MODE field	174
PULSESERADC field.....	173, 174
message data	175
STATUS field	175
Custom data	171
Customer Change Notification Service.....	238
Customer Notification Service	238
Customer Support.....	238

D

Delta values	11
Diagnostic debug data	38
Diagnostic Debug T37 object.....	14
configuration controls	18
DATA[] field	18
MODE field	18
PAGE field.....	18
DC data mode	17
delta mode (mutual capacitance)	15
delta mode (self capacitance).....	16
device information mode	16
product data store mode.....	17
reference mode (mutual capacitance)	15
reference mode (self capacitance)	16
system diagnostic debug modes	14
Distance touch suppression.....	97
Double tap sequence gesture.....	138
Drift compensation.....	32
Dual X drive	114
Dynamic Configuration Container T71 object	189
configuration controls	189
DATA[] field	189
Dynamic Configuration Controller T70 object	181
configuration checks.....	187
configuration controls	183
CTRL field.....	183
DSTOFFSET field.....	187
EVENT field	184
LENGTH field	187
OBJINST field.....	187
OBJTYPE field.....	186
SRCOFFSET field	187
event triggers.....	181

MXT144U 1.0

message data.....	188
STATUS field	188

E

Edge correction	
Multiple Touch Touchscreen T100 object	46
Edge touch suppression.....	97
Evaluation grip suppression mode	93
Event triggers	181
calibration.....	184
Dynamic Configuration Controller T70 object	181
Lens bending force.....	186
moisture processing	185
noise suppression	184
Noise Suppression T72 object	115
touch detection.....	184
touch suppression	184

F

Fast detect integration (Fast DI).....	26
Finger touches	216
Force on touchscreen, correcting for	104
Frequency hopping	
Noise Suppression T72 object	111
Self Capacitance Noise Suppression T108 object	145

G

Gestures	
double tap sequence.....	138
one-touch gestures	80
two-touch gestures.....	89
Glove Detection T78 object.....	128
configuration checks	130
configuration controls.....	128
CONFTHR field.....	129
CTRL field	128
DISCRIMTHR field.....	130
GLOVEMODETO field	129
MINAREA field	129
MINDIST field.....	129
SCTGLOVEHYST field	130
SCTGLOVETHR field	130
SCTMINAREA field.....	130
SUPTO field	129
SYNCSPEX field	130
glove confidence mode	128
glove normal mode.....	128
message data.....	131
Glove touches	128, 216
Grip suppression	
mutual capacitance measurements	92–96
Grip suppression boundary	92
Grip suppression mode	93
Grip Suppression T40 object.....	92
configuration controls.....	94
CTRL field	95
NUMTCHBEFSUP field	95
XHIGRIP field.....	95
XLOGRIP field	95
YHIGRIP field.....	95
YLOGRIP field	95
evaluation mode.....	93
grip mode	93
grip suppression boundary.....	92
message data.....	96
number of touches	92

H

High speed SPI.....	164
---------------------	-----

I

I ² C communications	
CHG line	20
ID information	6
Information block	5
Information Block T254 object	226
configuration controls	
Checksum field	227
Object Table Element Fields	226
Internet Address	238

K

Key Array T15 object	41
configuration checks	44
configuration controls	41
AKSCFG field	42
BLEN field.....	43
CFG field	44
CTRL field.....	41
TCHDI field	43
TCHHYST field	43
TCHTHR field	43
XORIGIN field.....	42
XSIZE field.....	42
YORIGIN field.....	42
YSIZE field.....	42
debug data.....	45
key deltas mode	45
key references mode.....	45
message data	44
KEYSTATE field	45
STATUS field	45
touch threshold	41

L

Large touch suppression	97, 217
Lens bending correction	104
Lens Bending T65 object	104
configuration checks	108
configuration controls	104
ATCHRATIO field	106
CTRL field.....	104
EXFRCTHR field	107
EXFRCTHRHYST field.....	107
EXFRCTO field.....	108
FORCEDI field.....	106
FORCEHYST field.....	106
FORCESCALE field.....	105
FORCETHR field	105
FORCETHRHYST field	106
GRADTHR field	105
LPFILTCOE field	105
YHINOISEDIV field.....	105
YHINOISEMUL field	105
YLONOISEDIV field.....	105
YLONOISEMUL field	105
event triggers.....	186
message data	109
FORCE field	109
STATUS field.....	109
noise suppression for shieldless sensor designs	104
Low Power Idle Configuration T126 object	212
configuration controls	212

CTRL field	212
DRIFTCOEF field.....	213
GAIN field.....	212
SIGFILTCOEF field.....	213
SYNCSPEL field.....	213
THRESHOLD field	213
TUNPARAM field	213
debug data	214
low power mode.....	214
message data.....	213
STATUS field	213

M

Maximum screen area suppression	97
Maximum touch suppression	97
Measurement processing	215
finger touches.....	216
glove touches.....	128, 216
measurement processing objects	215
objects used.....	215
suppressed large touch.....	217
touch classification	216
touch detection.....	216
Measurement types.....	215
selecting	35
Memory map structure	5
Message Count T44 object	172
configuration controls	172
COUNT field.....	172
Message Processor T5 object.....	20
configuration controls	20
CHECKSUM field.....	21
MESSAGE field.....	20
REPORTID field.....	20
Microchip Internet Web Site	238
Minimum touch threshold	112
Moisture compensation	132
Multiple Touch Touchscreen T100 object	46
configuration checks	72
configuration controls	48
AKSCFG field.....	56
AMPLCOEFF field	70
AMPLHYST field	68
AMPLOFFSET field	71
CFG1 field.....	49
CFG2 field.....	69
CTRL field	49
CUTOFFTHR field	63
DXGAIN field.....	63
DXTHRSF field	64
DXXEDGECFG field	61
DXXEDGECFGHI field.....	68
DXXEDGEDIST field.....	62
DXXEDGEDISTHI field	69
GAIN field.....	62
INTTHR field	63
INTTHRHYST field.....	68
JLMMOVINTTHR field	72
JLMMOVTHR field	71
JUMPLIMIT field	65
JUMPLIMITMOV field	71
MOVFILTER field.....	65
MOVHYSTCFG field	69
MOVHYSTI field.....	67
MOVHYSTN field.....	67
MOVPRD field	66

MOVSMOOTH field	66
MRGHYST field.....	64
MRGTHR field	64
MRGTHRADJSTR field	64
NEXTTCHDI field	65
NOISESF field	63
NUMTCH field	56
SCRAREAHYST field	68
SCRAUX field	50
TCHAUX field	52
TCHDIDOWN field.....	64
TCHDIUP field.....	65
TCHEVENTCFG field	56
TCHHYST field	63
TCHTHR field	63
XEDGECFG field.....	61
XEDGECFGHI field	68
XEDGEDIST field	62
XEDGEDISTHI field.....	69
XHICLIP field	58
XLOCLIP field.....	58
XORIGIN field.....	57
XPITCH field.....	58
XRANGE field.....	58
XSIZE field.....	57
XYCFG field.....	57
YEDGECFG field.....	61
YEDGECFGHI field	68
YEDGEDIST field	62
YEDGEDISTHI field.....	69
YHICLIP field	58
YLOCLIP field.....	58
YORIGIN field.....	57
YPITCH field.....	58
YRANGE field.....	58
YSIZE field.....	57
debug data.....	77
touch status data mode	78
touchscreen mode	78
edge correction	46
event triggers	184
message data	74
AUXDATA[] field	75, 77
SCRSTATUS field	75
TCHSTATUS field	76
XPOS field.....	77
YPOS field.....	77
SUP message.....	96
touch threshold	46
UNSUP message	96
Mutual capacitance measurements	
grip suppression	92–96
noise suppression.....	110–127
selecting	35
setting as default	35, 36
touch recovery process	30
mutual capacitance measurements	
gain, setting	43, 62
thresholds, setting	41, 46

N

Noise lines threshold	113
Noise measurement.....	110
Noise notification mechanisms	
Noise Suppression T72 object.....	110
Self Capacitance Noise Suppression T108 object	144

Noise suppression		
event triggers	184	
mutual capacitance	110–127	
self capacitance	144–154	
shieldless sensor designs	104	
Noise Suppression T72 object	110	
ADCs per X	114	
background scanning and touch hold-off qualification	113	
configuration checks	124	
configuration controls	115	
BGNLRATIONOIS field	124	
BGNLRATIOSTAB field	124	
BGNLRATIOVNOI field	124	
BGSCAN field	120	
BLKNLTHR field	120	
CALCFG1 field	118	
CFG1 field	118	
CFG2 field	118	
CFG3 field	121	
CTRL field	117	
FALLNLTHR field	119	
field	123	
HOPCNT field	118	
HOPCNTPER field	118	
HOPEVALTO field	119	
HOPST field	119	
INCNLTHR field	119	
MINNLTHR field	119	
MINTHRADJ field	120	
NLGAINDUALX field	119	
NLGAINSINGX field	120	
NLRATIO field	124	
NLTHRLIMIT field	120	
NLTHRMARGIN field	119	
NOISCNT field	122	
NOISCTRL field	121	
NOISFREQ[] field	122	
NOISHIGHNLTHR field	122	
NOISLOWNLTHR field	122	
NOISNOTCHAPX[] field	122	
NOISTCHAPX[] field	122	
NOTCHMINDIFF field	123	
NOTCHMINHOP field	123	
STABCTRL field	121	
STABFREQ[] field	122	
STABHIGHNLTHR field	122	
STABNOTCHAPX[] field	122	
STABTCHAPX[] field	122	
TCHMINDIFF field	123	
TCHMINHOP field	124	
VNOICNT field	122	
VNOICTRL field	121	
VNOIFREQ[] field	122	
VNOILOWNLTHR field	122	
VNOINOTCHAPX[] field	122	
VNOITCHAPX[] field	122	
dual X drive	114	
event triggers	115, 184	
frequency hopping	111	
general initial settings	115	
message data	126	
MINTCHTHR field	127	
NLTHR field	127	
NOISELVL field	127	
PKNOISELVL field	127	
STATUS1 field	126	
STATUS2 field	127	
minimum touch threshold	112	
noise lines threshold	113	
noise measurement	110	
noise notification mechanisms	110	
resetting the noise suppression algorithm	114	
transitions between noise states	110	
Number of instances	7	
Number of touches		
grip suppression	92	
Multiple Touch Touchscreen T100 object	56	
O		
Object Protocol	5	
byte order	11	
classes of objects	8	
compatibility of object versions	10	
configuration checks	11	
configuration values	10	
delta values	11	
ID information	6	
information block	5	
memory map structure	5	
number of instances	7	
object instances	9	
object table	6	
object types	7	
objects	8	
report IDs	7	
Object table	6	
Object types	7	
Objects	8	
Acquisition Configuration T8	29	
Auxiliary Touch Configuration T104	190	
Command Processor T6	22	
Communications Configuration T18	164	
compatibility of object versions	10	
configuration and tuning	11	
configuration checks	11	
configuration values	10	
CTE Configuration T46	173	
Diagnostic Debug T37	14	
Dynamic Configuration Container T71	189	
Dynamic Configuration Controller T70	181	
Glove Detection T78	128	
Grip Suppression T40	92	
Information Block T254 object	226	
instances	9	
Key Array T15	41	
Lens Bending T65	104	
Low Power Idle Configuration T126	212	
Message Count T44	172	
Message Processor T5	20	
Multiple Touch Touchscreen T100	46	
Noise Suppression T72	110	
One-touch Gesture Processor T24	80	
Power Configuration T7	25	
Retransmission Compensation T80	132	
Self Capacitance Configuration T111	197	
Self Capacitance Global Configuration T109	193	
Self Capacitance Measurement Configuration T113	206	
Self Capacitance Noise Suppression T108	144	
Self Capacitance Tuning Parameters T110	196	
Self Test T25	165	
Sensor Correction T121	161	
Serial Data Command T68	178	

Shieldless T56.....	102
Symbol Gesture Configuration T116.....	208
Symbol Gesture Processor T115.....	155
Timer T61.....	176
Touch Sequence Processor T93.....	138
Touch Suppression T42.....	97
Two-touch Gesture Processor T27.....	89
User Data T38.....	171
One-touch Gesture Processor T24 object.....	80
configuration controls.....	82
CTRL field.....	82
DRAGTHR field.....	86
DRAGTO field.....	84
FLICKTHR field.....	86
FLICKTO field.....	84
GESTEN field.....	83
LPRESSTO field.....	85
NUMGEST field.....	83
PROCESS field.....	83
REPPRESSTO field.....	85
SPRESSTO field.....	84
TAPTHR field.....	86
TAPTO field.....	84
THROWTHR field.....	86
message data.....	86
DIR field.....	87
DIST field.....	88
STATUS field.....	87
XPOSMSB field.....	87
XYPOSLSB field.....	87
XYPOSMSB field.....	87
YPOSMSB field.....	87
One-touch gestures.....	80

P

PDS data type.....	178
Power Configuration T7 object.....	25
active to idle timeout.....	25
configuration controls.....	27
ACTV2IDLETO field.....	27
ACTVACQINT field.....	27
ACTVACQINTFINE field.....	28
CFG field.....	28
CFG2 field.....	28
IDLEACQINT field.....	27
IDLEACQINTFINE field.....	28
power modes.....	25, 27
Pulses per ADC.....	173

R

Report IDs.....	7
Retransmission compensation.....	132
Retransmission Compensation T80 object.....	132
configuration checks.....	136
configuration controls.....	132
ATCHTHR field.....	133
COMPCFG field.....	135
COMPGAIN field.....	133
COMPSTRTHR field.....	135
COMPTHR field.....	133
CTRL field.....	132
MOISTCFG field.....	134
MOISTCFG2 field.....	134
MOISTCFG3 field.....	136
MOISTINVATCHTHR field.....	134
MOISTTHR field.....	134

MOISTVLDTHRSF field.....	135
TARGETDELTA field.....	133
event triggers.....	185
message data.....	136
STATUS field.....	137

S

Self Capacitance Configuration T111 object.....	197
configuration checks.....	204
configuration controls.....	198
ACTVSYNCSPERL field.....	200
ALTDELAYTIMEX field.....	202
ALTINTTIMEX field.....	201
CALRECSTR field.....	201
CTRL field.....	199
DBGCTRL field.....	200
DCCALERRRATIO field.....	203
DCCALRECSTR field.....	203
DCDRIFT field.....	202
DCFILTER field.....	202
DCGAINSF field.....	203
DCIDLESLEWMIN field.....	204
DCPROCTHRX field.....	204
DCPROCTHRY field.....	204
DCTHRX field.....	203
DCTHRY field.....	203
DELAYTIME field.....	200
DRIFT field.....	200
DRIFTST field.....	200
IDLESYNCSPERL field.....	200
INRUSHCFG field.....	201
INTTIME field.....	200
instances.....	197
self capacitance measurements.....	198
self capacitance single-ended measurements.....	198
Self Capacitance Global Configuration T109 object.....	193
configuration checks.....	195
configuration controls.....	193
CMD field.....	193
CMDONRESET field.....	193
CTRL field.....	193
TUNECFG field.....	194
message data.....	195
CMD field.....	195
ERRORCODE field.....	195
Self Capacitance Measurement Configuration T113 object.....	206
configuration checks.....	206
configuration controls.....	206
CTRL field.....	206
GAINX field.....	206
GAINY field.....	206
Self capacitance measurements.....	198
configuration.....	198
gain, setting.....	190
global configuration.....	193
glove touches.....	128
noise suppression.....	144–154
selecting.....	35
setting as default.....	35, 36
thresholds, setting.....	190
self capacitance measurements.....	206
gain, setting of.....	206
Self Capacitance Noise Suppression T108 object.....	144
background scanning and touch hold-off qualification.....	145
configuration checks.....	152
configuration controls.....	147

BLKNLTDTHR field.....	152	TYPE_INSTANCE field	170
CALCFG1 field.....	149	TYPE_NUM field	170
CFG1 field.....	149	X_PIN field.....	169
CFG2 field.....	149	Y_PIN field.....	169
CFG3 field.....	149	Self-test routines.....	165
CTRL field.....	148	Sensor Correction T121 object.....	161
HOPST field.....	150	configuration controls	161
IIRCOEFF field.....	150	CFG1 field.....	161
MINNLTDIFF field	150	CTRL field.....	161
MINNLTDHOP field.....	150	YEDGESF field.....	162
NLGAIN field	150	Serial Data Command T68 object.....	178
NOISCNT field	152	configuration controls	178
NOISCTRL field.....	150	CMD field.....	179
NOISFREQ[] field.....	151	CTRL field.....	178
NOISHINLTDTHR field	151	DATA field	178
NOISLONLTDTHR field	151	DATATYPE field.....	178
NOISNOTCHAPX[] field.....	151	LENGTH field	178
NOISTCHAPX[] field	151	debug data.....	180
STABCTRL field.....	150	product data store mode.....	17, 180
STABFREQ[] field	151	message data	179
STABHINLTDTHR field.....	151	CHECKSUM field	180
STABNOTCHAPX[] field.....	151	STATUS field.....	179
STABTCHAPX[] field	151	PDS data type	178
VNOICNT field	152	Shieldless T56 object.....	102
VNOICTRL field.....	150	configuration checks	103
VNOIFREQ[] field.....	151	configuration controls	102
VNOILONLTDTHR field	151	CTRL field.....	102
VNOINOTCHAPX[] field.....	151	INTDELAY[] field	102
VNOITCHAPX[] field	151	INTTIME field.....	102
frequency hopping.....	145	OPTINT field.....	102
message data.....	153	message data	103
PKNLTDDIFF field	154	STATUS field.....	103
PKNLTDHOP field	154	Software tools.....	12
PKNLTDLVL field.....	154	SUP message.....	96
STATUS1 field	153	Symbol Gesture Configuration T116 object.....	208
STATUS2 field	154	configuration controls	208
SUSHOPS field	154	Control field	211
noise notification mechanisms	144	SYMCFG1 field.....	208
suggested initial settings.....	146	SYMCFG2 field.....	209
transitions between noise states	144	SYMCFG3 field.....	210
Self capacitance single-ended measurements		SYMDATA[] field.....	208
configuration.....	198	SYMRPTCFG field	211
gain, setting of.....	206	SYMSTRSEQ[] field	211
selecting.....	35	Symbol Gesture Processor T115 object.....	155
setting as default.....	35	configuration checks	158
Self capacitance single-ended. See also <i>Self capacitance measurements</i>		configuration controls	155
Self Capacitance Tuning Parameters T110 object.....	196	CTRL field.....	155
configuration controls	196	EXTSTRCNTMAX field.....	156
PARAMS[] field	196	LSTRANGLE field.....	157
Self Test T25 object	165	LSTRCFG1 field	157
configuration checks	167	LSTRCNTTHR field	157
configuration controls	165	LSTRNUMTCH field	157
CMD field	166	MOVX field	156
CTRL field	166	MOVY field	156
LOSIGLIM[] field	166	SYMTIMEMAX field.....	156
PINDWELLUS field	167	SYMTO field	156
PINTHR field	167	XMAX field.....	156
SIGRANGELIM[] field	167	XMIN field.....	156
UPSIGLIM[] field	166	YMAX field.....	156
debug data	170	YMIN field.....	156
self test mode.....	170	debug data.....	160
message data.....	168	symbol gesture mode	160
INFO field.....	168	message data	159
SEQ_NUM field.....	169	DATA0 field	160
STATUS field	168	DATA1 field	160
		RPTSYM field	159

T

TCHDI fields	26
Timer T61 object	176
configuration controls	176
CMD field	176
CTRL field	176
MODE field	176
PERIOD field	176
message data	177
CNTVAL field	177
STATUS field	177
Timers	176
Touch classification	216
Touch detection	216
Touch Sequence Processor T93 object	138
configuration checks	141
configuration controls	139
CTRL field	140
DTNOTCHMOVMAX field	140
DTNOTCHTIMMAX field	141
DTNOTCHTIMMIN field	141
DTPOSTTIMMIN field	140
DTPRETIMMIN field	140
DTTCHMOVMAX field	140
DTTCHTIMMAX field	141
DTTCHTIMMIN field	140
FAILRPEN field	141
XMAX field	140
XMIN field	140
YMAX field	140
YMIN field	140
double tap sequence	138
message data	142
FAILREASON field	142
STATUS field	142
touch movement	138
touch position	138
Touch suppression	
event triggers	184
Touch Suppression T42 object	97
configuration checks	100
configuration controls	97
CFG field	100
CTRL field	97
DITHYST field	100
EDGSUPSTR field	100
MAXAPPRAREA field	98
MAXNUMTCHS field	99
MAXSCRNAREA field	100
MAXTCHAREA field	98
SHAPESTRENGTH field	99
SUPDIST field	99
SUPEXTTO field	99
SUPSTRENGTH field	98
distance touch suppression	97
edge touch suppression	97
event triggers	184
large touch suppression	97
maximum screen area suppression	97
maximum touch suppression	97
message data	101
Touch threshold	
Key Array T15 object	41
Multiple Touch Touchscreen T100 object	46
Transitions between noise states	

Noise Suppression T72 object	110
Self Capacitance Noise Suppression T108 object	144
Two-touch Gesture Processor T27 object	89
configuration controls	89
CTRL field	89
GESTEN field	90
NUMGEST field	90
ROTATETHR field	90
ZOOMTHR field	90
message data	90
ANGLE field	91
SEP field	91
STATUS field	90
XPOSMSB field	91
XYPOSLSB field	91
XYPOSMSB field	91
YPOSMSB field	91
Two-touch gestures	89

U

UNSUP message	96
User Data T38 object	171
configuration controls	171
DATA[] field	171

W

WWW Address	238
-------------------	-----

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: <http://microchip.com/support>

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

**QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
= ISO/TS 16949 =**

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KlearNet, KlearNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2017, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-2398-0

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
[http://www.microchip.com/
support](http://www.microchip.com/support)
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820