Atmel QTouch

Introduction to QTouch Design Parameters using SAM D21 Xplained Pro

TRAINING MANUALS

Introduction

This hands-on demonstrates the ease of using various Atmel® QTouch® technology tools used to develop and tune capacitive buttons, sliders and wheels designs. It utilizes Atmel Studio, the most advanced integrated environment for developing touch-enabled Atmel AVR® or ARM® Cortex®-M processor-based applications.

The following topics will be covered:

- Creating a QTouch Project using Atmel QTouch Composer along with the required source, header files and QTouch library. These outputs are used to configure the Peripheral Touch Controller of the Atmel | SMART SAM D21 MCU to monitor the touch sensors situated on the QT1 Xplained Pro extension kit.
- Atmel QTouch Analyzer to provide graphical representation and demonstration of real-time touch data.

Prerequisites

The following hardware and software are required for executing the demonstrations presented in this training manual.

- **Hardware Prerequisites**
  - Atmel | SMART SAM D21 Xplained Pro Evaluation Kit
  - Atmel QT1 Xplained Pro Kit
  - Micro-USB Cable (type-A / Micro-B)
- **Software Prerequisites**
  - Atmel Studio 7
  - Atmel Software Framework (ASF) 3.30.1 or later
  - Atmel QTouch Composer 5.9 or later
  - Atmel QTouch Library 5.9 or later

Estimated completion time: 105 minutes

Icon Key Identifiers

Following icons are used in this document to identify different assignment sections and to reduce complexity.

Info: Delivers contextual information about a specific topic

Tip: Highlights useful tips and techniques

To do: Highlights objectives to be completed

Result: Highlights the expected result of an assignment step

Warning: Indicates important information

Execute: Highlights actions to be executed out of the target when necessary

Table of Contents

Introduction......1

Prerequisites......1

Icon Key Identifiers......2

1. Training Module Architecture......4
   - 1.1. Atmel Studio Extension (.vsix)......4
   - 1.2. Atmel Training Executable (.exe)...... 4

2. Introduction to Atmel QTouch Composer......5

3. Assignment 1: Getting Started with the Atmel QTouch Project Builder...... 7
   - 3.1. Introduction to QTouch Project Builder......7
   - 3.2. Project Creation......7
   - 3.3. Conclusion......18

4. Assignment 2: Getting Started with the Atmel QTouch Library API......19
   - 4.1. Atmel QTouch Library Introduction...... 19
   - 4.2. QTouch Library API Files......19
   - 4.3. Reading and Using Button Sensor Status......21
   - 4.4. Conclusion......26

5. Assignment 3: Getting Started with the Atmel QTouch Analyzer......27
   - 5.1. Atmel QTouch Analyzer Introduction...... 27
   - 5.2. Basic Touch Sensor Debugging Information...... 27
   - 5.3. Conclusion......33

6. Assignment 4: Understanding Sensor Individual Parameters...... 34
   - 6.1. Detect Threshold...... 34
   - 6.2. Detect Hysteresis...... 37
   - 6.3. Adjacent Key Suppression (AKS)......40
   - 6.4. Position Resolution......47
   - 6.5. Position Hysteresis......47
   - 6.6. Conclusion......48

7. Summary...... 49

8. Revision History......50

1. Training Module Architecture

This training material is available through the following delivery modes:

- Atmel Studio Extension (.vsix file), which can be found on the Atmel Gallery website (http://gallery.atmel.com/) or using the Atmel Studio Extension manager
- Atmel Training Executable (.exe file) usually provided during Atmel Training sessions

Depending on the delivery type, the various resources required to complete this training (hands-on documentation, datasheets, application notes, software, and tools) will be found in different locations.

1.1. Atmel Studio Extension (.vsix)

After the extension has been installed, you can open and create the different projects associated with the training using the `New Example Project from ASF...` menu in Atmel Studio.

Info: The example projects installed through an extension are usually under `Atmel Training > Atmel Corp. Extension Name`.
There are different projects which can be available depending on the extension:

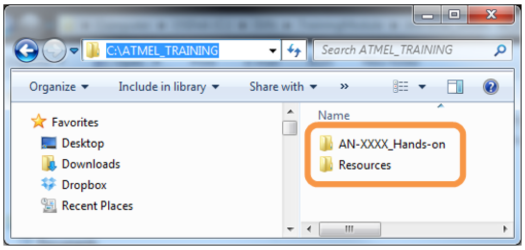
- **Hands-on Documentation**: contains the required documentation
- **Hands-on Assignment**: contains the initial project that may be required to start
- **Hands-on Solution**: contains the final application, which is a solution project for this hands-on

Info: Each time a reference is made to some resources in the following pages, the user must refer to the **Hands-on Documentation** project folder.

1.2. Atmel Training Executable (.exe)

Depending on where the executable has been installed, you will find the following architecture which is comprised of two main folders:

- `AN-XXXX_Hands-on`: contains the initial project that may be required to start and a solution
- `Resources`: contains required resources (datasheets, software, and tools…)



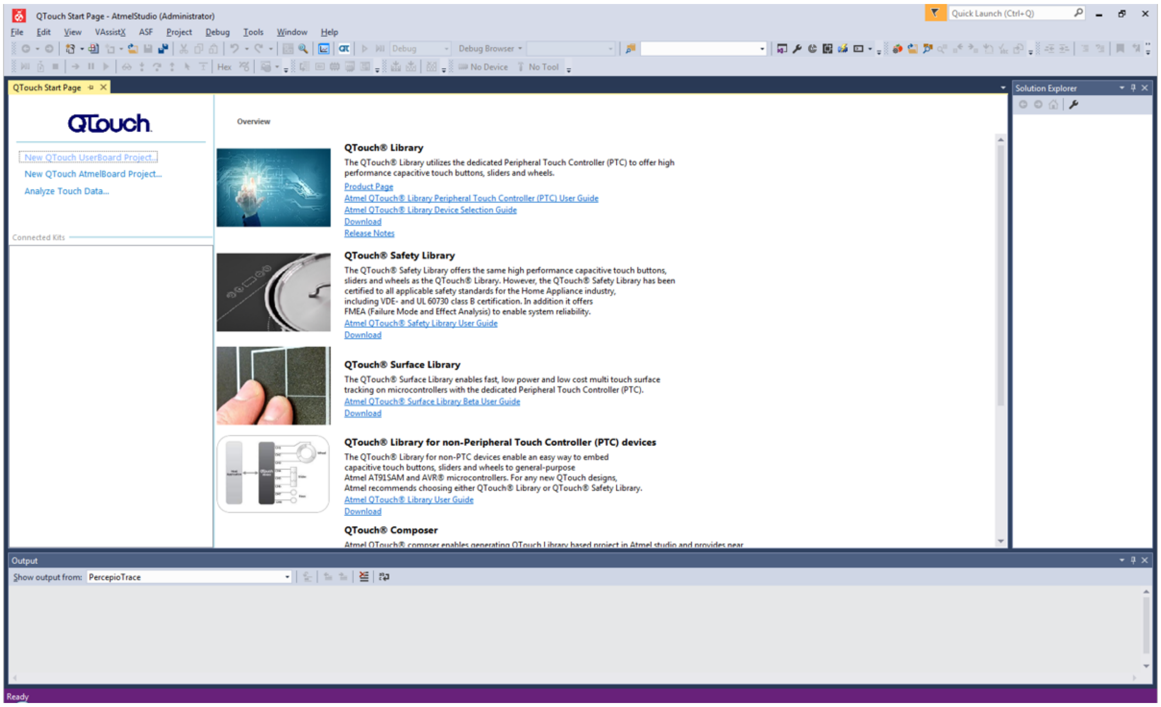
Info: Unless a specific location is specified, each time a reference is made to some resources in the following pages, the user must refer to this `Resources` folder.

2. Introduction to Atmel QTouch Composer

The Atmel QTouch Composer enables generating Atmel QTouch Library based projects and provides near real-time visualization of the QTouch data sent from the Atmel Touch kits. Using QTouch Composer, the user can perform run-time tuning of kits by setting various QTouch library parameters on the Touch kit.

QTouch Start page can be opened by clicking QT (**QT**) icon in the toolbar.

This start page provides an easy access for the user to create QTouch Atmel Board projects, QTouch User Board projects, and to analyze data from the Touch kits. It also contains information about the connected kits and a brief overview with basic information about the QTouch composer and QTouch Library.



QTouch® Library

The QTouch® Library utilizes the dedicated Peripheral Touch Controller (PTC) to offer high performance capacitive touch buttons, sliders and wheels.

Product Page
Atmel QTouch® Library Peripheral Touch Controller (PTC) User Guide
Atmel QTouch® Library Device Selection Guide
Download
Release Notes

QTouch® Safety Library

The QTouch® Safety Library offers the same high performance capacitive touch buttons, sliders and wheels as the QTouch® Library. However, the QTouch® Safety Library has been certified to all applicable safety standards for the Home Appliance industry, including VDE- and UL 60730 class B certification. In addition it offers FMEA (Failure Mode and Effect Analysis) to enable system reliability.

Atmel QTouch® Safety Library User Guide
Download

QTouch® Surface Library

The QTouch® Surface Library enables fast, low power and low cost multi touch surface tracking on microcontrollers with the dedicated Peripheral Touch Controller (PTC).

Atmel QTouch® Surface Library Beta User Guide
Download

QTouch® Library for non-Peripheral Touch Controller (PTC) devices

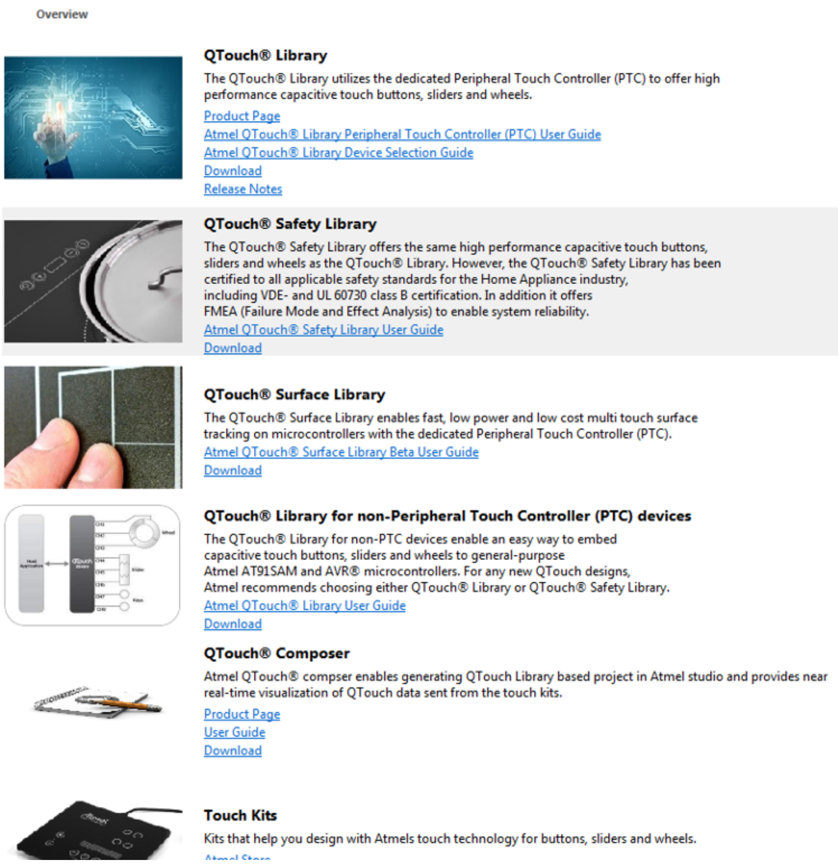
The QTouch® Library for non-PTC devices enable an easy way to embed capacitive touch buttons, sliders and wheels to general-purpose Atmel AT91SAM and AVR® microcontrollers. For any new QTouch designs, Atmel recommends choosing either QTouch® Library or QTouch® Safety Library.

Atmel QTouch® Library User Guide
Download

QTouch® Composer

Atmel QTouch® compser enables generating QTouch Library based project in Atmel studio and provides near

The right side section contains information about the QTouch Composer and QTouch Library. The relevant tutorials are also provided to get started with the new and existing touch kits quickly.



Overview

QTouch® Library

The QTouch® Library utilizes the dedicated Peripheral Touch Controller (PTC) to offer high performance capacitive touch buttons, sliders and wheels.

Product Page
Atmel QTouch® Library Peripheral Touch Controller (PTC) User Guide
Atmel QTouch® Library Device Selection Guide
Download
Release Notes

QTouch® Safety Library

The QTouch® Safety Library offers the same high performance capacitive touch buttons, sliders and wheels as the QTouch® Library. However, the QTouch® Safety Library has been certified to all applicable safety standards for the Home Appliance industry, including VDE- and UL 60730 class B certification. In addition it offers FMEA (Failure Mode and Effect Analysis) to enable system reliability.

Atmel QTouch® Safety Library User Guide
Download

QTouch® Surface Library

The QTouch® Surface Library enables fast, low power and low cost multi touch surface tracking on microcontrollers with the dedicated Peripheral Touch Controller (PTC).

Atmel QTouch® Surface Library Beta User Guide
Download

QTouch® Library for non-Peripheral Touch Controller (PTC) devices

The QTouch® Library for non-PTC devices enable an easy way to embed capacitive touch buttons, sliders and wheels to general-purpose Atmel AT91SAM and AVR® microcontrollers. For any new QTouch designs, Atmel recommends choosing either QTouch® Library or QTouch® Safety Library.

Atmel QTouch® Library User Guide
Download

QTouch® Composer

Atmel QTouch® compser enables generating QTouch Library based project in Atmel studio and provides near real-time visualization of QTouch data sent from the touch kits.

Product Page
User Guide
Download

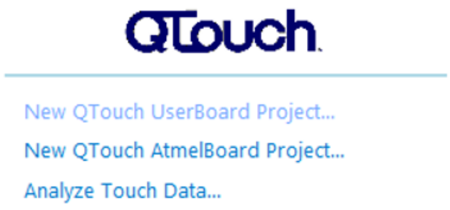
Touch Kits

Kits that help you design with Atmels touch technology for buttons, sliders and wheels.

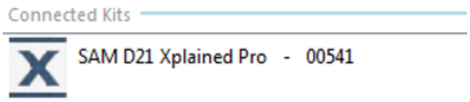
Atmel Store

The top-left section contains:

- **New QTouch UserBoard Project** - Allows the user to create new GCC C QTouch Executable user board projects through the QTouch project builder wizard.
- **New QTouch AtmelBoard Project** - Allows the user to create QTouch Atmel board example projects through the ASF wizard.
- **Analyze Touch Data** - Allows user to open the QTouch Analyzer tool window where the user can analyze multiple devices connected to the system.


QTouch
New QTouch UserBoard Project...
New QTouch AtmelBoard Project...
Analyze Touch Data...

The bottom-left section provides the lists of the QTouch kits connected to the system. When a new kit is connected, it gets listed on this page with an image of the kit, kit name, and serial number. Right click the kit in the **Connected Kits** view to open the QTouch Analyzer tool window and open the Atmel Board example projects related to the connected kit.



3. Assignment 1: Getting Started with the Atmel QTouch Project Builder

3.1. Introduction to QTouch Project Builder

The **QTouch Project Builder** is part of Atmel QTouch Composer, which guides the developers through the selection wizards of the device and capacitive-touch sensor. It helps to generate the Atmel QTouch Library based project. The wizard guides the developers through the flow of embedding capacitive-touch button, slider, and wheel functionality into general-purpose Atmel AVR and ARM-based microcontroller applications.

When connecting physical sensors to an MCU running the QTouch Library, the QTouch Library must be configured according to the physical connection, i.e. which pins are connected to which sensors. The project builder wizard guides the user in selecting and setting up the QTouch library according to the intended electrical connections between the physical sensors and the MCU.

Touch sensors can be based on two technologies (acquisition methods): Self-capacitance and Mutual capacitance. The project builder wizard supports both technologies. The wizard will guide the user through various pages where the user can provide the following settings.

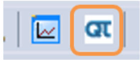
- Number and type of sensors
- Technology
- Device
- Port pin assignments for the sensors
- Touch data Interface

The wizard will validate the settings before the user proceeds to the next page. If the settings are invalid, the wizard will provide error messages and mark the invalid data fields. Hovering the mouse over the error marker will provide the description of the error and a possible fix.

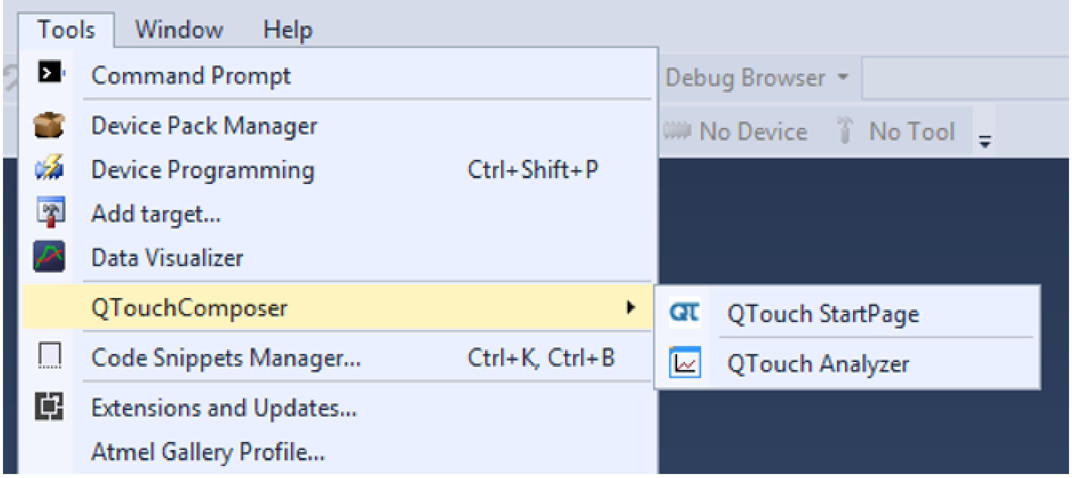
3.2. Project Creation

To do: Create a new QTouch Project.

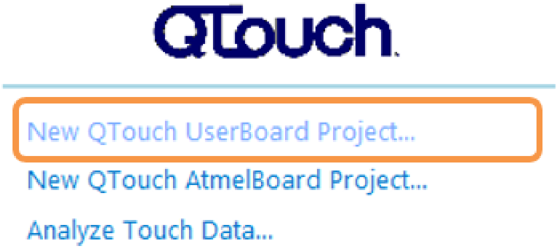
1. Open **Atmel Studio 7**.

2. Click on the **QT** icon to open the **QTouch Start Page**  .

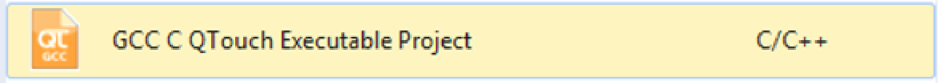
Info: **QTouch Start Page** is also accessible through the **Tools** menu.



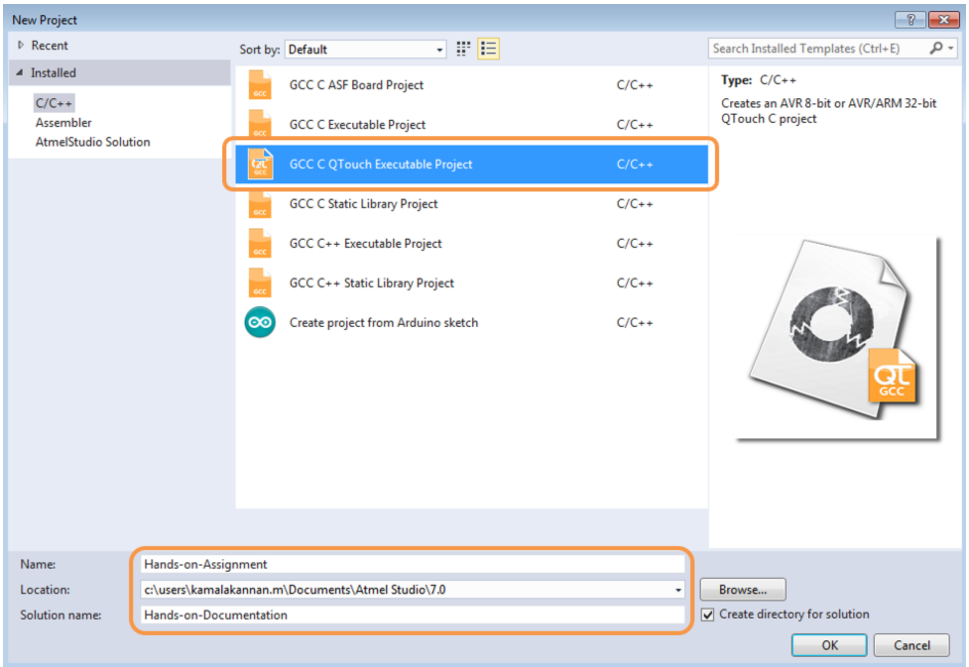
3. Select **New QTouch UserBoard Project…**.


QTouch
New QTouch UserBoard Project...
New QTouch AtmelBoard Project...
Analyze Touch Data...

4. Select **GCC C QTouch Executable Project**.



5. Fill in the New Project fields according to following use cases:
   - **Atmel Training Executable Case**
     - **Name:** Hands-on Assignment.
     - **Location:** `AN-7846_SAMD21-XPRO_QTouch_Parameters\assignments` (relative path in the ATMEL_TRAINING installation folder).
     - **Solution name:** Hands-on Assignment.
     - Click **OK**
   - **Atmel Extension Case (downloaded from Atmel Gallery or Studio Extension Manager)**
     - **Name:** Hands-on Assignment.
     - **Location:** existing Hands-on Documentation solution path.
     - **Solution name:** Hands-on Documentation.
     - Click **OK**



New Project

GCC C ASF Board Project	C/C++
GCC C Executable Project	C/C++
GCC C QTouch Executable Project	C/C++
GCC C Static Library Project	C/C++
GCC C++ Executable Project	C/C++
GCC C++ Static Library Project	C/C++
Create project from Arduino sketch	C/C++

Type: C/C++

Creates an AVR 8-bit or AVR/ARM 32-bit QTouch C project

Name: Hands-on-Assignment

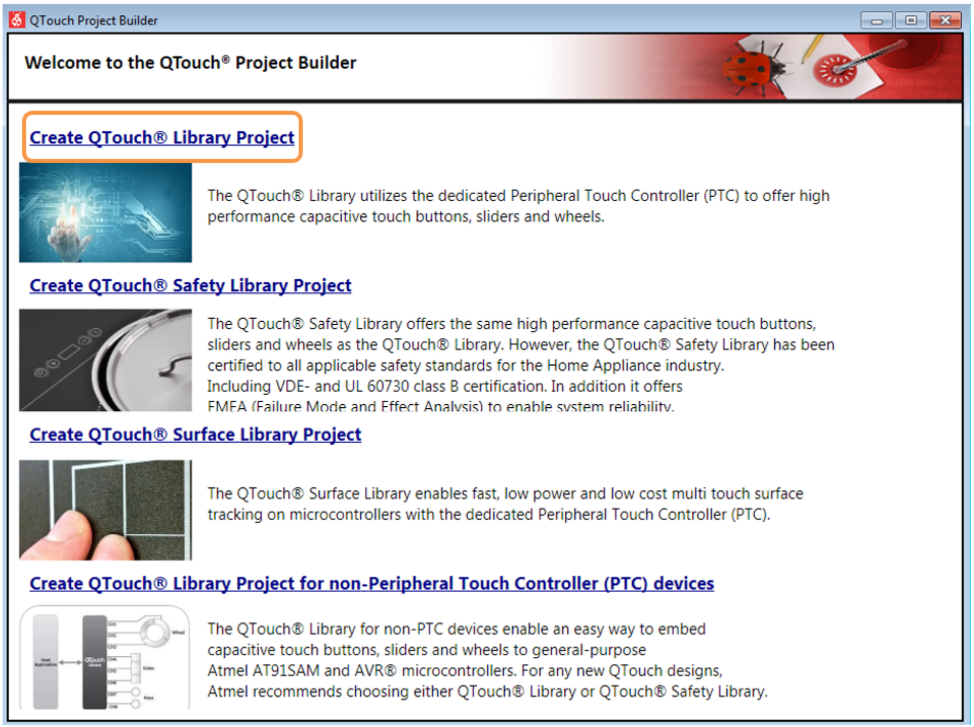
Location: c:\users\kamalakannan.m\Documents\Atmel Studio\7.0

Solution name: Hands-on-Documentation

☑ Create directory for solution

Info: The **QTouch Project Builder** wizard will now guide you through the steps involved in creating a QTouch project.

6. Click **Create QTouch® Library Project** in the **QTouch Project Builder**.



QTouch Project Builder

Welcome to the QTouch® Project Builder

Create QTouch® Library Project

The QTouch® Library utilizes the dedicated Peripheral Touch Controller (PTC) to offer high performance capacitive touch buttons, sliders and wheels.

Create QTouch® Safety Library Project

The QTouch® Safety Library offers the same high performance capacitive touch buttons, sliders and wheels as the QTouch® Library. However, the QTouch® Safety Library has been certified to all applicable safety standards for the Home Appliance industry. Including VDE- and UL 60730 class B certification. In addition it offers FMEA (Failure Mode and Effect Analysis) to enable system reliability.

Create QTouch® Surface Library Project

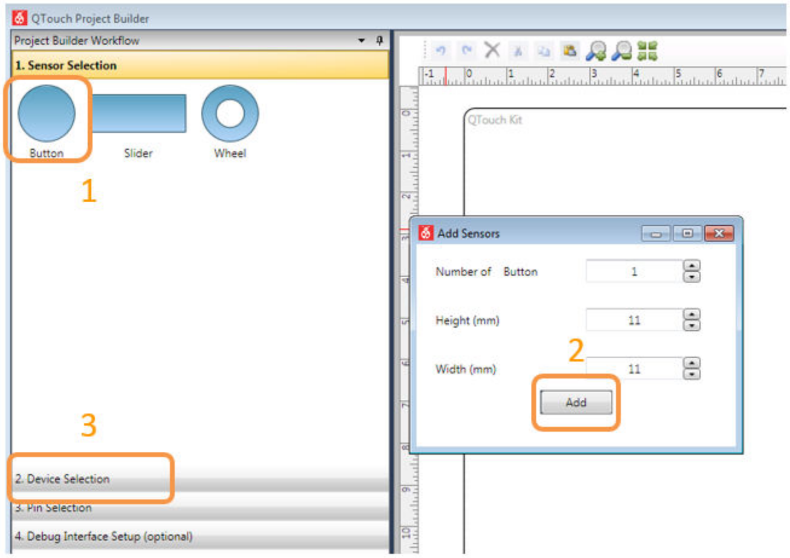
The QTouch® Surface Library enables fast, low power and low cost multi touch surface tracking on microcontrollers with the dedicated Peripheral Touch Controller (PTC).

Create QTouch® Library Project for non-Peripheral Touch Controller (PTC) devices

The QTouch® Library for non-PTC devices enable an easy way to embed capacitive touch buttons, sliders and wheels to general-purpose Atmel AT91SAM and AVR® microcontrollers. For any new QTouch designs, Atmel recommends choosing either QTouch® Library or QTouch® Safety Library.

Info: The **Sensor Selection** page provides an option to add, delete, or move QTouch sensors such as button, wheel, and slider. It also provides options to set up the physical and firmware properties of the board and sensors.

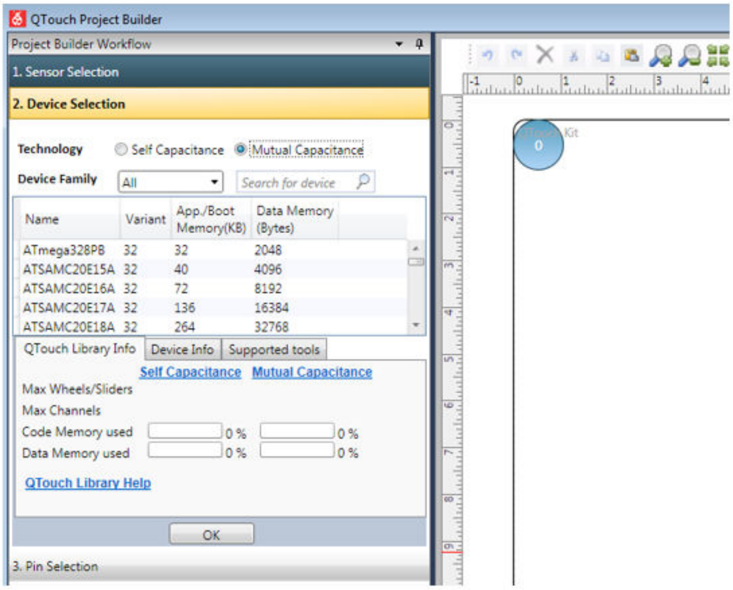
7. Add a button in the **Sensor Selection** page by clicking on the **Button** icon, then the **Add**, and the **Device Selection**.



Number of Button: 1
Height (mm): 11
Width (mm): 11

Info: The **Device Selection** page lets you select the specific device and acquisition technology to use, i.e. Self-capacitance or Mutual capacitance.

8. We will consider for this hands-on that our application hardware design may be exposed to moisture or water droplets. So, select **Mutual capacitance** in the radio button selection for **Technology**.



QTouch Project Builder

Project Builder Workflow

1. Sensor Selection

2. Device Selection

Technology ○ Self Capacitance ● Mutual Capacitance

Device Family All

Search for device

Name	Variant	App./Boot Memory(KB)	Data Memory (Bytes)
ATmega328PB	32	32	2048
ATSAMC20E15A	32	40	4096
ATSAMC20E16A	32	72	8192
ATSAMC20E17A	32	136	16384
ATSAMC20E18A	32	264	32768

QTouch Library Info | Device Info | Supported tools

	Self Capacitance	Mutual Capacitance
Max Wheels/Sliders		
Max Channels		
Code Memory used	0 %	0 %
Data Memory used	0 %	0 %

QTouch Library Help

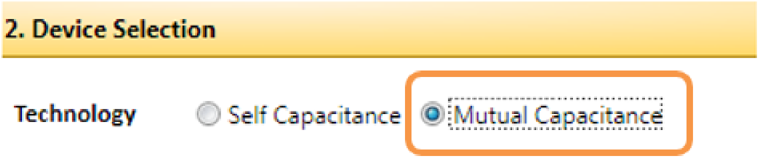
OK

3. Pin Selection

Info: For more information about designing the touch sensor, refer to *QTAN0079: Buttons*, *Sliders and Wheels Sensor Design Guide* available at www.atmel.com/images/doc10752.pdf.

Info: The **Device Selection** page provides the option to view the number of QTouch sensors such as button, wheel, and slider that was selected in the **Sensor Selection** page. Touch sensors can be based on two sub-technologies; **Self-capacitance** and **Mutual capacitance**.

9. Verify that **Mutual Capacitance** has been correctly selected.

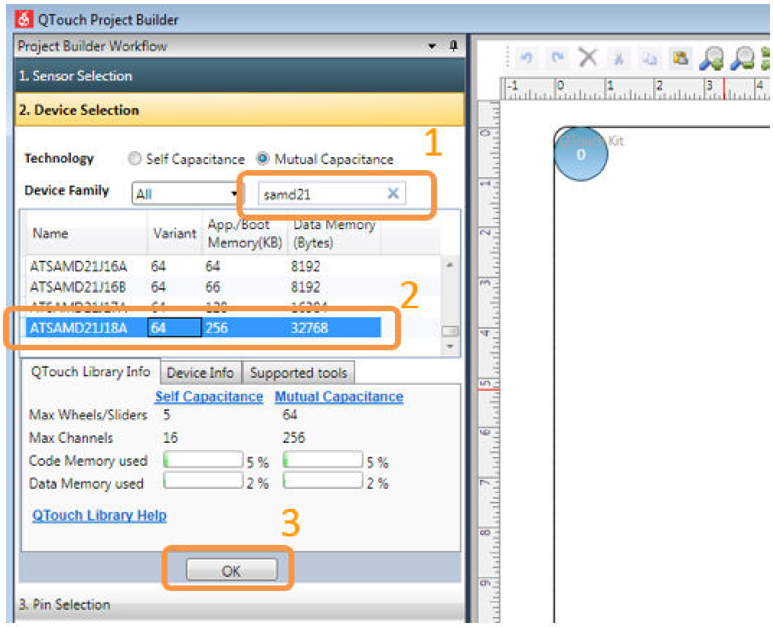


2. Device Selection

Technology ○ Self Capacitance ● Mutual Capacitance

Info: The different tabs display the device details such as QTouch Library Info, Device Info, and Supported tools. Data listing will vary based on device family, the number of selected sensors, and technology as only devices supporting all choices made will be shown. Additionally the user has the option to search for a particular device.

10. Type **SAMD21** in the search box to make a first filtering.
11. Select **ATSAMD21J18A** in the **Device Selection** page and click **OK**. This is the device on the SAM D21 Xplained Pro kit.



QTouch Project Builder

Project Builder Workflow

1. Sensor Selection

2. Device Selection

Technology ○ Self Capacitance ● Mutual Capacitance

Device Family All samd21

Name	Variant	App./Boot Memory(KB)	Data Memory (Bytes)
ATSAMD21J16A	64	64	8192
ATSAMD21J16B	64	66	8192
ATSAMD21J17A	64	128	16384
ATSAMD21J18A	64	256	32768

QTouch Library Info | Device Info | Supported tools

	Self Capacitance	Mutual Capacitance
Max Wheels/Sliders	5	64
Max Channels	16	256
Code Memory used	5 %	5 %
Data Memory used	2 %	2 %

QTouch Library Help

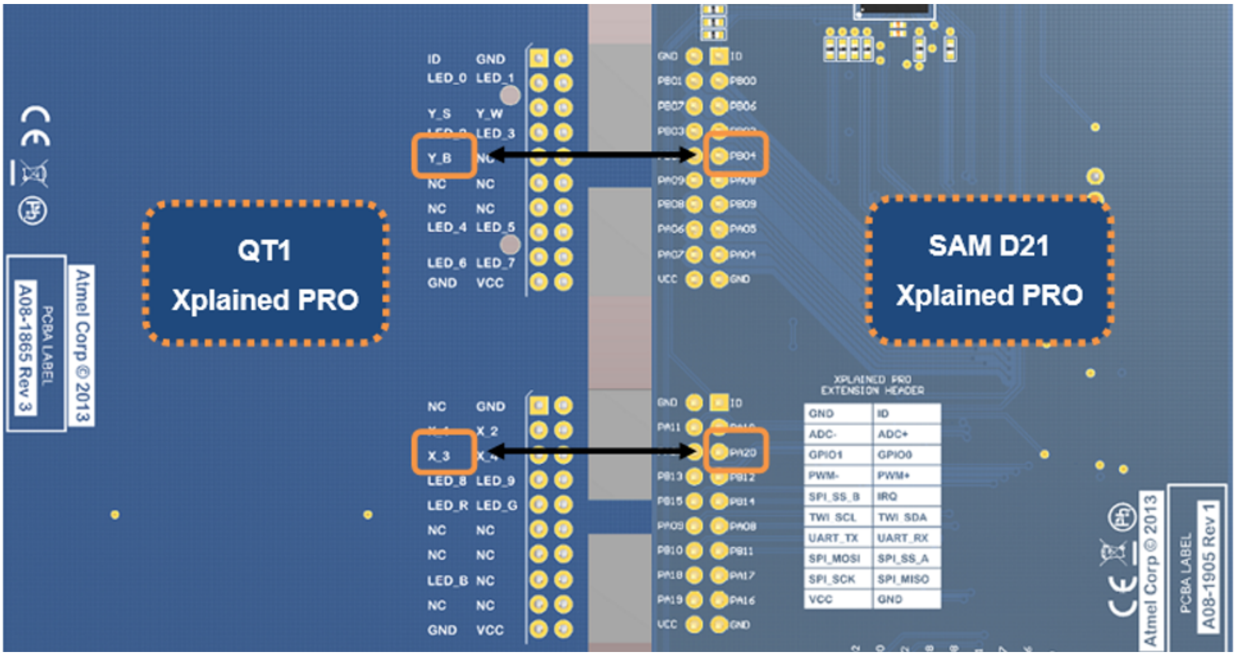
OK

3. Pin Selection

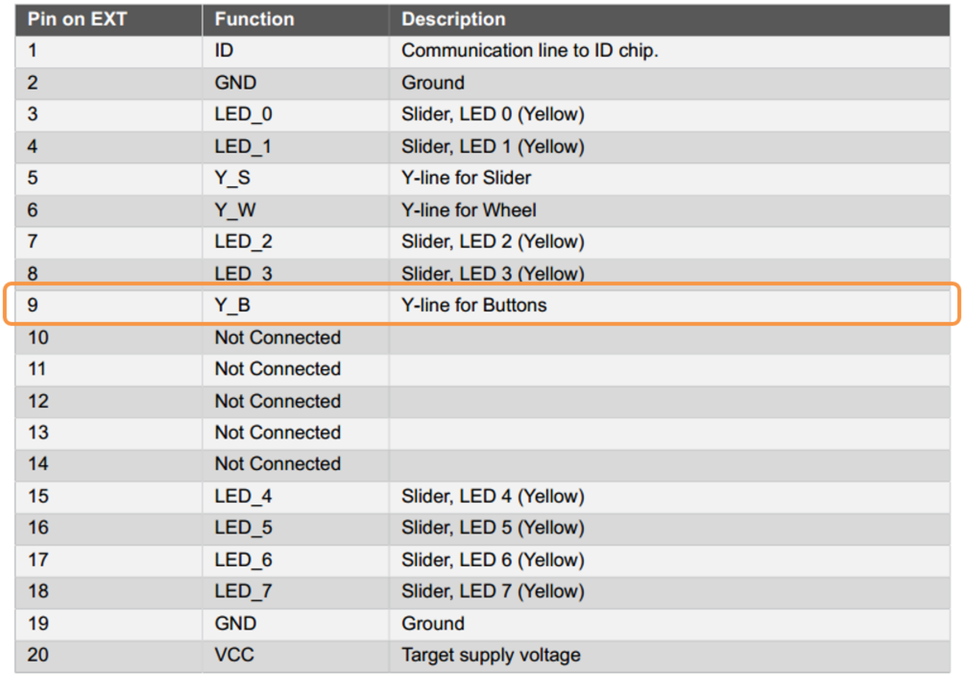
Info: Mutual capacitance method uses a pair of sensing electrodes for each touch channel. These electrodes are denoted as X and Y lines. A mutual capacitance touch button sensor is formed using a single X-Y channel combination, while a touch rotor or slider sensor is formed using three to eight X-Y channel combinations.
For sensors made of several channels, wheels, and sliders, all channels must use the same Y-Line.

Warning: The X-line and Y-line numbering in the `Atmel QT1 Xplained Pro User Guide` (http://www.atmel.com/Images/Atmel-42193-QT1-Xplained-Pro_User-Guide.pdf) does not correspond to the X-line and Y-line numbering in the SAM D21 PTC module as the QT1 Xplained Pro is a generic touch board (not dedicated to the SAM D21).

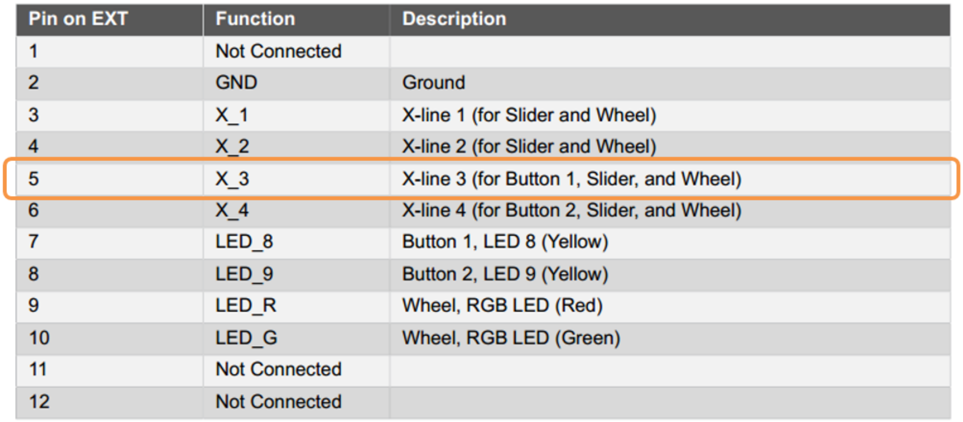
12. Flip the SAM D21 Xplained Pro and QT1 Xplained Pro boards and determine which SAM D21 I/O pins that are connected to Button 1.


QT1
Xplained PRO
SAM D21
Xplained PRO
ID GND
LED_0 LED_1
Y_S Y_W
LED_3
Y_B NC
NC NC
NC NC
LED_4 LED_5
LED_6 LED_7
GND VCC
GND ID
PB01 PB00
PB07 PB06
PB03
PB04
PA09
PB08 PB09
PA06 PA05
PA07 PA04
VCC GND
NC GND
X_2
X_3 X_4
LED_8 LED_9
LED_R LED_G
NC NC
NC NC
LED_B NC
NC NC
GND VCC
GND ID
PA11
PA20
PB13 PB12
PB15 PB14
PA09 PA08
PB10 PB11
PA18 PA17
PA19 PA16
VCC GND
XPLAINED PRO
EXTENSION HEADER
GND ID
ADC- ADC+
GPIO1 GPIO0
PWM- PWM+
SPI_SS_B IRQ
TWI SCL TWI SDA
UART_TX UART_RX
SPI_MOSI SPI_SS_A
SPI_SCK SPI_MISO
VCC GND
Atmel Corp © 2013
PCBA LABEL
A08-1865 Rev 3
A08-1905 Rev 1

Info: The X-line and Y-line for Button 1 can be retrieved from the `Atmel QT1 Xplained Pro User Guide` (http://www.atmel.com/Images/Atmel-42193-QT1-Xplained-Pro_User-Guide.pdf).

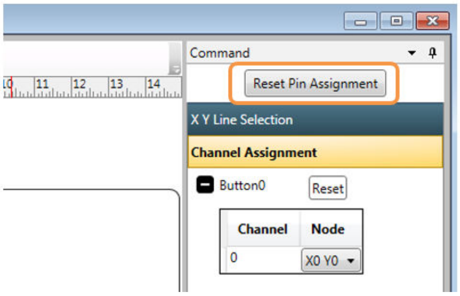


Pin on EXT	Function	Description
1	ID	Communication line to ID chip.
2	GND	Ground
3	LED_0	Slider, LED 0 (Yellow)
4	LED_1	Slider, LED 1 (Yellow)
5	Y_S	Y-line for Slider
6	Y_W	Y-line for Wheel
7	LED_2	Slider, LED 2 (Yellow)
8	LED_3	Slider, LED 3 (Yellow)
9	Y_B	Y-line for Buttons
10	Not Connected	
11	Not Connected	
12	Not Connected	
13	Not Connected	
14	Not Connected	
15	LED_4	Slider, LED 4 (Yellow)
16	LED_5	Slider, LED 5 (Yellow)
17	LED_6	Slider, LED 6 (Yellow)
18	LED_7	Slider, LED 7 (Yellow)
19	GND	Ground
20	VCC	Target supply voltage



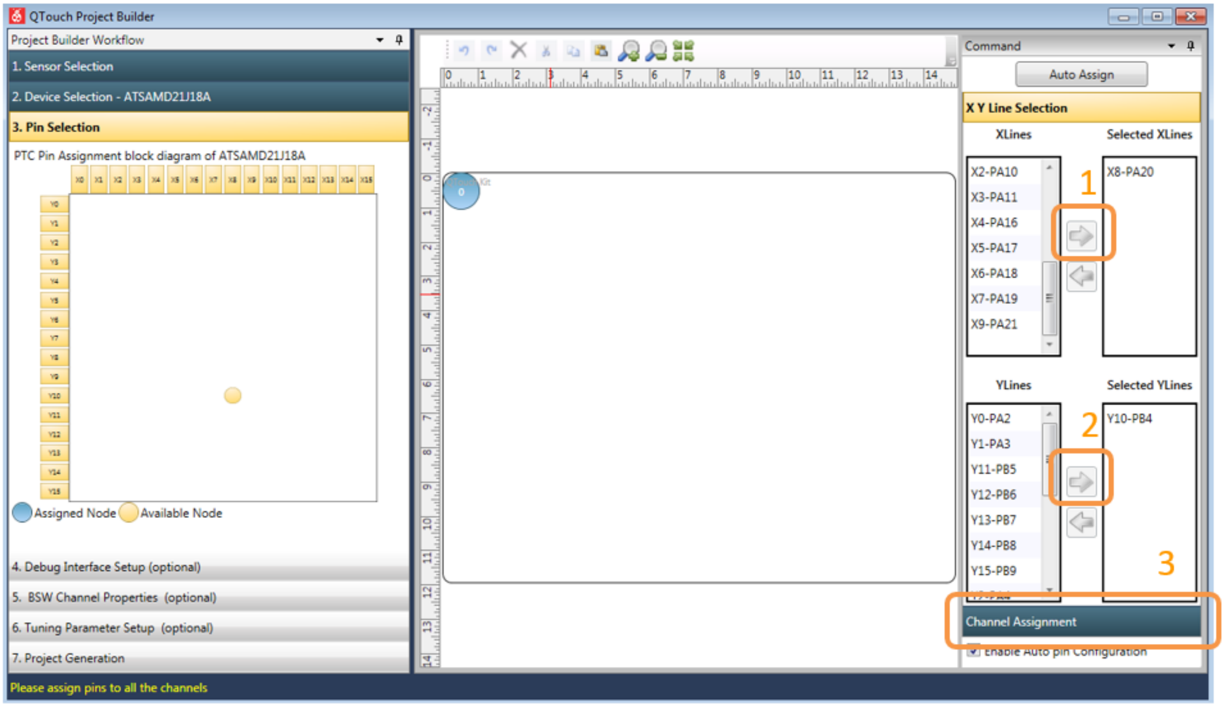
Pin on EXT	Function	Description
1	Not Connected	
2	GND	Ground
3	X_1	X-line 1 (for Slider and Wheel)
4	X_2	X-line 2 (for Slider and Wheel)
5	X_3	X-line 3 (for Button 1, Slider, and Wheel)
6	X_4	X-line 4 (for Button 2, Slider, and Wheel)
7	LED_8	Button 1, LED 8 (Yellow)
8	LED_9	Button 2, LED 9 (Yellow)
9	LED_R	Wheel, RGB LED (Red)
10	LED_G	Wheel, RGB LED (Green)
11	Not Connected	
12	Not Connected	

13. Reset the automatic settings by pressing the **Reset Pin Assignment** button in the upper right corner of the **QTouch Project Builder**.

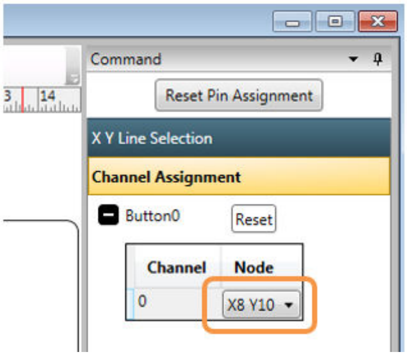


Channel	Node
0	X0 Y0

14. Use the **X Y Line Selection** option of **Pin Selection** page to select the correct X- and Y-lines, and click **Channel Assignment**.



15. We also need to select the X-Y node for each channel.



Channel	Node
0	X8 Y10

16. Click **Debug Interface Setup (optional)**.

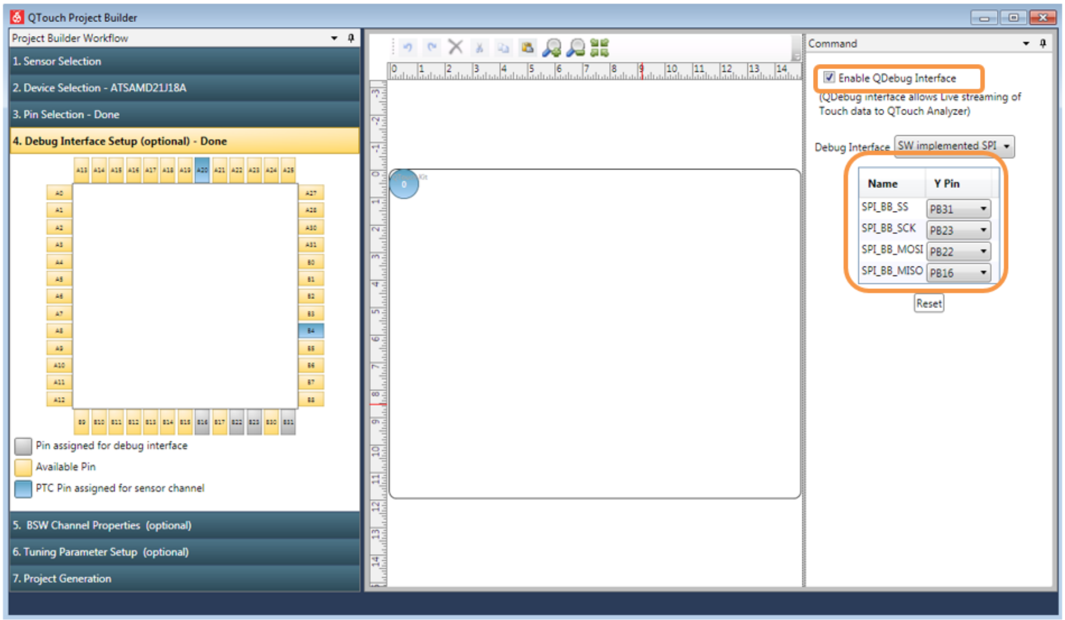
17. Check the **Enable QDebug Interface** checkbox, which allows live streaming of touch data with **QTouch Analyzer**.

Info: QDebug is the touch data protocol used by **QTouch Analyzer** to communicate with the SAM D21 through the Atmel **D**ata **G**ateway **I**nterface (**DGI**).
The Atmel Embedded Debugger (**EDBG**) offers a Data Gateway Interface (**DGI**) for streaming data to a host PC. This is meant as an aid in debugging and demonstration of features in the application running on the target device. DGI consists of multiple interfaces for data streaming. The supported interfaces are SPI, USART, TWI, and GPIO.

Tip: SAM D21 XPRO Kit Data Gateway Interface supports SPI or I²C.

18. Assign the right port(s) and pins to the SPI interface. These can be found in the `SAM D21 Xplained Pro User Guide` on the `Advanced Options` page (http://www.atmel.com/Images/Atmel-42220-SAMD21-Xplained-Pro_User-Guide.pdf).

Pin on SAM D21	Function
PB31	SERCOM5 PAD[1] SPI SS (Slave select) (SAM D21 is Master)
PB16	SERCOM5 PAD[0] SPI MISO (Master In, Slave Out)
PB22	SERCOM5 PAD[2] SPI MOSI (Master Out, Slave in)
PB23	SERCOM5 PAD[3] SPI SCK (Clock Out)



QTouch Project Builder

Project Builder Workflow

1. Sensor Selection
2. Device Selection - ATSAMD21J18A
3. Pin Selection - Done
4. Debug Interface Setup (optional) - Done
5. BSW Channel Properties (optional)
6. Tuning Parameter Setup (optional)
7. Project Generation

- Pin assigned for debug interface
- Available Pin
- PTC Pin assigned for sensor channel

Command

☑ Enable QDebug Interface

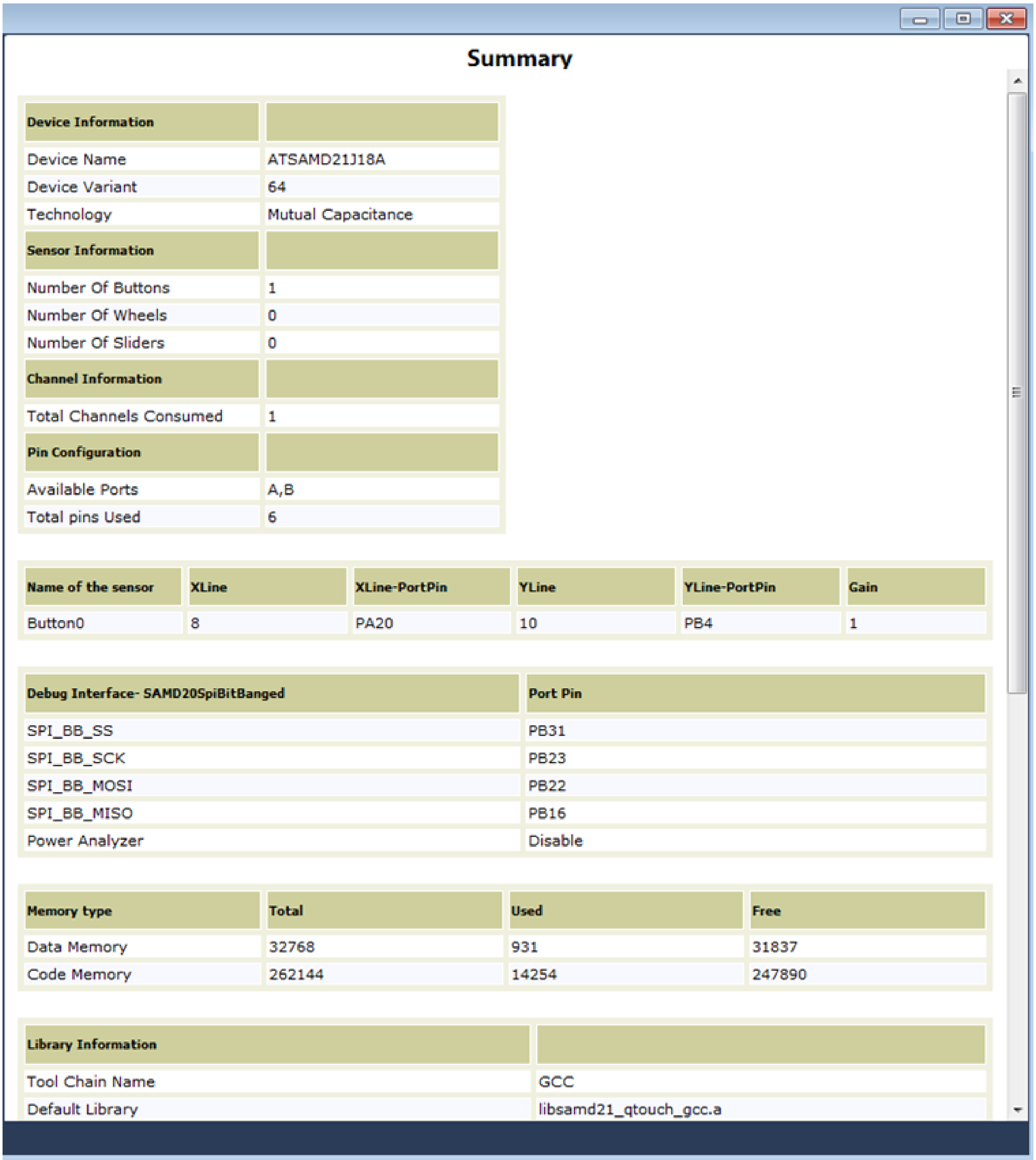
(QDebug interface allows Live streaming of Touch data to QTouch Analyzer)

Debug Interface: SW implemented SPI

Name	Y Pin
SPI_BB_SS	PB31
SPI_BB_SCK	PB23
SPI_BB_MOSI	PB22
SPI_BB_MISO	PB16

Reset

19. Click **Project Generation** as default BSW and Tuning parameters are considered as OK for our application.
20. Review the **Summary** page which provides a summary of the settings provided by the previous set of wizard pages. Then, click on **Finish** and wait for your touch project to be generated.



Summary

Device Information	
Device Name	ATSAMD21J18A
Device Variant	64
Technology	Mutual Capacitance
Sensor Information	
Number Of Buttons	1
Number Of Wheels	0
Number Of Sliders	0
Channel Information	
Total Channels Consumed	1
Pin Configuration	
Available Ports	A,B
Total pins Used	6

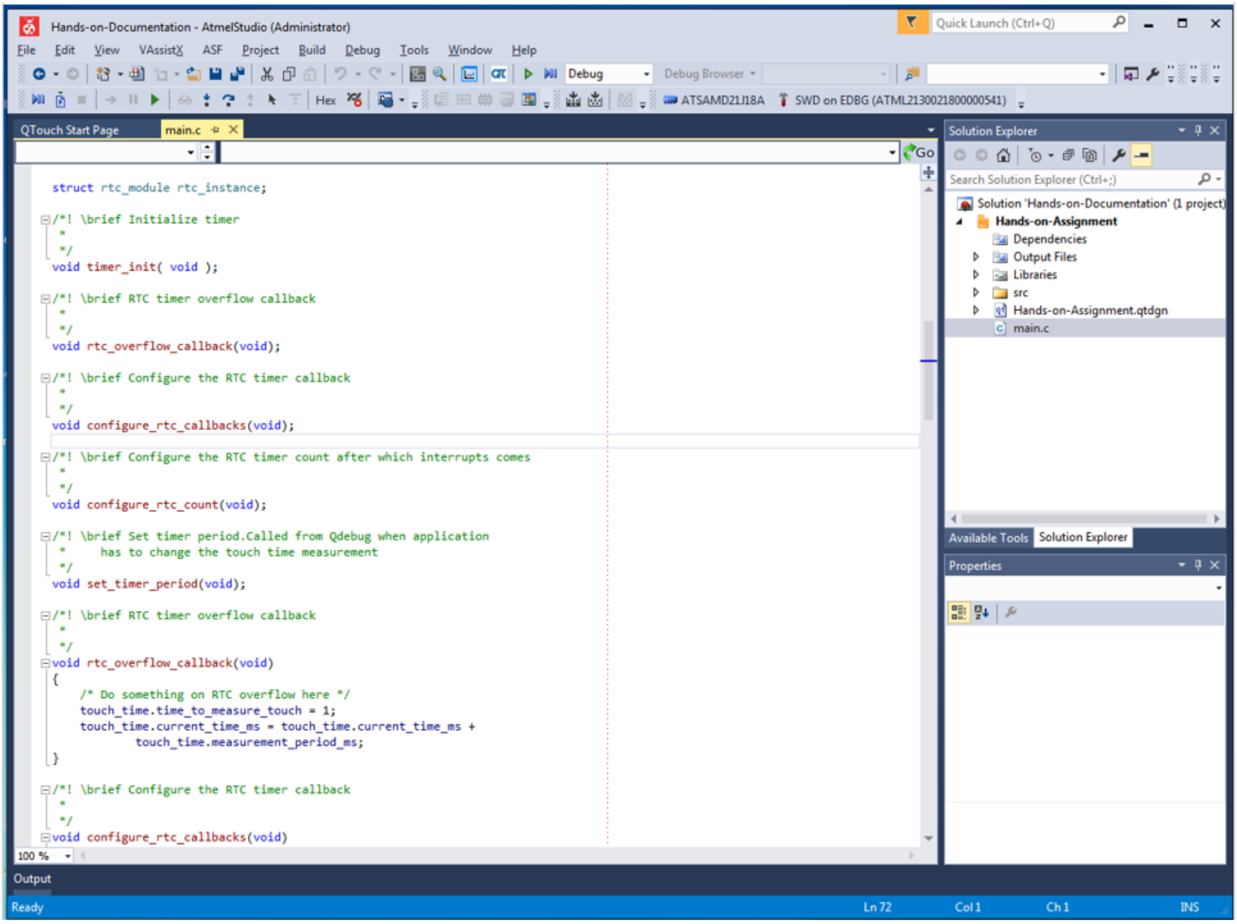
Name of the sensor	XLine	XLine-PortPin	YLine	YLine-PortPin	Gain
Button0	8	PA20	10	PB4	1

Debug Interface- SAMD20SpiBitBanged	Port Pin
SPI_BB_SS	PB31
SPI_BB_SCK	PB23
SPI_BB_MOSI	PB22
SPI_BB_MISO	PB16
Power Analyzer	Disable

Memory type	Total	Used	Free
Data Memory	32768	931	31837
Code Memory	262144	14254	247890

Library Information	
Tool Chain Name	GCC
Default Library	libsamd21_qtouch_gcc.a

✓ **Result:** You have created your first QTouch project with a Mutual Capacitance technology touch button.



Hands-on-Documentation - AtmelStudio (Administrator)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
ATSAMD21J18A
SWD on EDBG (ATML2130021800000541)
QTouch Start Page
main.c
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Hands-on-Documentation' (1 project)
Hands-on-Assignment
Dependencies
Output Files
Libraries
src
Hands-on-Assignment.qtdgn
main.c
Available Tools
Solution Explorer
Properties
100 %
Output
Ready
Ln 72
Col 1
Ch 1
INS

```
struct rtc_module rtc_instance;

/*! \brief Initialize timer
 *
 */
void timer_init( void );

/*! \brief RTC timer overflow callback
 *
 */
void rtc_overflow_callback(void);

/*! \brief Configure the RTC timer callback
 *
 */
void configure_rtc_callbacks(void);

/*! \brief Configure the RTC timer count after which interrupts comes
 *
 */
void configure_rtc_count(void);

/*! \brief Set timer period.Called from Qdebug when application
 *     has to change the touch time measurement
 */
void set_timer_period(void);

/*! \brief RTC timer overflow callback
 *
 */
void rtc_overflow_callback(void)
{
    /* Do something on RTC overflow here */
    touch_time.time_to_measure_touch = 1;
    touch_time.current_time_ms = touch_time.current_time_ms +
            touch_time.measurement_period_ms;
}

/*! \brief Configure the RTC timer callback
 *
 */
void configure_rtc_callbacks(void)
```

3.3. Conclusion

In the first assignment, you have learned:

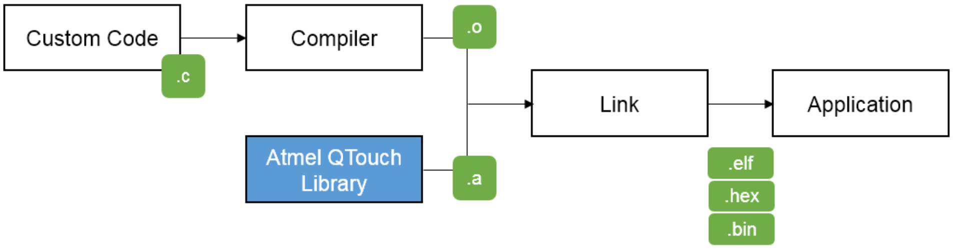
- How to create a QTouch Project using QTouch Composer Project Builder
- How easy it is to use QTouch Composer to design a QTouch project on the SAM D21 MCUs

4. Assignment 2: Getting Started with the Atmel QTouch Library API

4.1. Atmel QTouch Library Introduction

Atmel QTouch Library makes it easy for developers to embed capacitive-touch button, slider, and wheel functionality into general-purpose Atmel AT91SAM and AVR® microcontroller applications.

The QTouch Library is royalty free and provides library files (for both IAR™ and GCC) for each device and supports different numbers of touch channels, enabling both flexibility and efficiency in touch applications.



The QTouch Library API for SAM D21 Peripheral Touch Controller (PTC) can be used for touch sensor pin configuration, acquisition parameter setting, as well as periodic sensor data capture and status update operations.

The QTouch Library interfaces with the PTC module to perform the necessary actions.

The PTC module interfaces with the external capacitive touch sensors and is capable of performing self-capacitance and mutual capacitance method measurements.

4.2. QTouch Library API Files

The table below lists the files required to be able to use the QTouch library functions.

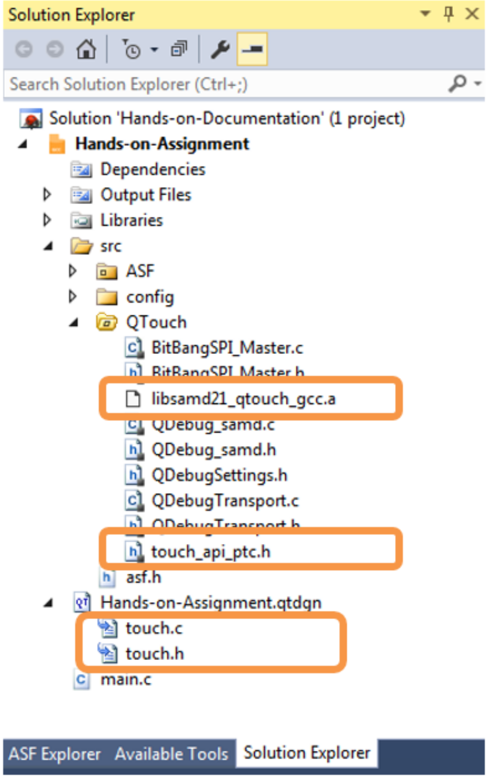
Table 4-1. File Description

File	Description
`touch_api_ptc.h`	QTouch Library API header file, contains API and Data structure used to interface with the library
`touch.h`	QTouch library configuration header file generated by QTouch Project Builder
`touch.c`	A helper file generated by QTouch Project Builder to demonstrate QTouch library initialization and sensors' configuration
`libsamd21_qtouch_iar.a` and `libsamd21_qtouch_gcc.a`	QTouch library compiled for IAR and GCC compiler that supports both self-capacitance and mutual capacitance sensors.

In order to add QTouch functionality into an existing user example project, these files and associated library based on the compiler should be added to the user project.

Info: This process is automatically performed when using the QTouch Project Builder.

To do: Check that these files have been added to your QTouch project.


Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'Hands-on-Documentation' (1 project)
  Hands-on-Assignment
    Dependencies
    Output Files
    Libraries
    src
      ASF
      config
      QTouch
        BitBangSPI_Master.c
        BitBangSPI_Master.h
        libsamd21_qtouch_gcc.a
        QDebug_samd.c
        QDebug_samd.h
        QDebugSettings.h
        QDebugTransport.c
        QDebugTransport.h
        touch_api_ptc.h
      asf.h
    Hands-on-Assignment.qtdgn
      touch.c
      touch.h
    main.c
```

The `touch.c` and `touch.h` are helper files, which are generated by QTouch Project Builder to help the user to get started very quickly on the QTouch library initialization and the different sensors configuration.

Warning: These files are automatically updated each time the QTouch Project Builder is used. So editing directly these files must be avoided as they will be overwritten by the QTouch Project Builder Wizard.

The `touch.c` pre-implements the following important functions:

- `touch_sensors_init:`
  - Initializes the QTouch library as well as the PTC peripheral. It also initializes the mutual capacitance or self-capacitance method specific pin, register, and global sensor configuration.
  - Configures each individual sensor.
  - Calibrates all the configured sensors and prepares them for normal operation.
- `touch_sensors_measure:` initiates a sensor measurement on all the configured sensors.
- `measure_complete_callback:` this function is called by the library when the touch measurement process is completed.

4.3. Reading and Using Button Sensor Status

You are now ready to understand how to use the QTouch Library to initialize a QTouch Button and use it to control a standard LED state.

Info: The following functions/structures also exist for the self-capacitance method.

The library returns sensor ON/OFF status as part of the measure data structure touch_measure_data_t. A pointer to this data structure is declared in touch.c and is initialized by the QTouch library after the touch_sensors_init is called.

```
/* ! Mutual capacitance method measured data pointer. */
touch_measure_data_t *p_mutlcap_measure_data = NULL;
```

The following macro implemented in touch_api_ptc.h can be used to read the sensor status.

```
/* ! Touch ON is 1u - Touch OFF is 0u */
GET_MUTLCAP_SENSOR_STATE(SENSOR_NUMBER);
```

Info: The order the sensors are added to the project is important as this assigns a SENSOR_NUMBER value for each sensor.

The sensor touch status must be read by the application only after the measurement_done_touch flag has been set by the measure_complete_callback() function.

So, the following code can be used to read the sensor status.

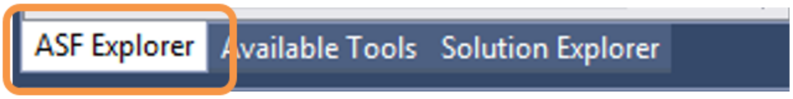
```
/* ! Start Touch Sensor Measurement. */
touch_sensors_measure();
/* Update touch status once measurement complete flag is set. */
uint8_t sensor_state;
if ((p_mutlcap_measure_data->measurement_done_touch == 1u)){
    sensor_state = GET_MUTLCAP_SENSOR_STATE(SENSOR_NUMBER);
    }
```

To do: Implement LED8 Configuration.

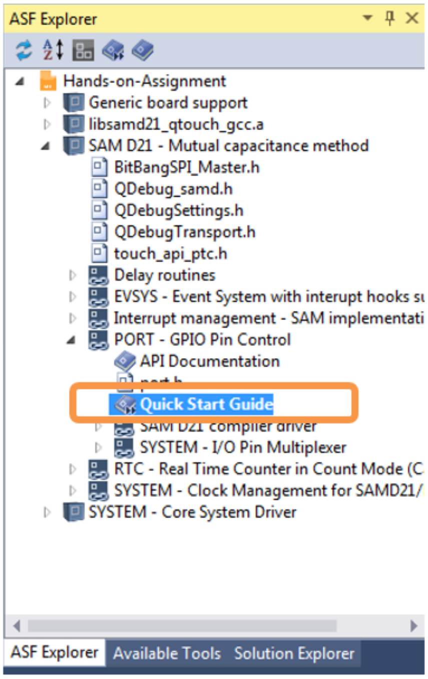
To control the LED8 using the Touch BUTTON1, we first need to configure it in our main.c file. The LED8 is controlled by a standard I/O using the SAM D21 Port Driver (PORT peripheral).

Its corresponding ASF driver for SAM D devices provides an interface for the configuration and management of the device's General Purpose Input/Output (GPIO) pin functionality, for manual pin state reading and writing.

Info: Using the Atmel Studio ASF Explorer, it is very simple to find examples to quickly implement peripherals functions.



1. Click on ASF Explorer and then double click on: **SAM D21 – Mutual capacitance method > PORT – GPIO Pin Control > Quick Start Guide**.



Warning: You need internet access to view the Quick Start Guide as all ASF documentation is on the web, available at asf.atmel.com. An extract of it is available below.

Info: Here is the workflow of a standard **Output** initialization extracted from the Quick Start Guide.

- Create a PORT module pin configuration struct, which can be filled out to adjust the configuration of a single port pin.
```
struct port_config config_port_pin;
```
- Initialize the pin configuration struct with the module's default values.
```
port_get_config_defaults(&config_port_pin);
```
- Adjust the configuration struct to request an output pin.
```
config_port_pin.direction = PORT_PIN_DIR_OUTPUT;
```

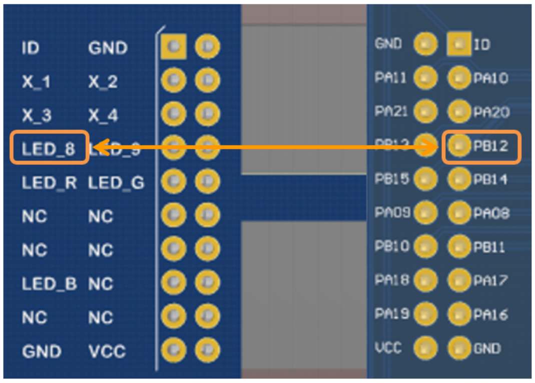
    – Configure the LED pin with the initialized pin configuration struct, to enable the output driver on the pin.
      port_pin_set_config(LED_x_PIN, &config_port_pin);

2. Open the main.c file and implement a function called configure_port_pins(void) just before the main() function, which configures the LED8 related I/O Port Pin as an output.

```
static void configure_port_pins(void)
{
    struct port_config config_port_pin;
    port_get_config_defaults(&config_port_pin);

    config_port_pin.direction = PORT_PIN_DIR_OUTPUT;
    port_pin_set_config(PIN_PB12, &config_port_pin);
}
```

Info: PIN_PB12 is used to control the LED8. This can be verified by looking at both the SAMD 21 and QT1 Xplained Pro User Guides or simply looking at the silkscreen at the bottom of each board.



3. Call the configure_port_pins() function in main() function just before entering the while(1) loop.

```
configure_port_pins();

while (1) {
```

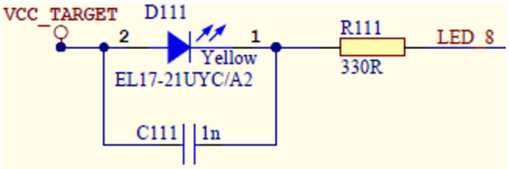
4. Add a call to the port_pin_set_output_level() function just after configure_port_pins() to clear the LED8.

```
configure_port_pins();
port_pin_set_output_level(PIN_PB12, true);

while (1) {
```

Info: port_pin_set_output_level() function sets the state of a port pin that is configured as an output.

Tip: The second parameter is the logical level to set the given pin to. In our case, LED8 will be turned OFF when PIN_PB12 is set.



5. Implement the Touch Button status management after the `touch_sensors_measure()` function (which is already implemented) using the QTouch Library API flag and macro we have described previously.

```
while (1) {
    /**
    * Go to STANDBY sleep mode, unless woken by
    * timer or PTC interrupt.
    */
    system_sleep();

    /**
    * Start touch sensor measurement
    */
    touch_sensors_measure();

    /* Update touch status once measurement
    * complete flag is set.
    */
    uint8_t sensor_state;

    if ((p_mutlcap_measure_data->measurement_done_touch == 1u))
    {
        sensor_state = GET_MUTLCAP_SENSOR_STATE(0);
    }
```

6. Ensure to reset the `measurement_done_touch` flag as soon as a touch measurement has been completed.

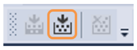
```
if ((p_mutlcap_measure_data->measurement_done_touch == 1u))
{
    p_mutlcap_measure_data->measurement_done_touch = 0u;
    sensor_state = GET_MUTLCAP_SENSOR_STATE(0);
}
```

7. Set or Clear LED8 based on the `sensor_state` value.

```
if ((p_mutlcap_measure_data->measurement_done_touch == 1u))
{
    p_mutlcap_measure_data->measurement_done_touch = 0u;
    sensor_state = GET_MUTLCAP_SENSOR_STATE(0);
    port_pin_set_output_level(PIN_PB12, !sensor_state);
}
```

Tip: `GET_MUTLCAP_SENSOR_STATE` returns ‘1’ if the sensor is touched.

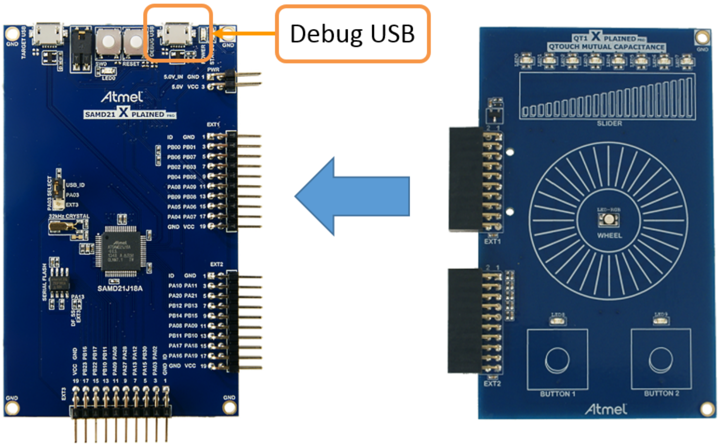
Info: `SENSOR_NUMBER` = 0 as we have only one sensor (one button).

8. Build the solution (F7) and ensure you get no errors 

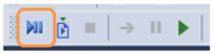
Result: The project to control LED8 using Button 1 is completed and you are ready to program the SAM D21 Xplained Pro board.

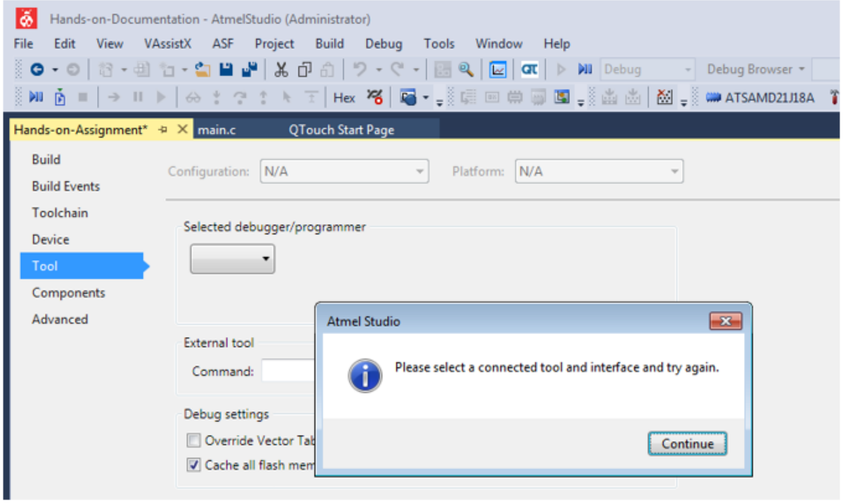
To do: Program the SAM D21 Xplained Pro.

1. Connect the QT1 Xplained Pro Mutual capacitance board to the SAM D21 Xplained Pro board.



2. Connect the SAM D21 Xplained Pro board to your PC using the DEBUG USB connector.

3. Program the application by clicking on the **Start Debugging and Break** icon .
4. You will be asked to select your debug tool.



Configuration: N/A Platform: N/A

Selected debugger/programmer

External tool

Command:

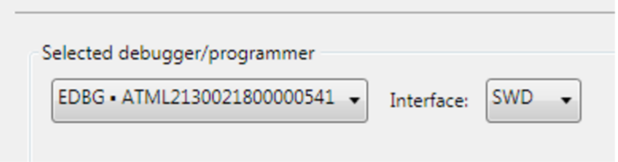
Debug settings

☐ Override Vector Tab

☑ Cache all flash mem

Please select a connected tool and interface and try again.

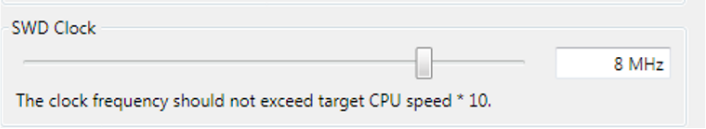
5. Select EDBG and SWD (Serial Wire Debug) as Interface:



Selected debugger/programmer

EDBG • ATML2130021800000541 Interface: SWD

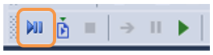
6. Set SWD clock to 8MHz to speed up programming.

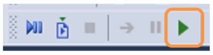


SWD Clock

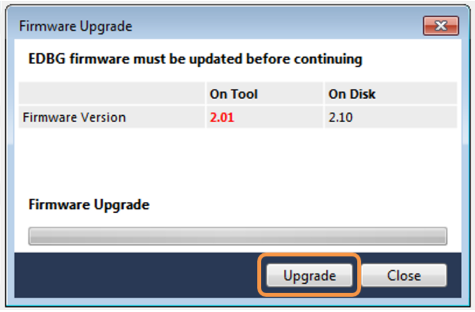
8 MHz

The clock frequency should not exceed target CPU speed * 10.

7. Click again on the **Start Debugging and Break** icon .
8. The application will be programmed in the SAM D21 embedded flash and breaks at main function.

   Click on **Continue** to execute the application .

Info: You may be asked to upgrade your EDBG firmware. If so, click on **Upgrade**.



Firmware Upgrade

EDBG firmware must be updated before continuing

	On Tool	On Disk
Firmware Version	2.01	2.10

Firmware Upgrade

Warning: The upgrade operation may take a few minutes. Wait for the operation to complete.

Result: Touch Button 1 and check that LED8 is lit when touched and not lit when Button 1 is not touched.

4.4. Conclusion

In this assignment, you have learned:

- Which files compose a QTouch application and the main QTouch Library API functions
- How easy it is to implement a Touch application using the QTouch Library

5. Assignment 3: Getting Started with the Atmel QTouch Analyzer

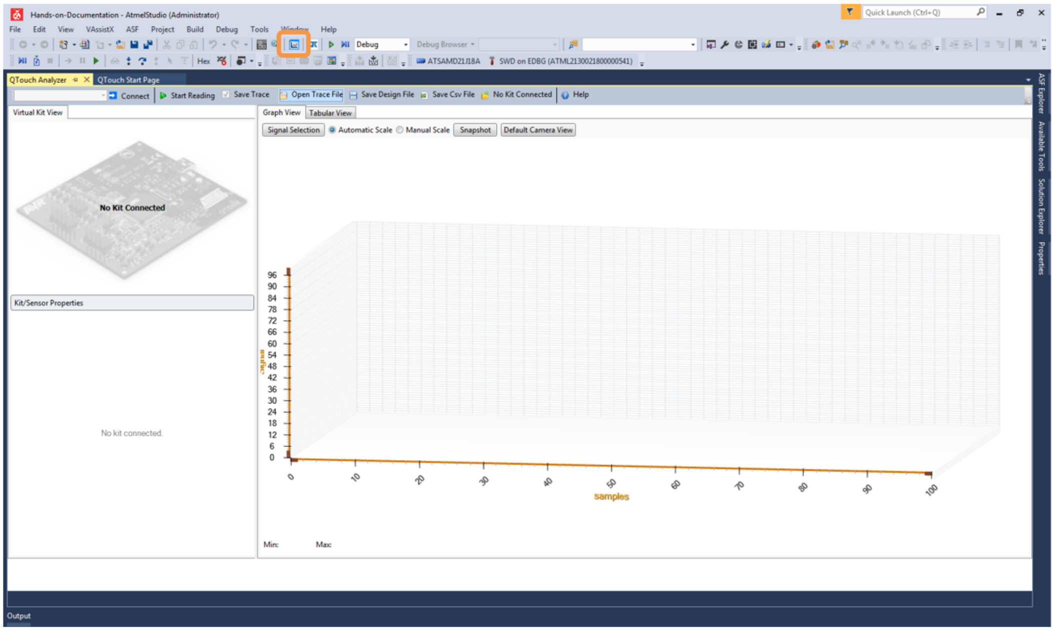
5.1. Atmel QTouch Analyzer Introduction

The QTouch Analyzer which is part of the Atmel QTouch Composer, provides near real-time visualization of the QTouch measurement data sent from the SAM D21 Xplained Pro through the Data Gateway Interface (DGI) of the on-board EDBG device.

It also allows the user to perform run-time tuning of the touch solution through setting of various QTouch library parameters.

The figure below shows the initial window setup for the QTouch Analyzer, showing the following views.

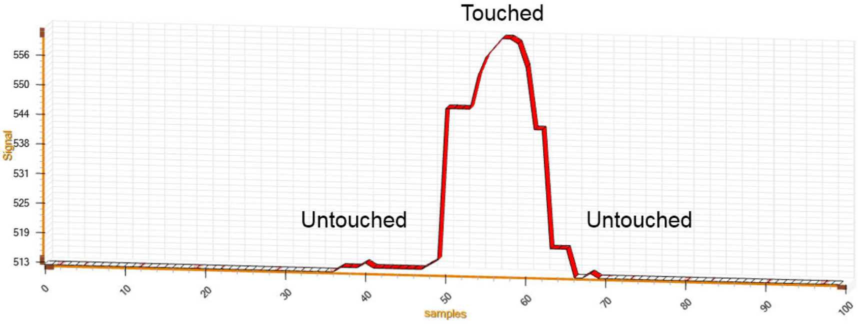
- **Virtual Kit**: this view presents a virtual image of the kit connected and its connection state.
- **Kit/Sensor Properties**: this view allows you to view the kit and sensor configuration options.
- **Trace View**: this view contains a chart showing user-selected data series of touch data.



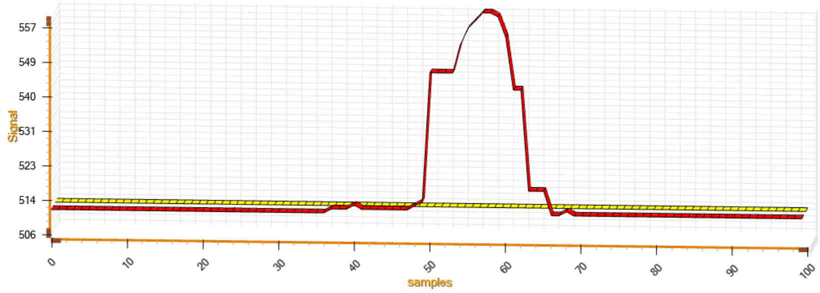
5.2. Basic Touch Sensor Debugging Information

The basic touch sensor debugging information is categorized as follows.

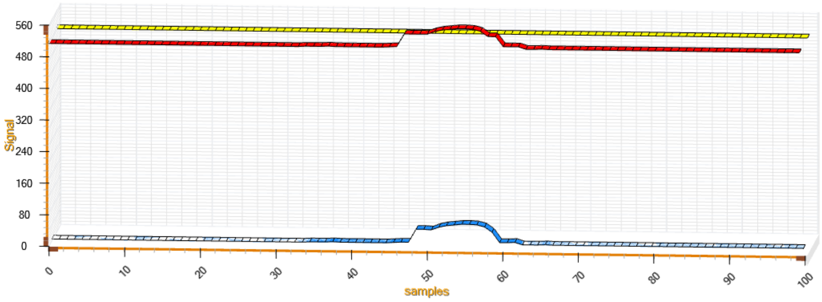
- **Signal**: raw measurement value of a channel, value increases with touch as seen below.



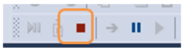
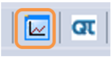
- **Reference**: long term average measurement on a channel (shown in yellow).
  - Also called “**Resting Signal**” when there is no touch.
  - Initial value obtained from calibration process.



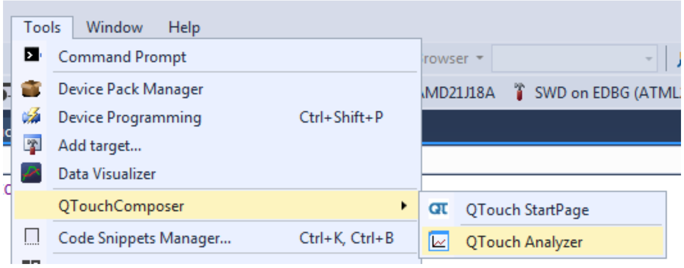
- **(Touch) Delta =** Signal – Reference (shown in blue)
  - Difference between Signal and Reference
  - Deltas increase with touch



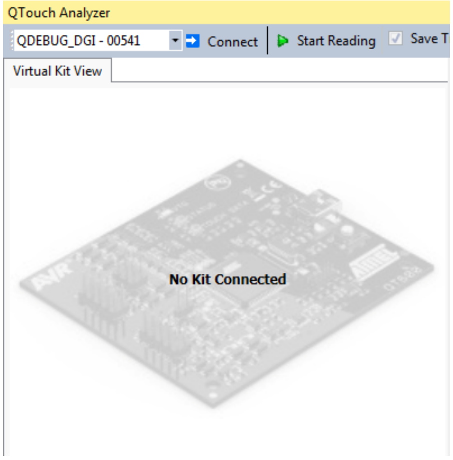
To do: Connect your kit to QTouch Analyzer.

1. Click on **Stop Debugging** .
2. Press the RESET button on the SAM D21 Xplained Pro.
3. Launch QTouch Analyzer by clicking on the **QTouch Analyzer** icon .

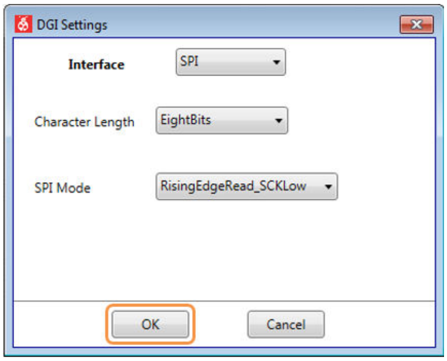
Info: QTouch Analyzer is also accessible through the **Tools** menu.



4. A kit named **QDEBUG_DGI** will be listed in the combo box. Select the kit and click on **Connect**.



5. The DGI settings dialog box will be displayed as shown in the figure. Click **OK**.

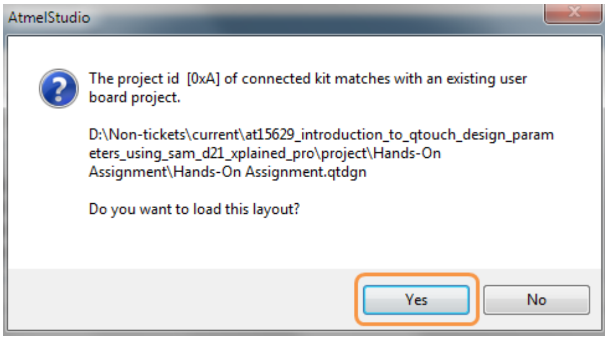

DGI Settings

Interface SPI

Character Length EightBits

SPI Mode RisingEdgeRead_SCKLow

6. The message to load the kit information will pop-up as follows. Click **Yes**.

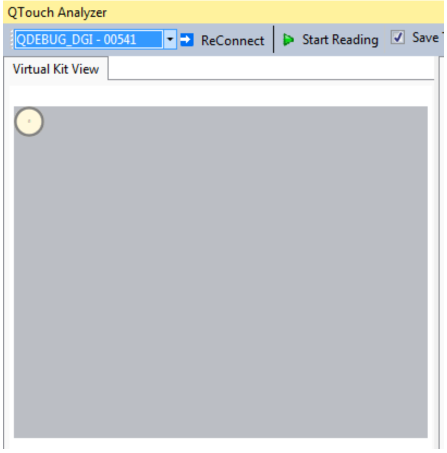

AtmelStudio

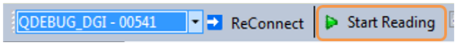
The project id [0xA] of connected kit matches with an existing user board project.

D:\Non-tickets\current\at15629_introduction_to_qtouch_design_parameters_using_sam_d21_xplained_pro\project\Hands-On Assignment\Hands-On Assignment.qtdgn

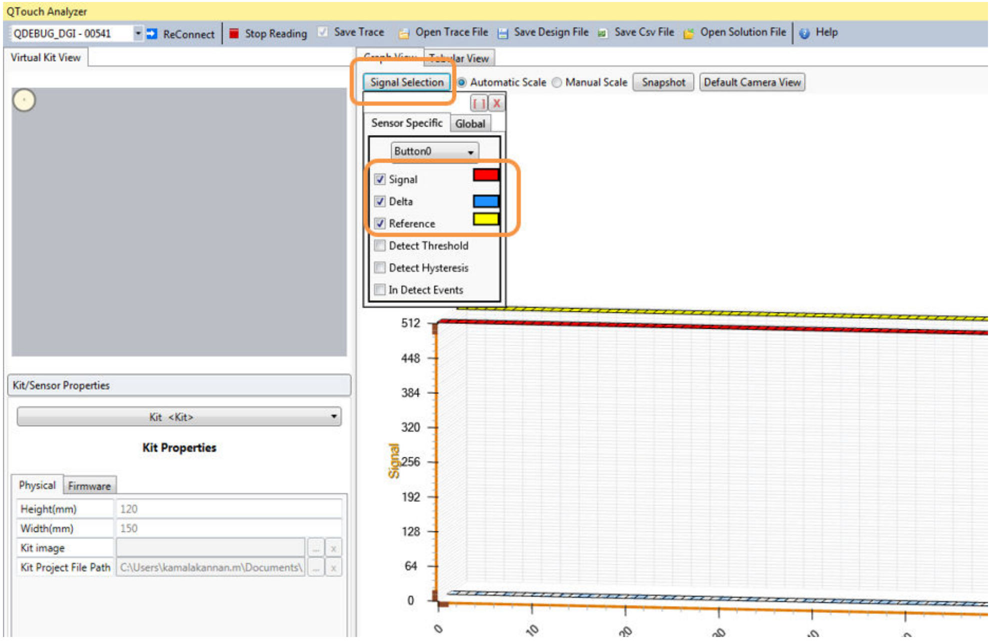
Do you want to load this layout?

7. The Virtual kit will be displayed, which shows it has been correctly connected.



8. Click on **Start Reading**  .

Result: QTouch Analyzer is now connected to your kit and displays both the sensor signal and reference values after selecting **Signal Selection** and tick off **Signal**, **Delta**, and **Reference**.

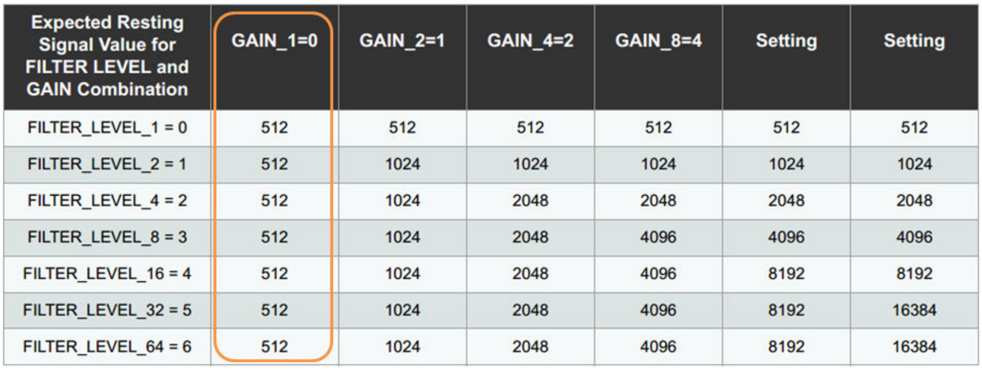


Height(mm)	120
Width(mm)	150
Kit image	
Kit Project File Path	C:\Users\kamalakannan.m\Documents\

To do: Check Reference Value.

The following table extracted from the `QTouch Library Peripheral Touch Controller User Guide` (http://www.atmel.com/Images/Atmel-42195-QTouch-Library-Peripheral-Touch-Controller_User-Guide.pdf), indicates the expected **Resting Signal** value (also called **Reference**) for a given combination of gain setting and filter level setting.

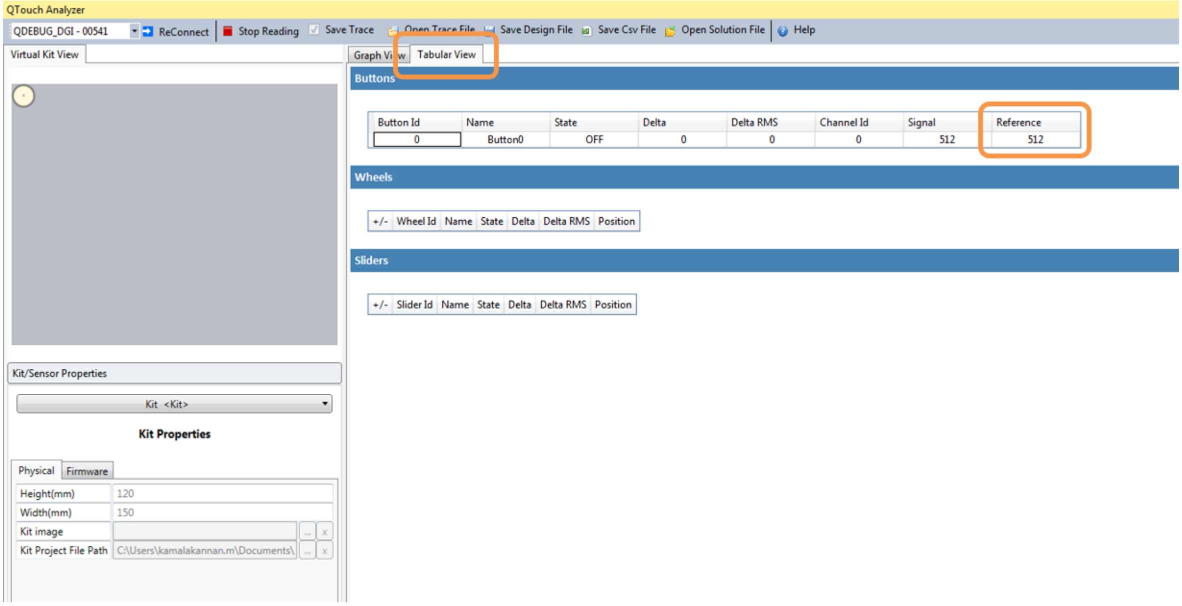
The values provided are only indicative, but the actual sensor signal values will usually be close to the suggested levels.



Expected Resting Signal Value for FILTER LEVEL and GAIN Combination	GAIN_1=0	GAIN_2=1	GAIN_4=2	GAIN_8=4	Setting	Setting
FILTER_LEVEL_1 = 0	512	512	512	512	512	512
FILTER_LEVEL_2 = 1	512	1024	1024	1024	1024	1024
FILTER_LEVEL_4 = 2	512	1024	2048	2048	2048	2048
FILTER_LEVEL_8 = 3	512	1024	2048	4096	4096	4096
FILTER_LEVEL_16 = 4	512	1024	2048	4096	8192	8192
FILTER_LEVEL_32 = 5	512	1024	2048	4096	8192	16384
FILTER_LEVEL_64 = 6	512	1024	2048	4096	8192	16384

Info: The filter level setting controls the number of samples taken to resolve each acquisition. A higher filter level setting provides improved signal to noise ratio under noisy conditions, while increasing the total time for measurement resulting in increased power consumption and response time.

1. Unselect **Signal** and **Delta** in the Trace View to only have **Reference** selected.
2. This can also be checked using the **Tabular View**.



Button Id	Name	State	Delta	Delta RMS	Channel Id	Signal	Reference
0	Button0	OFF	0	0	0	512	512

+/-	Wheel Id	Name	State	Delta	Delta RMS	Position

+/-	Slider Id	Name	State	Delta	Delta RMS	Position

Height(mm)	120
Width(mm)	150
Kit image	
Kit Project File Path	C:\Users\kamalakannan.m\Documents\

3. Check that the Reference signal value is in the same range as in the above table.

Tip: The Gain which has been chosen in the Project Builder is equal to 1, which corresponds to the setting of GAIN_1 = 0. In that case FILTER_LEVEL value has no impact.

Result: The Reference value is coherent with its expected value.

To do: Check touch Delta Value.

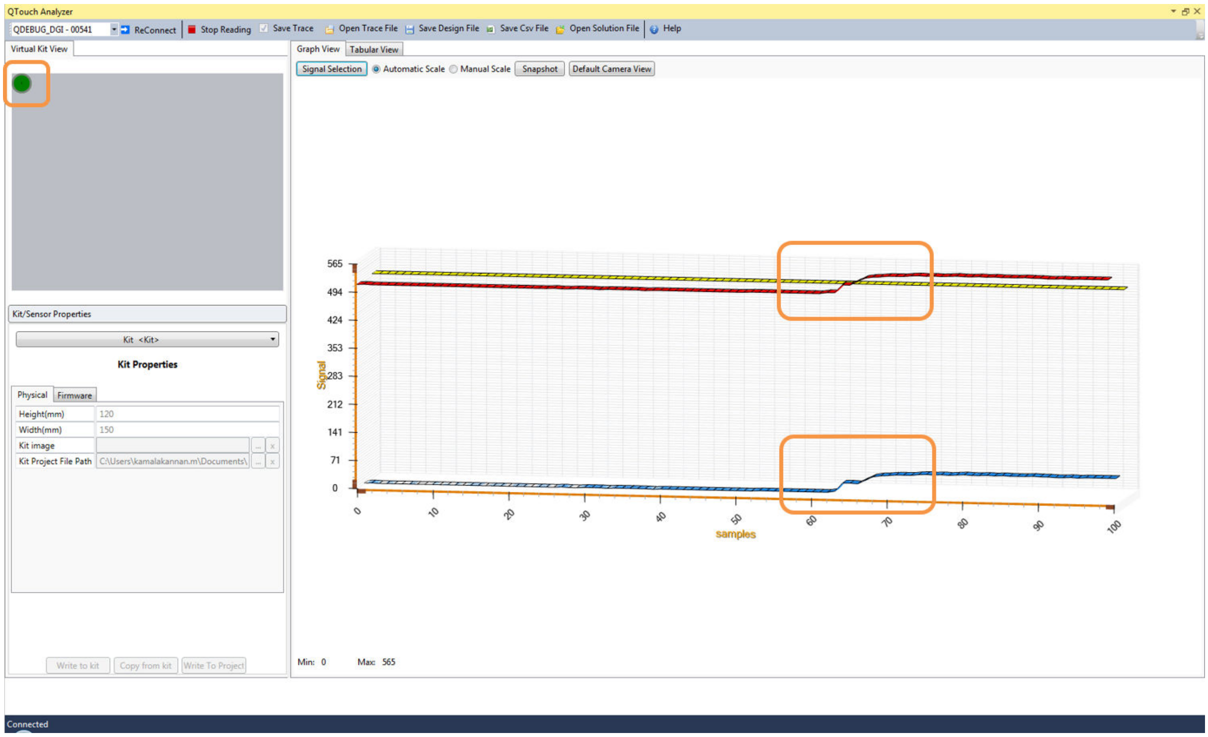
Desired touch delta for buttons is ~30 to 80 counts (or readings) and for wheels or sliders ~50 to 120 counts.

This is a preferred range for non-noisy environment.

For systems with noise, the users should tune delta/threshold/hysteresis and global parameters to avoid false touch. This tuning is dependent on how the signal value variation pattern changes after it is affected by the noise.

In case the touch **Delta** is out of the recommended range, the user has to update the sensor **Gain** so that the touch **Delta** value fits within the expected range.

1. Select Signals, References, and Deltas in the Trace View, or use the **Tabular View**.
2. Press BUTTON 1 on your kit, which is displayed as Button0 on the QTouch Analyzer and check that it becomes green when touched.
3. Check in Trace View that the Delta value has increased with the same amount as the Signal one.



Kit/Sensor Properties

Kit <Kit>

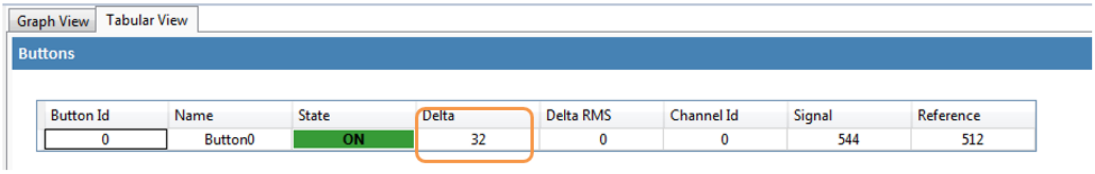
Kit Properties

Height(mm)	120
Width(mm)	150
Kit image	
Kit Project File Path	C:\Users\kamalakannan.m\Documents\

Min: 0 Max: 565

Connected

Tip: You can also easily check signal, reference, and delta values as well as Touch Button state by using the **Tabular View** instead of the Trace View .



Button Id	Name	State	Delta	Delta RMS	Channel Id	Signal	Reference
0	Button0	ON	32	0	0	544	512

4. Check that the touch **Delta** is in the desired range for the non-noisy environment case.

Tip: After you have completed this assignment, click on Stop Reading to prevent the QTouch Analyzer from collecting touch data over a long period as this might slow down the computer .

Result: The touch Delta value is coherent with the expected range.

5.3. Conclusion

In this assignment, you have learned:

- How to get started very quickly with QTouch Analyzer
- How QTouch Analyzer provides near real-time visualization of the QTouch data sent from the Atmel QTouch kits

6. Assignment 4: Understanding Sensor Individual Parameters

After we have covered the Basic Touch Sensor Debugging Information and made a first use of QTouch Analyzer, we will now focus on the important settings that are specific for each individual sensor.

- Buttons, Sliders, and Wheels
  - Detect Threshold
  - Detect Hysteresis
  - Adjacent Key Suppression™ (AKS™)
- Sliders and Wheels
  - Position Resolution
  - Position Hysteresis (Mutual Capacitance technology only)

6.1. Detect Threshold

The **Detect Threshold** parameter of a sensor defines how much its **Signal** must increase above its **Reference** level to be qualified as a potential touch detect.

Larger threshold values desensitize sensors since the signal must change more (i.e. requires larger touch) to exceed the threshold level. Conversely, lower threshold levels make sensors more sensitive.

The threshold setting depends on the amount of signal swing when a sensor is touched. Usually, thicker front panels or smaller electrodes have smaller signal swing on touch, thus requiring lower threshold levels.

To do: In QTouch Analyzer, click on **Start Reading** and approach slowly BUTTON 1. Using the Tabular View, check that a detect touch can occur BEFORE touching the sensor.

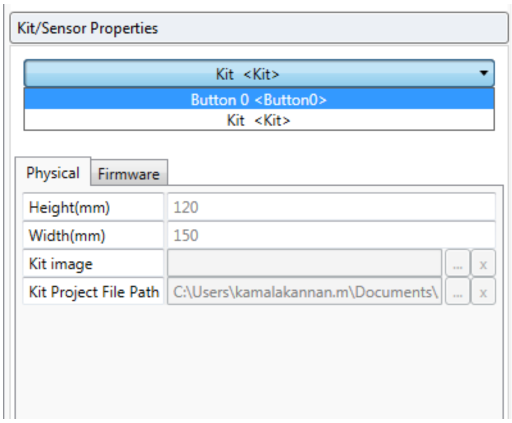
For a regular touch button, this is an undesired proximity effect.

This may be solved either by reducing the **Gain** or increasing the **Detect Threshold** level. Since the gain was set to 1 in previous assignment, we cannot reduce it anymore, so we need to adjust the **Detect Threshold**.

To do: Update Detect Threshold to remove this proximity effect.

To do that, we will use the **Write to Kit** feature of the QTouch Analyzer, which can be used to update the values of the sensor configuration options on the connected kit.

1. In the Kit/Sensor Properties view window, select the **Button 0** Sensor Configuration Options.



Kit/Sensor Properties

Kit <Kit>

Button 0 <Button0>

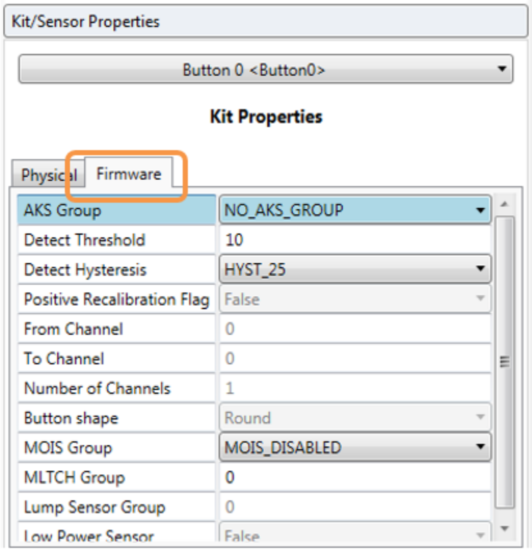
Kit <Kit>

Physical | Firmware

Height(mm)	120
Width(mm)	150
Kit image	
Kit Project File Path	C:\Users\kamalakannan.m\Documents\

Info: The **Kit/Sensor Configuration** view allows you to view the kit and sensor configuration options. After a kit that supports run-time configuration is connected, the kit/ sensor configuration options appear in the kit/sensor configuration view and display the current values of the various kit/sensor configuration options.

2. Select the **Firmware** tab (Sensor Firmware properties).



Kit/Sensor Properties

Button 0 <Button0>

Kit Properties

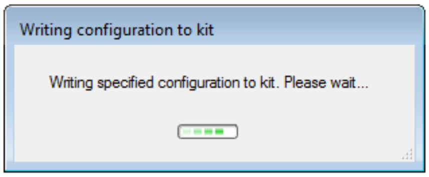
Physical | Firmware

AKS Group	NO_AKS_GROUP
Detect Threshold	10
Detect Hysteresis	HYST_25
Positive Recalibration Flag	False
From Channel	0
To Channel	0
Number of Channels	1
Button shape	Round
MOIS Group	MOIS_DISABLED
MLTCH Group	0
Lump Sensor Group	0
Low Power Sensor	False

3. Update the Detect Threshold.

Tip: For most cases, a touch threshold which is 50% of the observed touch delta is sufficient.

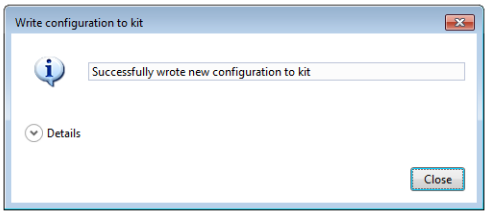
4. Click on **Write to Kit** to update this value directly on the kit.



Writing configuration to kit

Writing specified configuration to kit. Please wait...

5. You will get the following message confirming that the operation succeeded.

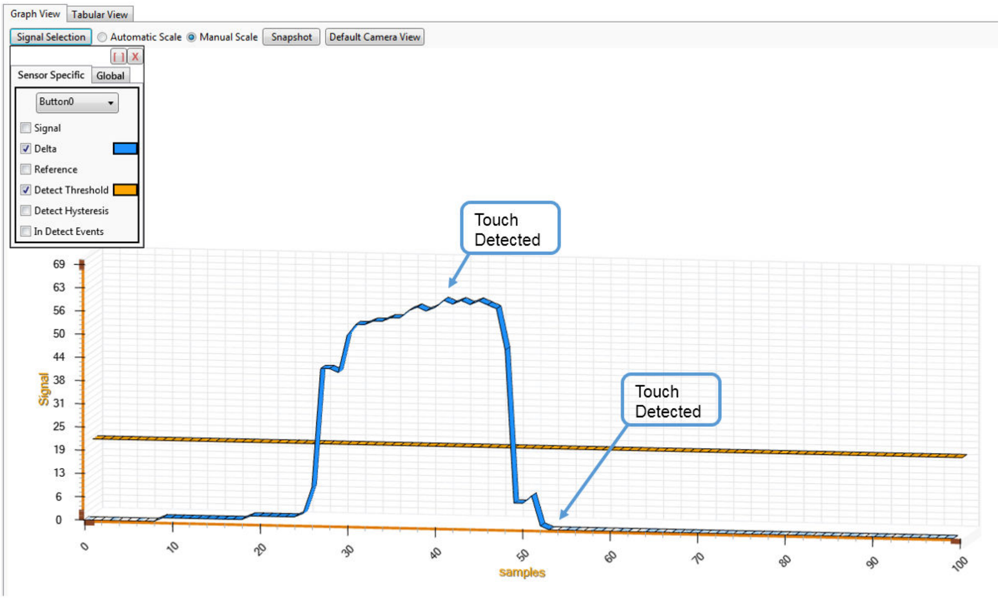


Write configuration to kit

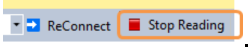
Successfully wrote new configuration to kit

6. Now you can test by approaching slowly BUTTON 1 that the proximity effect has disappeared.

Info: You can also check the Detect Threshold parameter using the Trace View by selecting the Detect Threshold and Deltas as an example.



Tip: After you have completed this assignment, click on **Stop Reading** to prevent the QTouch Analyzer from collecting touch data over a long period as this might slow down the computer



.

Result: We have understood the properties of the **Detect Threshold** parameter. This Sensor parameter can be used to play with the sensor sensitivity depending on the customer application constraints and his hardware sensor design.

6.2. Detect Hysteresis

This setting is the sensor detection hysteresis value.

It is expressed as a percentage of the sensor detection threshold setting.

After a sensor goes into detect its threshold level is reduced (by the hysteresis value) in order to avoid the sensor dither in and out of detect if the signal level is close to original threshold level.

- Setting of 0 = 50% of detect threshold value (HYST_50)
- Setting of 1 = 25% of detect threshold value (HYST_25)
- Setting of 2 = 12.5% of detect threshold value (HYST_12_5)
- Setting of 3 = 6.25% of detect threshold value (HYST_6_25)

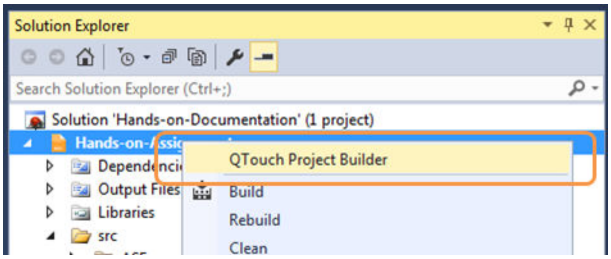
So Detect Hysteresis has only an effect when a touch delta is varying around the threshold.

Info: With a properly tuned system and in the absence of noise, there is no apparent difference in operation between 6.25% and 50%.

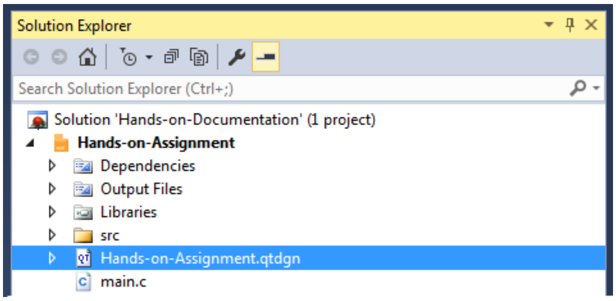
To demonstrate this feature, the idea will be to consider a proximity sensor where we will show that the sensor turns ON by moving the finger closer but it doesn't turn OFF until the finger moves away more than a specific distance.

To do: Update Sensor Button 1 Parameters to add a proximity effect.

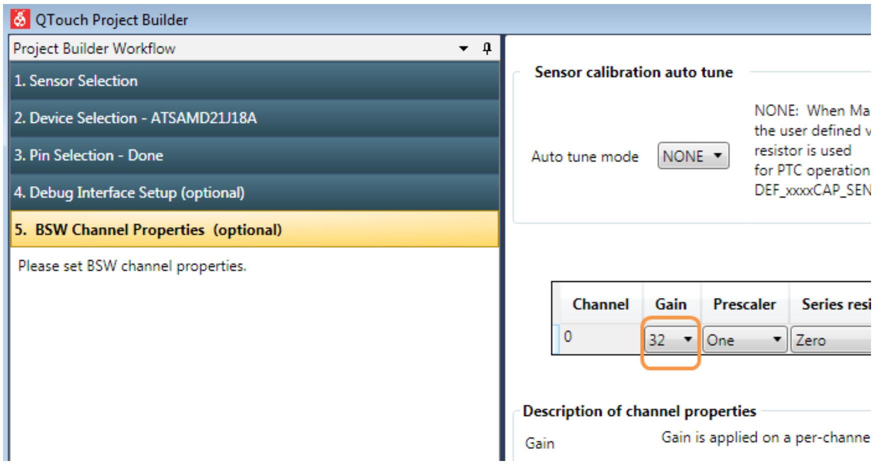
1. Click on **Solution Explorer**.
2. Launch QTouch Project Builder by right clicking on the **Hands-on Assignment** project then select **QTouch Project Builder**.



Tip: You can also directly double click on the `Hands-on Assignment.qtdgn` file.


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Hands-on-Documentation' (1 project)
Hands-on-Assignment
Dependencies
Output Files
Libraries
src
Hands-on-Assignment.qtdgn
main.c

3. Update **Gain** to 32 on the **BSW Channel Properties** page.


QTouch Project Builder
Project Builder Workflow
1. Sensor Selection
2. Device Selection - ATSAMD21J18A
3. Pin Selection - Done
4. Debug Interface Setup (optional)
5. BSW Channel Properties (optional)
Sensor calibration auto tune
Description of channel properties

Please set BSW channel properties.

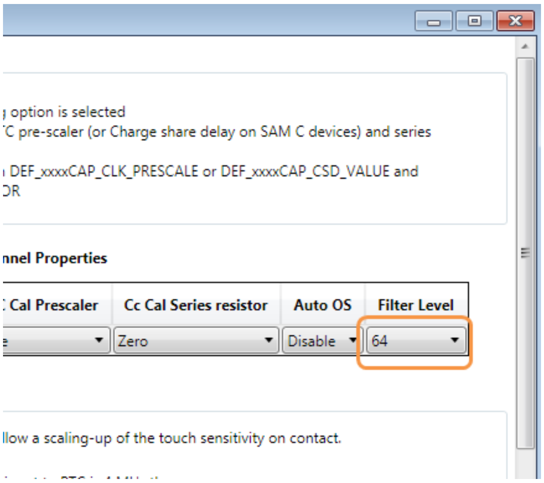
Auto tune mode NONE

NONE: When Ma
the user defined v
resistor is used
for PTC operation
DEF_xxxxCAP_SEN

Channel	Gain	Prescaler	Series resi
0	32	One	Zero

Gain — Gain is applied on a per-channe

4. Update **Filter Level** to 64 in the **BSW Channel Properties** page. Click **Apply**.



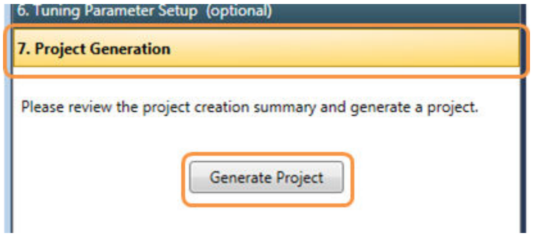
; option is selected
'C pre-scaler (or Charge share delay on SAM C devices) and series

ı DEF_xxxxCAP_CLK_PRESCALE or DEF_xxxxCAP_CSD_VALUE and
)R

: Cal Prescaler	Cc Cal Series resistor	Auto OS	Filter Level
e	Zero	Disable	64

llow a scaling-up of the touch sensitivity on contact.

5. Click **Generate Project** on the **Project Generation** page to update your QTouch Project.



6. Tuning Parameter Setup (optional)

7. Project Generation

Please review the project creation summary and generate a project.

Info: By increasing both the Gain and the Filter Level, we will be able to increase the Reference value, and so scaling up the Touch Delta value.

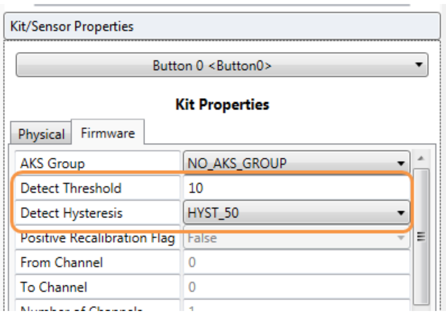
6. Build the solution and ensure you get no errors .

7. Program the application by clicking on the Start Without Debugging icon .

Result: Your QTouch project is now updated and programmed on the SAM D21 Xplained Pro.

To do: Update Detect Hysteresis to demonstrate the Hysteresis effect.

1. Press **RESET** button on the SAM D21 Xplained Pro.
2. In QTouch Analyzer, click on **Connect** or **ReConnect** then **Start Reading**.
3. In the Kit/Sensor Properties view, select the **Button 0** Sensor Configuration Options.
4. Select Firmware tab and update.
   – Detect Threshold to 10
   – Detect Hysteresis to HYST_50



Kit/Sensor Properties

Button 0 <Button0>

Kit Properties

AKS Group	NO_AKS_GROUP
Detect Threshold	10
Detect Hysteresis	HYST_50
Positive Recalibration Flag	False
From Channel	0
To Channel	0
Number of Channels	1

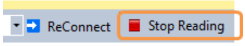
Info: Using a very low Detect Threshold value and 50% of that value for the Detect Hysteresis parameter will give the highest hysteresis value, and so demonstrate the hysteresis effect more easily.

5. Click on **Write to Kit** to update that new value directly on the kit.
6. Test that the proximity effect has increased a lot by slowly approaching BUTTON 1.

Tip: You can look directly at the LED8 or use the Tabular View Sensor State.

7. Now test that BUTTON 1 doesn't turn OFF even if your finger moves below the Detect threshold level, before it's outside the hysteresis.

Tip: After you have completed this assignment, click on Stop Reading to prevent QTouch Analyzer from collecting touch data over a long period as this might slow down the computer



.

Result: We have understood the purpose of the **Detect Hysteresis** parameter. This Sensor parameter can be used to avoid the sensor dither in and out of detect if the signal level is close to original threshold level. As discussed previously, tuning this parameter is important when noise is present in the system.

6.3. Adjacent Key Suppression (AKS)

In designs where the sensors are close together or configured for high sensitivity, multiple sensors might report a detection simultaneously.

To allow applications to determine the intended single touch, the touch library provides the user the ability to configure a certain number of sensors in an Adjacent Key Suppression (AKS) group.

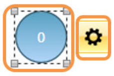
When a group of sensors are in the same AKS group, only the first sensor which has strongest touch delta will report detection. The sensor reporting detection will continue to report detection even if another sensor's delta becomes stronger.

The sensor stays detected until its delta falls below its detection threshold. If more sensors in the AKS group are still detected only the strongest will report detection.

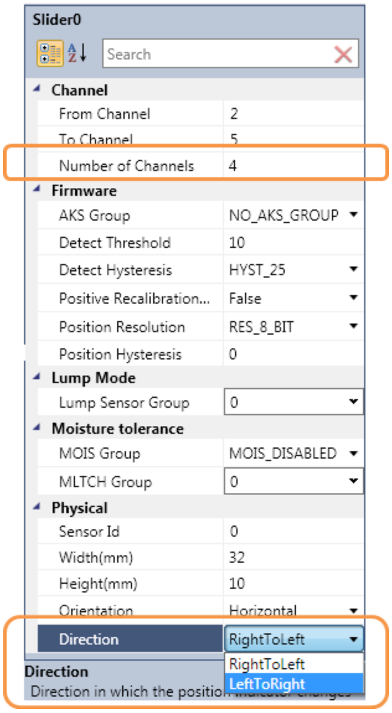
Info: At a given time point, only one sensor from each AKS group is reported to be detected.

To demonstrate this feature, the idea will be to add two other sensors, one button and one slider, and then associate one AKS group to the two buttons. We will then be able to check that only one button can be reported as a Touch whereas we could use the slider in parallel.

To do: Add Sensor BUTTON 2 and the SLIDER.

1. Launch QTouch Project Builder by right clicking on the Hands-on Assignment Project.
2. Update the Detect Threshold of the Button 0 (BUTTON 1 on the QT1 Xplained Pro) to remove any proximity effect (value retrieved from Detect Threshold assignment) by clicking the sensor and selecting the cogwheel icon (threshold:23, hysteresis: HYST_25)  .

3. Add a new button in the **Sensor Selection** page by clicking on the **Button** icon, click **Add**.
4. Click the sensor and Update the Detect Threshold of the Button 1 (BUTTON 2 on the QT1 Xplained Pro) to remove any proximity effect, same as explained above.
5. Add a slider in the **Sensor Selection** page by clicking on the **Slider** icon, click **Add**.
   – Click the sensor and select the cogwheel, and change the **Number of Channels** to 4 in **Channel** tab. Also, change the direction to **LeftToRight** in the **Physical** tab to get correct visualization in QTouch Analyzer (do not update the other parameters at this stage).



Slider0

Search

Channel	
From Channel	2
To Channel	5
Number of Channels	4
Firmware	
AKS Group	NO_AKS_GROUP
Detect Threshold	10
Detect Hysteresis	HYST_25
Positive Recalibration...	False
Position Resolution	RES_8_BIT
Position Hysteresis	0
Lump Mode	
Lump Sensor Group	0
Moisture tolerance	
MOIS Group	MOIS_DISABLED
MLTCH Group	0
Physical	
Sensor Id	0
Width(mm)	32
Height(mm)	10
Orientation	Horizontal
Direction	RightToLeft

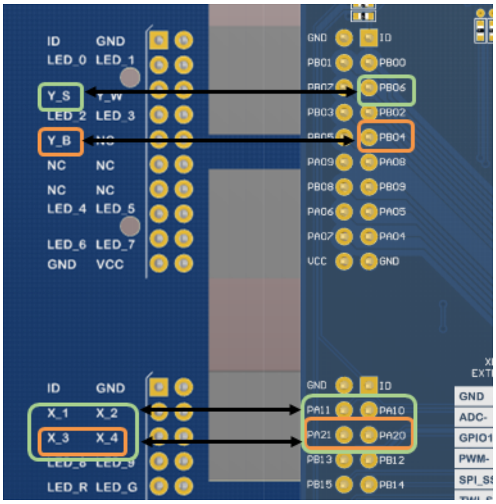
Direction
Direction in which the positio

6. Update X-Y lines and Gain for Buttons 0, 1, and the Slider.

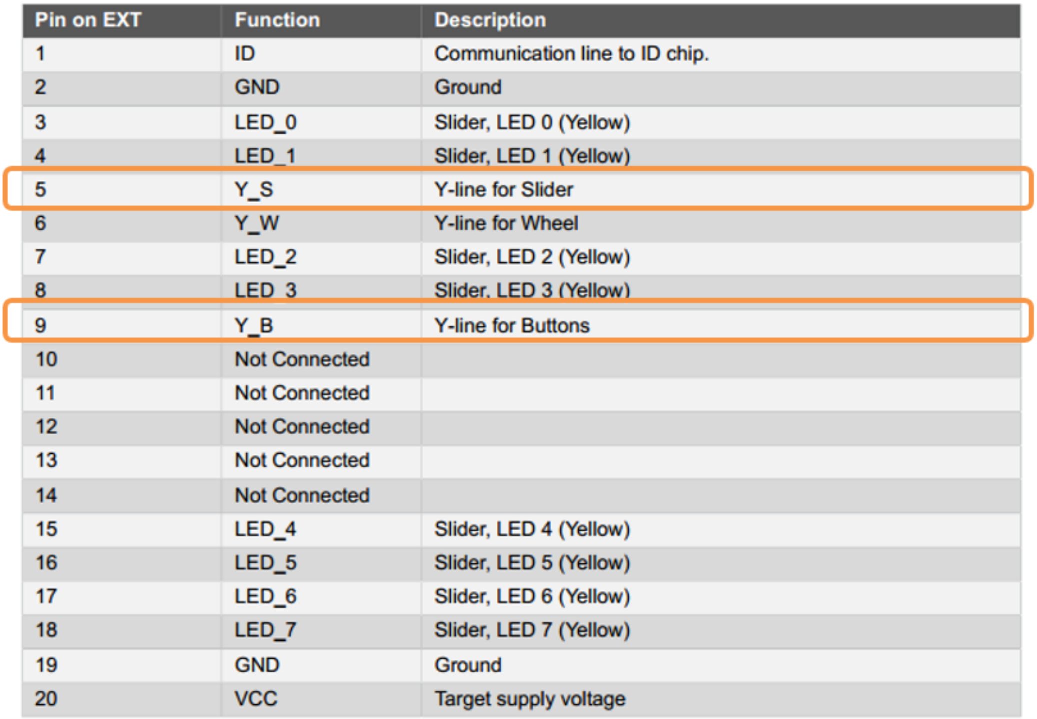
Info: For sensors consisting of several channels, wheels, and sliders, all channels must use the same Y-Line.

Warning: The X-line and Y-line numbering in the QT1 Xplained Pro User Guide does not correspond to the X-line and Y-line numbering in the SAM D21 PTC module as the QT1 Xplained Pro is a generic touch board (not dedicated to the SAM D21).

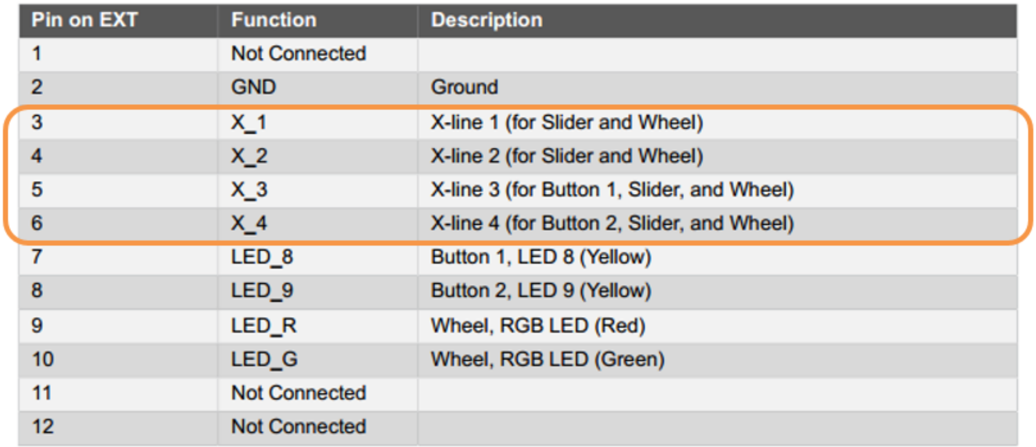
7. Flip the SAM D21 Xplained Pro and QT1 Xplained Pro boards and determine the SAM D21 I/O pins that are connected to Buttons 1, 2, and the Slider.



Info: X-line and Y-line for Buttons 1, 2, and Slider can be retrieved from the Atmel QT1 Xplained Pro User Guide (http://www.atmel.com/Images/Atmel-42193-QT1-Xplained-Pro_User-Guide.pdf).

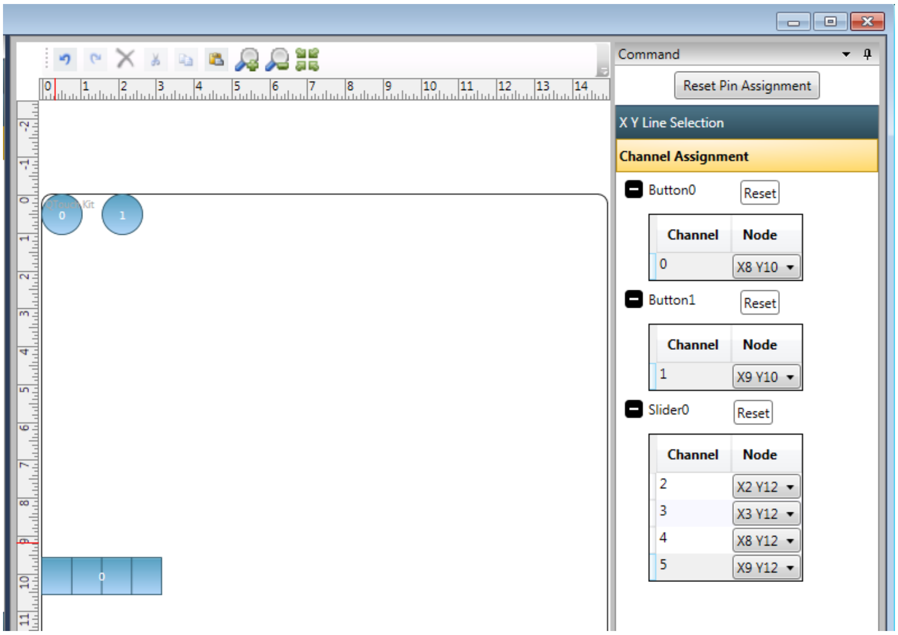


Pin on EXT	Function	Description
1	ID	Communication line to ID chip.
2	GND	Ground
3	LED_0	Slider, LED 0 (Yellow)
4	LED_1	Slider, LED 1 (Yellow)
5	Y_S	Y-line for Slider
6	Y_W	Y-line for Wheel
7	LED_2	Slider, LED 2 (Yellow)
8	LED_3	Slider, LED 3 (Yellow)
9	Y_B	Y-line for Buttons
10	Not Connected	
11	Not Connected	
12	Not Connected	
13	Not Connected	
14	Not Connected	
15	LED_4	Slider, LED 4 (Yellow)
16	LED_5	Slider, LED 5 (Yellow)
17	LED_6	Slider, LED 6 (Yellow)
18	LED_7	Slider, LED 7 (Yellow)
19	GND	Ground
20	VCC	Target supply voltage



Pin on EXT	Function	Description
1	Not Connected	
2	GND	Ground
3	X_1	X-line 1 (for Slider and Wheel)
4	X_2	X-line 2 (for Slider and Wheel)
5	X_3	X-line 3 (for Button 1, Slider, and Wheel)
6	X_4	X-line 4 (for Button 2, Slider, and Wheel)
7	LED_8	Button 1, LED 8 (Yellow)
8	LED_9	Button 2, LED 9 (Yellow)
9	LED_R	Wheel, RGB LED (Red)
10	LED_G	Wheel, RGB LED (Green)
11	Not Connected	
12	Not Connected	

8. Use the **Pin Selection** page from QTouch Project Builder to select the correct X- and Y-lines.
9. You should get the following page.



Button0 Reset

Channel	Node
0	X8 Y10

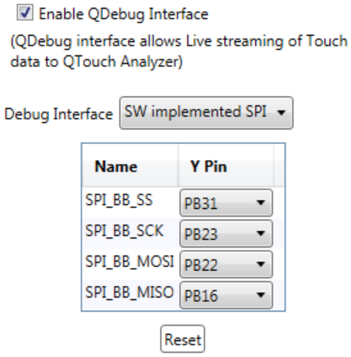
Button1 Reset

Channel	Node
1	X9 Y10

Slider0 Reset

Channel	Node
2	X2 Y12
3	X3 Y12
4	X8 Y12
5	X9 Y12

10. Re-enable the QDebug Interface and re-assign the right port(s) and pins to the SPI interface in the **Debug Interface Setup (optional)** page.



☑ Enable QDebug Interface
(QDebug interface allows Live streaming of Touch data to QTouch Analyzer)

Debug Interface: SW implemented SPI

Name	Y Pin
SPI_BB_SS	PB31
SPI_BB_SCK	PB23
SPI_BB_MOSI	PB22
SPI_BB_MISO	PB16

11. Update **Gain** to **1** and **Filter Level** (under **Tuning Parameters**) to **16,** for all sensors. Click **Apply**.
12. Click on the **Project Generation** page and click **Generate Project** to update your project.

Result: You have added a new button and the slider to your QTouch project.

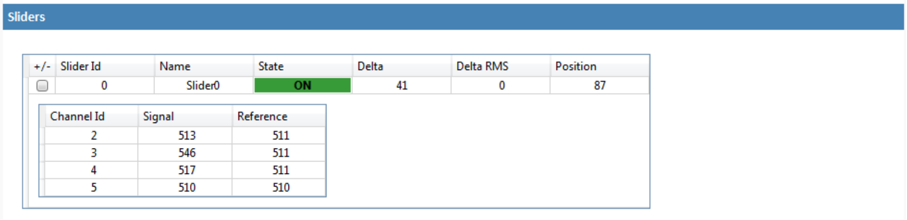
To do: Check Slider Touch Delta Value.

As previously covered, the desired touch delta for buttons is ~30 to 80 counts (or readings) and for wheels or sliders ~50 to 120 counts.

This is a preferred range for non-noisy environment.

In case the touch **Delta** is out of this range, the user has to update the sensor **Gain** so that the touch **Delta** value fits within the expected range.

1. Build the solution and program it by clicking on the **Start Without Debugging** icon ▶ .
2. Press the RESET button on the SAM D21 Xplained Pro.
3. In the QTouch Analyzer, click on **Connect**/**ReConnect** and then **Start Reading**.
4. Press the Slider on your kit and check its delta value using the Tabular View.



Sliders

+/-	Slider Id	Name	State	Delta	Delta RMS	Position
□	0	Slider0	ON	41	0	87

Channel Id	Signal	Reference
2	513	511
3	546	511
4	517	511
5	510	510

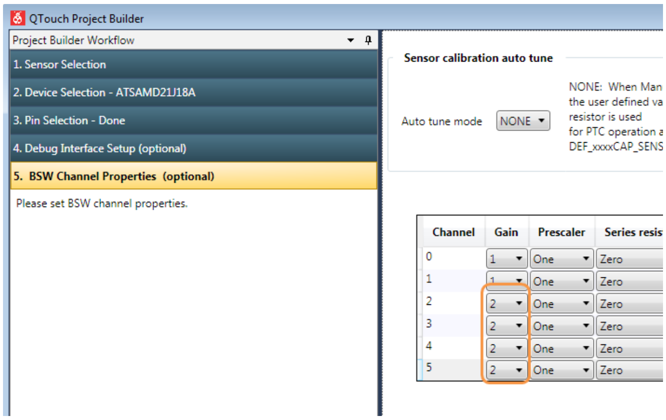
Result: Depending on the position of the slider, you should observe delta values in the range of 20 to 60 counts, which is lower than recommended. You will then need to increase the gain so that it fits with the expected range. Before continuing make sure to press the **Stop Reading** button.

To do: Update the Slider Gain by using the QTouch Project Builder.

Info: **Gain setting** is applied on a per-channel basis to allow superior sensitivity upon contact. Recommended gain settings depend on the sensor design and touch panel thickness.

Tip: The increase in the gain value is usually required by a thick front panel but also depends on the sensor design (size of the sensor and coupling between X and Y for mutual capacitance sensors). Thicker front panels (3mm and more) can result in reduced sensitivity upon touch. Likewise, smaller size sensors or sensors surrounded by ground layer or sensors with ground on the bottom side typically require an increased gain setting.

1. Using QTouch Project Builder, update the **SliderGain** from **1** to **2**.



QTouch Project Builder
Project Builder Workflow
1. Sensor Selection
2. Device Selection - ATSAMD21J18A
3. Pin Selection - Done
4. Debug Interface Setup (optional)
5. BSW Channel Properties (optional)

Please set BSW channel properties.

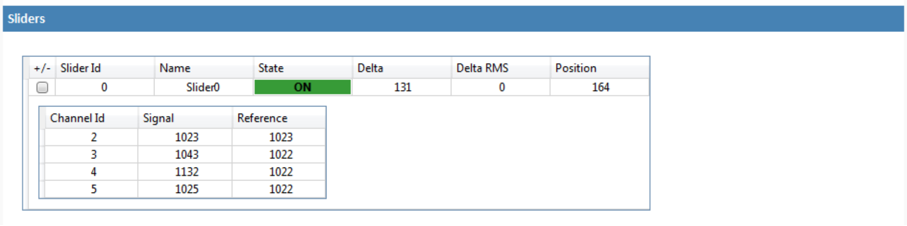
Sensor calibration auto tune

Auto tune mode: NONE

NONE: When Man
the user defined va
resistor is used
for PTC operation a
DEF_xxxxCAP_SENS

Channel	Gain	Prescaler	Series resis
0	1	One	Zero
1	1	One	Zero
2	2	One	Zero
3	2	One	Zero
4	2	One	Zero
5	2	One	Zero

2. Generate the project, build the solution, and program it by clicking on the **Start Without Debugging** icon .
3. Press the RESET button on the SAM D21 Xplained Pro.
4. In QTouch Analyzer, click on the **Connect/ReConnect** and then **Start Reading**.
5. Press the Slider on your kit and check its delta value using Tabular View.



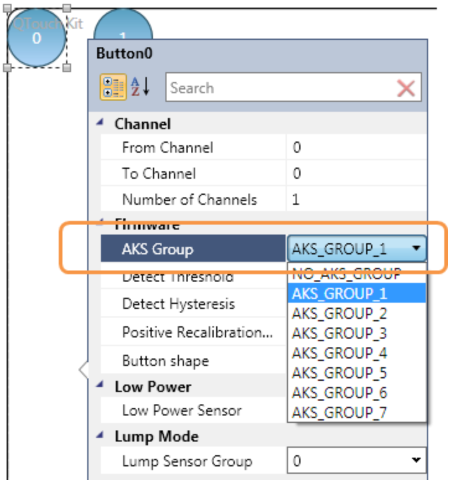
+/-	Slider Id	Name	State	Delta	Delta RMS	Position
□	0	Slider0	ON	131	0	164

Channel Id	Signal	Reference
2	1023	1023
3	1043	1022
4	1132	1022
5	1025	1022

Result: The touch Delta value is now coherent with the expected range.

To do: Update the AKS Group of Buttons 0 and 1 by using the QTouch Analyzer.

1. Update the AKS Group of the Button 0 (BUTTON 1 on the QT1 Xplained Pro) to AKS_GROUP_1 by clicking the sensor and selecting the cogwheel icon .

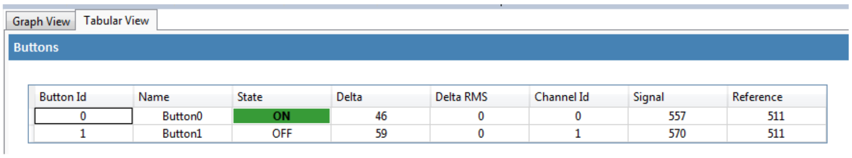


Button0	
Channel	
From Channel	0
To Channel	0
Number of Channels	1
Firmware	
AKS Group	AKS_GROUP_1
Detect Threshold	
Detect Hysteresis	
Positive Recalibration...	
Button shape	
Low Power	
Low Power Sensor	
Lump Mode	
Lump Sensor Group	0

2. Update Button 1 AKS Group with the same Group as Button 0.
3. Generate the project, build the solution, and program it by clicking on the **Start Without Debugging** icon .
4. Press both buttons and check **using Tabular view** that only one sensor is reported to be detected.

Info: On the other hand, you can also check that it is possible to use the Slider in parallel of the Buttons as we did not assign any AKS Group to it (or its AKS Group is different from the Buttons one).

Info: This example shows that only a touch on Button 0 (BUTTON 1 on the kit) will be reported as detected as it's the first one to reach its Detect Threshold.



Button Id	Name	State	Delta	Delta RMS	Channel Id	Signal	Reference
0	Button0	ON	46	0	0	557	511
1	Button1	OFF	59	0	1	570	511

Result: We have understood the purpose of the **AKS** parameter.
When a group of sensors are in the same AKS group, only the first strongest sensor will report detection.

Using this parameter is important in designs where the sensors are close together or configured for high sensitivity as it will avoid multiple sensors to report a detect simultaneously.

6.4. Position Resolution

The rotor or slider needs the **Position Resolution** (angle resolution in case of rotor and linear resolution in case of slider) to be set.

Position Resolution is the number of bits needed to report the position of rotor or slider. It can have values from 2 bits to 8 bits.

Position Resolution is usually set to 8 bits even if the majority of the applications will never need such accuracy.

Info: For a typical 40mm slider, 8-bit resolution means six positions per millimeter.

An application will generally act on a status change: touch contact applied or removed or moved along the slider.

- If we have high resolution position calculation then we report many changes during a slide of only a few mm. The application code will read the position each time, compare it to the previous position and decide if any action is required.
- Using lower resolution means the application code is entering this decision making process only when a change occurs, which is significant in the application context

Result: Using a high resolution does not represent an issue with our QTouch Library as it only needs a few instructions to read a new position. So no differences in term of performance or power consumption will be seen.

6.5. Position Hysteresis

In case of Mutual Capacitance, the rotor or slider needs the position hysteresis (angle hysteresis in case of rotor and linear hysteresis in case of slider) to be set.

It is the number of positions the user has to move back, before touch position is reported when the direction of scrolling is changed and during the first scrolling after user press.

✓ **Result:** Position Hysteresis may be demonstrated in a noisy environment by setting its value to 0, which will show flickering on the slider position. Setting its value to a higher one will remove flickering.

6.6. Conclusion

In this assignment, you have learned:

- How to easily tune your QTouch application using QTouch Analyzer thanks to the Write to Kit feature
- The different characteristics of the Sensor Individual Parameters and how to visualize them on QTouch Analyzer

7. Summary

This hands-on demonstrated the ease of use of the different Atmel QTouch technology tools by presenting the different steps a user need to follow to successfully build his first QTouch application.

The following topics have been covered:

- Atmel QTouch Composer
- Atmel QTouch Library API
- Atmel QTouch Analyzer
- The Sensor Individual Parameters

You have seen how the embedded peripheral touch controller (PTC) of the Atmel | SMART SAM D21 MCU makes it easy to add capacitive touch sensing to your project with buttons, sliders, wheels, and proximity.

8. Revision History

Doc. Rev.	Date	Comments
42271D	03/2016	Updated training for Composer and Library version 5.9
42271C	05/2015	Updated training for Composer and Library version 5.6
42271B	07/2014	Updated review comments from Marcom
42271A	05/2014	Initial document release

Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**

DISCLAIMER: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

SAFETY-CRITICAL, MILITARY, AND AUTOMOTIVE APPLICATIONS DISCLAIMER: Atmel products are not designed for and will not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death (“Safety-Critical Applications”) without an Atmel officer's specific written consent. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Atmel products are not designed nor intended for use in military or aerospace applications or environments unless specifically designated by Atmel as military-grade. Atmel products are not designed nor intended for use in automotive applications unless specifically designated by Atmel as automotive-grade.