UM1913
User manual

Developing applications on STM32Cube with STMTouch® touch sensing library

Introduction

STMCube™ initiative was originated by STMicroelectronics to ease developers’ life by reducing development efforts, time and cost. STM32Cube covers STM32 portfolio.

STM32Cube Version 1.x includes:

- The STM32CubeMX, a graphical software configuration tool that allows to generate C initialization code using graphical wizards.
- A comprehensive embedded software platform, delivered per series (such as STM32CubeF0 for STM32F0 series)
  - The STM32Cube HAL, an STM32 abstraction layer embedded software, ensuring maximized portability across STM32 portfolio
  - A consistent set of middleware components such as RTOS, USB, TCP/IP, Graphics, STMTouch®.
  - All embedded software utilities coming with a full set of examples.

This user manual describes the STMTouch® touch sensing library which is part of the STM32Cube firmware package that is available as a free download from the ST website (http://www.st.com/stm32cube). It is intended for developers who use STM32Cube firmware on STM32 microcontrollers from the STM32F0, STM32F3, STM32L0, STM32L1 or STM32L4 series. It describes how to start and implement a touch sensing application.

The STMTouch® touch sensing library includes:

- A complete register address mapping with all bits, bitfields and registers declared in C. This avoids a cumbersome task and more importantly, it brings the benefits of a bug free reference mapping file, speeding up the early project phase.
- A collection of routines and data structures covering all functions to manage the touch sensing technology.

The source code is developed using the ANSI-C standard. It is fully documented and is MISRA®-C 2004 compliant. Writing the whole library in 'Strict ANSI-C' makes it independent from the development tools. Only the start-up files depend on the development tools.

Since this library is generic and covers many functionalities and microcontrollers, the size and/or execution speed of the application code may not be optimized. For many applications, this library may be used as is. However, for applications having tough constraints in terms of code size and/or execution speed, this library may need to be fine tuned.

Contents

List of tables

List of figures

1 Coding rules and conventions

1.1 Acronyms and abbreviations

The table below summarizes all acronyms and abbreviations used inside this user manual.

Table 1. Terms and Acronyms

Name	Definition
Bank	A group of channels acquired simultaneously
Channel	Elementary acquisition item
Cs	Charge-Transfer sampling capacitor or capacitance
Ct	Equivalent touch capacitance
CT	Charge-Transfer acquisition principle
Cx	Equivalent sensor capacitance
Delta	Difference between the Measure and the Reference
DTO	Detection Time Out
DXS	Detection Exclusion System
ECS	Environment Change System
Linear sensor	Multi-channels sensor with the electrodes positioned in a linear way
LinRot sensor	A linear or rotary touch sensor
Measure or Meas	Current signal measured on a channel
Reference or Ref	Reference signal initialized during calibration and then regularly updated by the ECS
Rotary sensor	Multi-channels sensor with the electrodes positioned in a circular way
Rs	ESD protection serial resistor
Sensor or Object	Any touch sensor (touchkey, linear, rotary,...)
Timer acquisition mode	Acquisition using two timers and PWM signals (also called hardware acquisition mode). Only available on STM32L1 series microcontrollers
Touchkey or TKey sensor	Single channel sensor
TSC	Touch sensing controller peripheral

1.2 Naming conventions

The following naming conventions are used in the STMTouch touch sensing library source files:

- Source and header files are in lower-case and preceded by 'tsl' or 'tsl_'.
- The microcontroller family is added at the end of the file name if needed.
- Functions, globals, typedefs and defines are preceded by 'TSL'.
- Constants are written in upper case and preceded by 'TSLPRM_'.
- Constants used in one file are defined within this file only.
- Constants used in more than one file are defined in a header file.
- Typedef names are suffixed with '_T'.
- Enum typedefs are suffixed with '_enum_T'.
- Functions are named according to the 'TSL_[module]_[function]' scheme.
  - [module]: abbreviation of the file (acq, tim, dxs, etc.).
  - [function]: the first letter in each word is in upper case.

1.3 Coding rules

This section describes the coding rules used in the STMTouch touch sensing library source files.

1.3.1 General

- Source code complies with ANSI C standard.
- No warning after compilation. Any warning that cannot be eliminated is commented in the source code.
- ANSI standard data types are used and defined in the ANSI C header file <stdint.h>.
- No blocking code is present and all required waiting loops (polling loops) are controlled by a timeout.

1.3.2 Variable types

Specific variable types are already defined with a fixed type and size.

- The types that are used by all modules are defined in the **tsl_types.h** file.
- Other variable types are defined in their corresponding module header file.

1.3.3 Peripheral registers

The peripheral registers are accessed using the pointers described in the CMSIS device peripheral access layer header file.

1.4 MISRA-C 2004 compliance

1.4.1 Generalities

The C programming language is growing in importance for embedded systems. However, when it comes to developing code for safety-critical applications, this language has many drawbacks. There are several unspecified, implementation-defined, and undefined aspects of the C language that make it unsuited for developing safety-critical systems.

The motor industry software reliability association describes a subset of the C language well suited for developing safety-critical systems in *[1] MISRA-C 2004 Guidelines for the use of the C language in critical systems*.

The STMTouch touch sensing library has been developed to be MISRA-C 2004 compliant.

The following section describes how the STMTouch touch sensing library complies with MISRA-C 2004 (as described in section *4.4 Claiming compliance of the standard* of [1]):

- A compliance matrix has been completed which shows how compliance has been enforced.
- The whole STMTouch touch sensing library source code is compliant with MISRA-C 2004 rules.
- Deviations are documented. A list of all instances of rules not being followed is being maintained, and for each instance there is an appropriately signed-off deviation.
- All the issues listed in section *4.2 The programming language and coding context of the standard* of [1], that need to be checked during the firmware development phase, have been addressed during the development of the STMTouch touch sensing library and appropriate measures have been taken.

1.4.2 Compliance matrix

The compliance of the STMTouch touch sensing library with MISRA-C 2004 has been checked in two ways:

- using PC-lint tool for C/C++ (NT) vers. 8.00v, copyright gimpel software 1985-2006
- performing regular code reviews.

The following table lists the MISRA-C 2004 rules that are frequently violated in the code:

Table 2. MISRA-C 2004 rules not followed

MISRA-C 2004 rule number	Required/ advisory	Summary	Reason of deviance
1.1 1.2	Required	All code shall conform to ISO 9899:1990 standard C, with no extensions permitted.	Compilers extensions are enabled. Comments starting with “//” symbol for code readability.
5.4	Required	A tag name shall be a unique identifier.	Due to the usage of objects methods.

Table 2. MISRA-C 2004 rules not followed (continued)

MISRA-C 2004 rule number	Required/ advisory	Summary	Reason of deviance
8.1	Required	No prototype seen. Functions shall always have prototype declarations and the prototype shall be visible at both the function definition.	This rule is violated as there is no functions prototypes for the objects methods.
10.1 10.2	Required	The value of an expression of integer/floating type shall not be implicitly converted to a different underlying type.	Code complexity
10.3	Required	The value of a complex expression of integer type may only be cast to a type that is narrower and of the same signedness as the underlying type of the expression.	Code complexity
10.5	Required	If the bitwise operators are applied to an operand of underlying type unsigned char or unsigned short, the result shall be immediately cast to the underlying type of the operand.	Use shift on signed quantity for the linear/rotary position
11.3	Advisory	A cast should not be performed between a pointer type and an integral type.	Needed when addressing memory mapped registers.
12.7	Required	Bitwise operators shall not be applied to operands whose underlying type is signed.	Shift of signed value needed
14.3	Required	Before preprocessing, a null statement shall only occur on a line by itself.	Usage of macros to simplify the code

Table 2. MISRA-C 2004 rules not followed (continued)

MISRA-C 2004 rule number	Required/ advisory	Summary	Reason of deviance
14.5	Required	The continue statement shall not be used.	Used to optimize the code speed execution.
19.11	Required	All macro identifiers in preprocessor directives shall be defined before use, except in ifdef and ifndef preprocessor directives and the defined() operator.	All parameters are checked in the check_config files

2 STMTouch touch sensing library

2.1 Supported microcontrollers and development tools

2.1.1 Supported microcontrollers

This STMTouch touch sensing library version supports the following microcontrollers and acquisition modes:

- **Any STM32** microcontroller using the embedded touch sensing controller (TSC): STM32F0 series, STM32F3 series, STM32L0 series and STM32L4 series
  - Surface charge-transfer acquisition principle managed by the touch sensing controller
  - Up to 24 channels (8 groups of 3 channels maximum)
  - Up to 8 channels can be acquired simultaneously
  - Spread spectrum feature
  - Programmable charge transfer frequency and max count value
- **STM32L1 series** microcontrollers
  - Surface charge-transfer acquisition principle managed by:
  - Two timers + routing interface (hardware acquisition mode). This mode is not supported on STM32L1 series microcontrollers featuring 256 K or less memory**.**
  - GPIOs + routing interface (software acquisition mode). This mode is supported by all microcontrollers.
  - Up to 34 channels
  - Up to 11 channels can be acquired simultaneously

2.1.2 Development tools

The STM32 microcontrollers are supported by a full range of development solutions from lead suppliers that deliver start-to-finish control of application development from a single integrated development environment.

The STMTouch touch sensing library has been developed with the following toolchains:

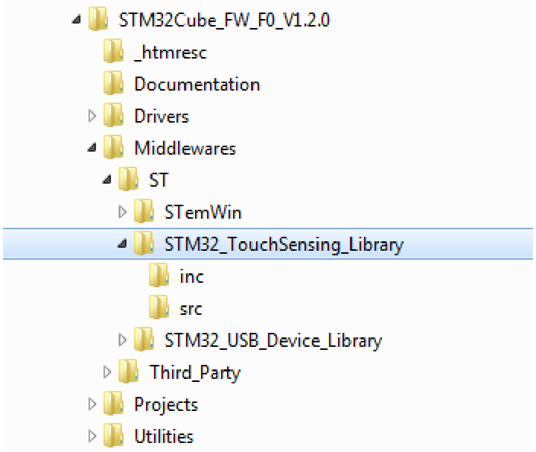
- **EWARM** (IAR)
- **MDK-ARM** (Keil)
- **SW4STM32** (AC6)

For more details about the compilers versions used, please see the STM32Cube package release note.

2.2 Package description

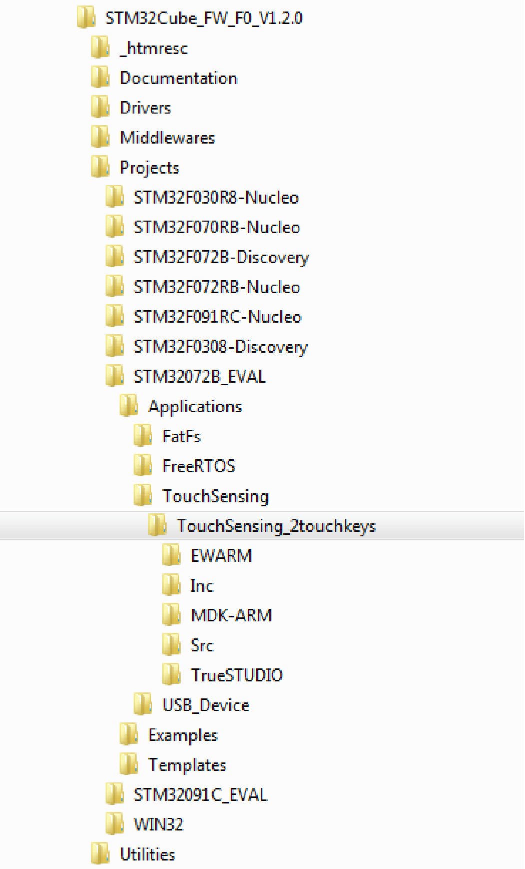
The following snapshots show an example of installation inside the STM32CubeF0 package.

Figure 1. Installation folder 1/2 (library)



```
STM32Cube_FW_F0_V1.2.0
    _htmresc
    Documentation
    Drivers
    Middlewares
        ST
            STemWin
            STM32_TouchSensing_Library
                inc
                src
            STM32_USB_Device_Library
        Third_Party
    Projects
    Utilities
```

Figure 2. Installation folder 2/2 (application example)



```
STM32Cube_FW_F0_V1.2.0
    _htmresc
    Documentation
    Drivers
    Middlewares
    Projects
        STM32F030R8-Nucleo
        STM32F070RB-Nucleo
        STM32F072B-Discovery
        STM32F072RB-Nucleo
        STM32F091RC-Nucleo
        STM32F0308-Discovery
        STM32072B_EVAL
            Applications
                FatFs
                FreeRTOS
                TouchSensing
                    TouchSensing_2touchkeys
                        EWARM
                        Inc
                        MDK-ARM
                        Src
                        TrueSTUDIO
                USB_Device
            Examples
            Templates
        STM32091C_EVAL
        WIN32
    Utilities
```

2.3 Main features

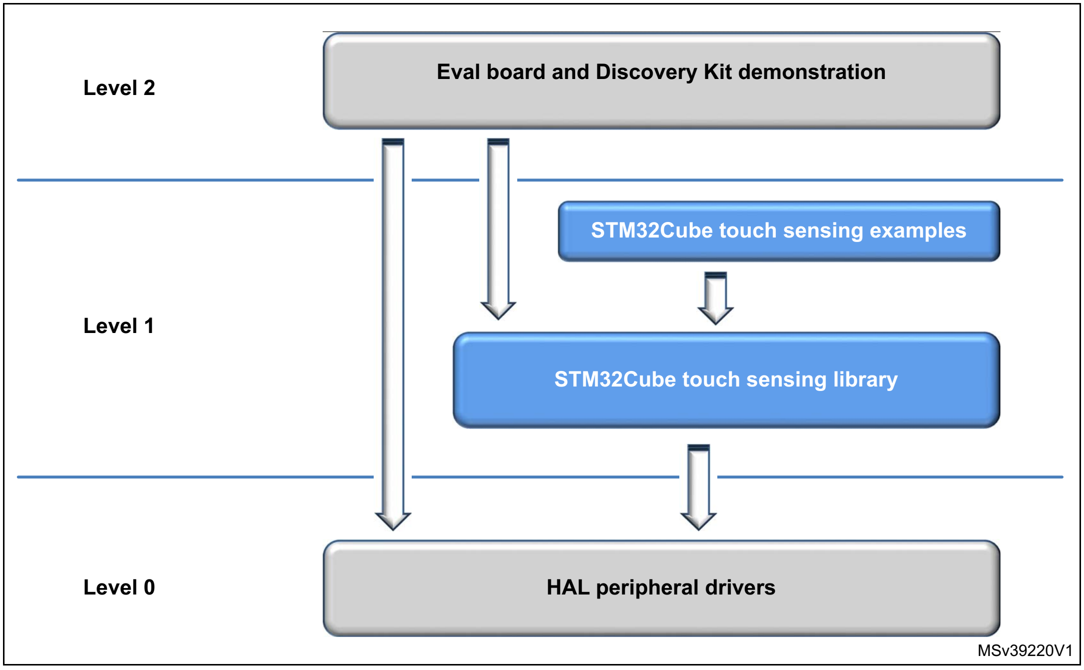
- Supports proximity, touchkeys, linear and rotary touch sensors
- Environment Change System (ECS)
- Detection Time Out (DTO)
- Detection Exclusion System (DXS)
- Noise filter
- Unlimited number of sensors
- Modular architecture allowing easy addition of new acquisitions or sensors
- Each sensor can have its own state machine
- Simplified timing management
- Management of error during acquisition

2.4 Architecture

2.4.1 Overview

The following figure shows the interactions between the STMTouch touch sensing library and the other firmware layers.

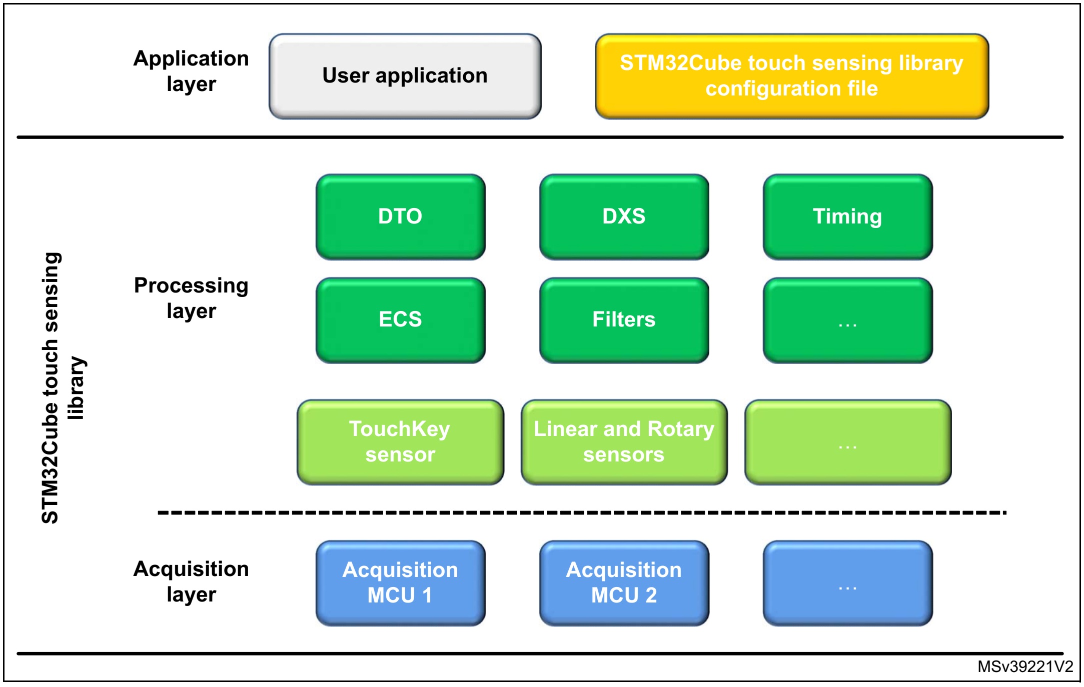
Figure 3. STM32Cube touch sensing library overview


Level 2
Eval board and Discovery Kit demonstration
Level 1
STM32Cube touch sensing examples
STM32Cube touch sensing library
Level 0
HAL peripheral drivers
MSv39220V1

2.4.2 STMTouch touch sensing library layers

The following figure shows a more detailed view of the different STM32Cube touch sensing library layers.

Figure 4. STMTouch touch sensing library detailed layers



The STMTouch touch sensing library is composed of three main layers:

- The acquisition layer
- The processing layer
- The configuration layer

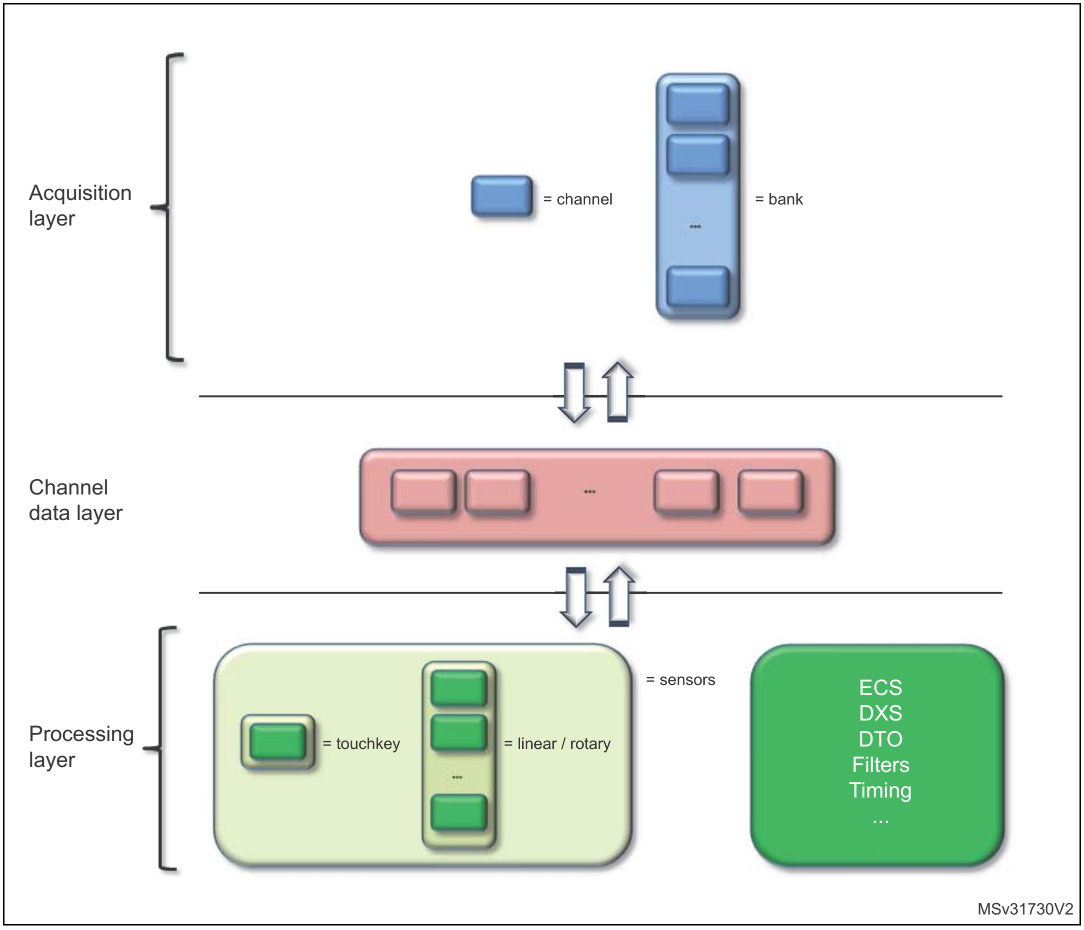
The configuration layer corresponds to what the user needs to write in his application code in order to correctly use the STMTouch touch sensing library. This includes all the channels and sensors declarations, the parameters, etc.

The acquisition and processing layers are described in more details below.

2.4.3 Acquisition and processing layers

The following figure details the acquisition and processing layers and the different elements used in each layer.

Figure 5. Acquisition and processing layers



The **acquisition layer** role is to perform the acquisition of the different channels. The result of the acquisition (measure and flags) is stored inside the channel data layer. These informations will be accessed by the processing layer.

The acquisition layer has only access to the channels and banks. It does not have access to the sensors.

The **channel data layer** role is to share information between the acquisition and processing layers. It stores the result of the acquisition (measure) for each channel and store different informations coming from the processing layer (reference, delta, flags, etc.).

Located in RAM, the ChannelData structure is the only interface between the acquisition and processing layers.

This **processing layer** consists in executing each sensors state machine, executing the different data processing like ECS, DXS, DTO and storing any useful information for the acquisition layer inside the channel data area.

The processing layer does not have direct access to the channels and banks. This access is made through the sensors.

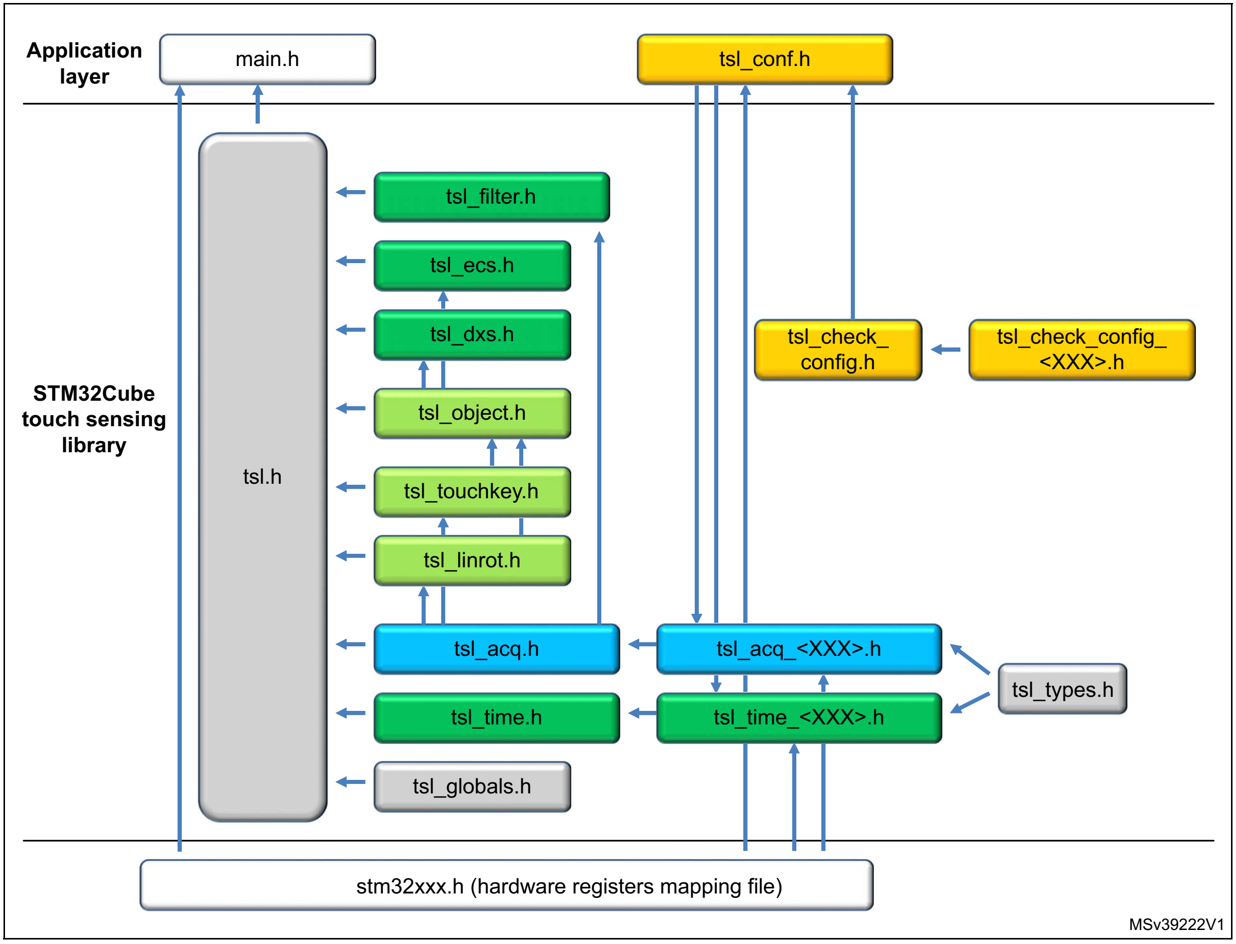
2.4.4 Header files inclusion

The figure below provides a global view of the STMTouch touch sensing library usage and the interaction between the different header files.

In the actual version of the STMTouch touch sensing library, the <XXX> is equal to “tsc” or “stm32l1xx”.

Note: To simplify the drawing, only the most important links are shown. For example the tsl_globals.h file is also included in different files.

Figure 6. Header files inclusion



2.5 Channel

2.5.1 Principle

A channel is the basic element that is used to store several information like:

- where the source measurement can be found after the acquisition is performed (i.e. TSC_IOGxCR registers for TSC acquisition).
- where are stored the measure, the reference, the delta, flags etc.

2.5.2 Resources

A channel is defined by 3 data structures:

- **TSL_ChannelSrc_T**: contains all information about the source measurement (index of the register containing the measurement, masks,...)
- **TSL_ChannelDest_T**: contains all information about the measurement destination (index in the channel data array).
- **TSL_ChannelData_T**: contains all data for the channel (measure, delta, reference, ...)

The channel depends on the acquisition technology. This is why the contents of this structures are not common for all acquisitions. They are declared in each acquisition header files (tsl_acq_<XXX>.h):

- **tsl_acq_stm32l1xx_hw.h** for STM32L1 series microcontrollers using the hardware acquisition mode
- **tsl_acq_stm32l1xx_sw.h** for STM32L1 series microcontrollers using the software acquisition mode
- **tsl_acq_tsc.h** for any STM32 microcontrollers featuring the TSC peripheral

The maximum number of channels is only limited by the device (memory size and channels supported).

The user must declare all the channels arrays in his application code. It can be done directly in the main.c file or in any other file.

2.5.3 Parameters

- TSLPRM_TOTAL_CHANNELS

2.5.4 Usage example

The 3 channels structures must be declared in the application code.

Example of **channel source** array declaration for microcontrollers featuring TSC peripheral. This structure must always be placed in ROM.

```
const TSL_ChannelSrc_T MyChannels_Src[TSLPRM_TOTAL_CHANNELS] =
{ { CHANNEL_0_SRC },
  { CHANNEL_1_SRC },
  { CHANNEL_2_SRC }};
```

Example of **channel destination** array declaration for microcontrollers featuring TSC peripheral. This structure must always be placed in ROM.

```
const TSL_ChannelDest_T MyChannels_Dest[TSLPRM_TOTAL_CHANNELS] =
{ { CHANNEL_0_DEST },
  { CHANNEL_1_DEST },
  { CHANNEL_2_DEST }};
```

Note: The "CHANNEL_x_SRC" and "CHANNEL_x_DEST" are "#define" constants and are used for readability. The values are acquisition dependant.

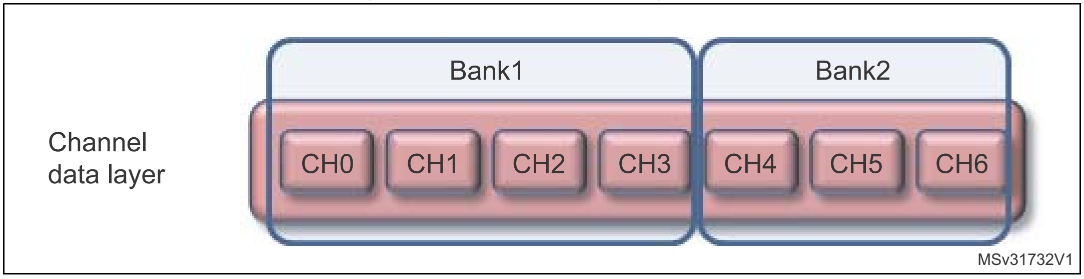
Example of **channel data** array declaration (i.e. channel data layer). This structure must always be placed in RAM.

```
TSL_ChannelData_T MyChannels_Data[TSLPRM_TOTAL_CHANNELS];
```

Warning: When several banks are present, it is mandatory to declare all channels of each bank consecutively in the source and destination structures.

Example:

Figure 7. Channels arrangement



Example of **channel source** array declaration for microcontrollers featuring TSC peripheral.

```
CONST TSL_ChannelSrc_T MyChannels_Src[TSLPRM_TOTAL_CHANNELS] =
{
// Bank 1
{ CHANNEL_0_SRC, CHANNEL_0_IO_MSK, CHANNEL_0_GRP_MSK },
{ CHANNEL_1_SRC, CHANNEL_1_IO_MSK, CHANNEL_1_GRP_MSK },
{ CHANNEL_2_SRC, CHANNEL_2_IO_MSK, CHANNEL_2_GRP_MSK },
{ CHANNEL_3_SRC, CHANNEL_3_IO_MSK, CHANNEL_3_GRP_MSK },
// Bank 2
{ CHANNEL_4_SRC, CHANNEL_4_IO_MSK, CHANNEL_4_GRP_MSK },
{ CHANNEL_5_SRC, CHANNEL_5_IO_MSK, CHANNEL_5_GRP_MSK },
{ CHANNEL_6_SRC, CHANNEL_6_IO_MSK, CHANNEL_6_GRP_MSK }
};
```

2.6 Bank

2.6.1 Principle

A bank is a group of channels that are acquired simultaneously. The number of channels in the bank is variable.

2.6.2 Resources

The bank data are held by only one structure:

- TSL_Bank_T
- The bank depends also on the acquisition technology. Structures are declared in each acquisition header files (tsl_acq_<XXX>.h):

The maximum number of banks is only limited by the device.

The user must declare all the bank arrays in his application code. It can be done directly in the main.c file or in any other file.

The banks are used mainly by the functions described below. Some functions are common whatever the device and acquisition technology. Some others are dependent on the device.

Common functions:

- TSL_acq_BankGetResult()
- TSL_acq_BankCalibrate()

Device dependent functions:

- TSL_acq_BankConfig()
- TSL_acq_BankStartAcq()
- TSL_acq_BankWaitEOC()

2.6.3 Parameters

- TSLPRM_TOTAL_BANKS

2.6.4 Usage example

Example of 3 banks declaration for microcontrollers featuring TSC peripheral:

```
CONST TSL_Bank_T MyBanks[TSLPRM_TOTAL_BANKS] = {
  {&MyChannels_Src[0], &MyChannels_Dest[0], MyChannels_Data,
BANK_0_NBCHANNELS, BANK_0_MSK_CHANNELS, BANK_0_MSK_GROUPS},
  {&MyChannels_Src[1], &MyChannels_Dest[1], MyChannels_Data,
BANK_1_NBCHANNELS, BANK_1_MSK_CHANNELS, BANK_1_MSK_GROUPS},
  {&MyChannels_Src[2], &MyChannels_Dest[2], MyChannels_Data,
BANK_2_NBCHANNELS, BANK_2_MSK_CHANNELS, BANK_2_MSK_GROUPS}
};
```

2.7 Objects

2.7.1 Principle

The term “object” or “sensor” stands for any sensor type (touchkeys, linear and rotary touch sensors, etc.) supported by the STMTouch touch sensing library.

2.7.2 Resources

All processing that affect the sensors in general are defined in:

- tsl_object.c
- tsl_object.h

The functions are:

- TSL_obj_GroupInit()
- TSL_obj_GroupProcess()
- TSL_obj_SetGlobalObj()

A sensor is described by the structures:

- TSL_Object_T
- TSL_ObjectGroup_T

2.7.3 Parameters

- TSLPRM_TOTAL_OBJECTS

2.7.4 Usage example

First, all touchkeys, linear and rotary touch sensors (described after) used in the application must be described first as 'generic' sensors or objects.

Example:

```
// Mix of touchkeys and Linear touch sensors
const TSL_Object_T MyObjects[TSLPRM_TOTAL_OBJECTS] =
{
  // TKeys
  { TSL_OBJ_TOUCHKEYB, (TSL_TouchKeyB_T *)&MyTKeys[0]   },
  { TSL_OBJ_TOUCHKEYB, (TSL_TouchKeyB_T *)&MyTKeys[1]   },
  // Linear touch sensors
  { TSL_OBJ_LINEARB,   (TSL_LinRotB_T   *)&MyLinRots[0] }
};
```

These objects must be placed in ROM memory.

Once this done, it is necessary to create at least one group of sensors. Groups of sensors are used by the different processing routines (ECS, DXS, etc.).

These groups of objects must be placed in RAM.

Example:

```
TSL_ObjectGroup_T MyObjGroup_All = {
 MyObjects,
 3,
 0,
 TSL_STATE_NOT_CHANGED
};
```

Then, all the sensors must be initialized and “processed”. This is done in the main function of the application:

```
int main(void) {
  ...
  TSL_obj_GroupInit(&MyObjGroup_All);
  ...
  while (1) {
    ...
    TSL_obj_GroupProcess(&MyObjGroup_All);
    ...
  }
}
```

2.8 Touchkey sensor

2.8.1 Principle

The touchkey sensor is composed of only one channel. It acts as a simple “button” with two states RELEASE and DETECT (or TOUCH if DXS is enabled).

2.8.2 Resources

All the functions related to this sensor are described in the files:

- tsl_touchkey.c
- tsl_touchkey.h

Two types of touchkey sensor are available:

- Basic: defined by the **TSL_TouchKeyB_T** structure
- Extended: defined by the **TSL_TouchKey_T** structure

Two functions (called methods) are used to initialize the sensor parameters and to run the sensor state machine:

- TSL_tkey_Init()
- TSL_tkey_Process()

The difference between the “basic” and “extended” types concerns the usage of the methods and sensor state machine.

For the “basic” sensor, the methods and state machine are those used in the **TSL_Params** structure.

For the “extended” sensor, the methods and state machine are those declared in their own structure.

2.8.3 Parameters

- TSLPRM_TOTAL_TKEYS

2.8.4 Usage example

The user must declare these methods in the application code.

Note: *One can also use one’s own initialization and process functions instead:*

```
const TSL_TouchKeyMethods_T MyTKeys_Methods =
{
  TSL_tkey_Init,
  TSL_tkey_Process
};
```

The declaration of the touchkey sensor is done by the user in the application code:

Example with “basic” sensor:

```
// "Basic" touchkeys: Always placed in ROM
const TSL_TouchKeyB_T MyTKeys[TSLPRM_TOTAL_TKEYS] =
{
  { &MyTKeys_Data[0], &MyTKeys_Param[0], &MyChannels_Data[0] },
  { &MyTKeys_Data[1], &MyTKeys_Param[1], &MyChannels_Data[1] },
  { &MyTKeys_Data[2], &MyTKeys_Param[2], &MyChannels_Data[2] }
};
```

Example with “extended” sensor:

```
// "Extended" TouchKeys: Always placed in ROM
const TSL_TouchKey_T MyTKeys[TSLPRM_TOTAL_TKEYS] =
{
  { &MyTKeys_Data[0], &MyTKeys_Param[0], &MyChannels_Data[0],
MyTKeys_StateMachine, &MyTKeys_Methods },
  { &MyTKeys_Data[1], &MyTKeys_Param[1], &MyChannels_Data[1],
MyTKeys_StateMachine, &MyTKeys_Methods },
  { &MyTKeys_Data[2], &MyTKeys_Param[2], &MyChannels_Data[2],
MyTKeys_StateMachine, &MyTKeys_Methods }
};
```

2.9 Linear and rotary touch sensors

2.9.1 Principle

The linear and rotary touch sensors are like a touchkey sensor except that they are composed of a variable number of channels. The difference between the linear and rotary touch sensors is how the electrodes are organized together.

The linear and rotary touch sensors have additional fields in their structure compared to touchkey sensors:

- Number of channels
- Delta coefficient table
- Position offset table
- Sector computation parameter
- Position correction parameter for linear sensor

The last 3 fields are used to calculate the position.

2.9.2 Number of channels

Only 1, 3, 4, 5 and 6 channels are supported today by the STMTouch touch sensing library. Additional number of channels can be added by the end-user.

Note: A linear touch sensor with 1 channel is equivalent to one touchkey sensor. When an application uses both touchkey sensor and linear and rotary sensor, it is better to use touchkeys with a 1-channel linear touch sensor. In this case the gain in memory size is important as the touchkey sensor state machine is not used.

2.9.3 Delta coefficient table

The delta coefficient table is used to adjust each channel of the linear and rotary touch sensors. Each value is a 16-bit integer. The MSB is the integer part, the LSB is the real part.

Examples:

To apply a factor of 1.10:

- MSB equal 0x01
- LSB equal 0x1A (0.10 x 256 = 25.6 -> rounded to 26 = 0x1A)

To apply a factor 1.00:

- MSB equal 0x01
- LSB equal 0x00

To apply a factor 0.90:

- MSB equal 0x00
- LSB equal 0xE6 (0.90 x 256 = 230.4 -> rounded to 230 = 0xE6)

This results in the following delta coefficient table:

```
CONST uint16_t MyLinRot0_DeltaCoeff[3] = {0x011A, 0x0100, 0x00E6};
```

The number of delta coefficient table is not limited. The same delta coefficient table can be shared by several linear and rotary touch sensors.

2.9.4 Electrodes placement

The placement (design) of the electrodes can be done in three different manners:

1. Mono electrode design

   The number of electrodes is equivalent to the number of channels. This design is used for linear and rotary touch sensors.

   Abbreviations: **LIN_M1**, **LIN_M2** and **ROT_M**

   Examples:

   - CH1 CH2 CH3
   - CH1 CH2 CH3 CH4
   - CH1 CH2 CH3 CH4 CH5

2. Dual electrode design

   All the electrodes are duplicated and interlaced together in order to increase the touch area.

   This design is used for linear and rotary touch sensors composed **with at least 5 channels**.

   Abbreviation: **LIN_D** and **ROT_D**

   Examples with 5 channels:

   - CH1 CH2 CH3 CH4 CH5 CH1 CH3 CH5 CH2 CH4
   - CH1 CH2 CH3 CH4 CH5 CH2 CH4 CH1 CH3 CH5
   - CH1 CH2 CH3 CH4 CH5 CH3 CH1 CH4 CH2 CH5

3. Half-ended electrode design

   The first electrode is duplicated and the replica is placed at the end. The size of the first and last electrode is **half the size** of the other electrodes. This design is used for **linear sensors only**. The 0 and 255 positions are obtained more easily compared to the Mono electrodes design.

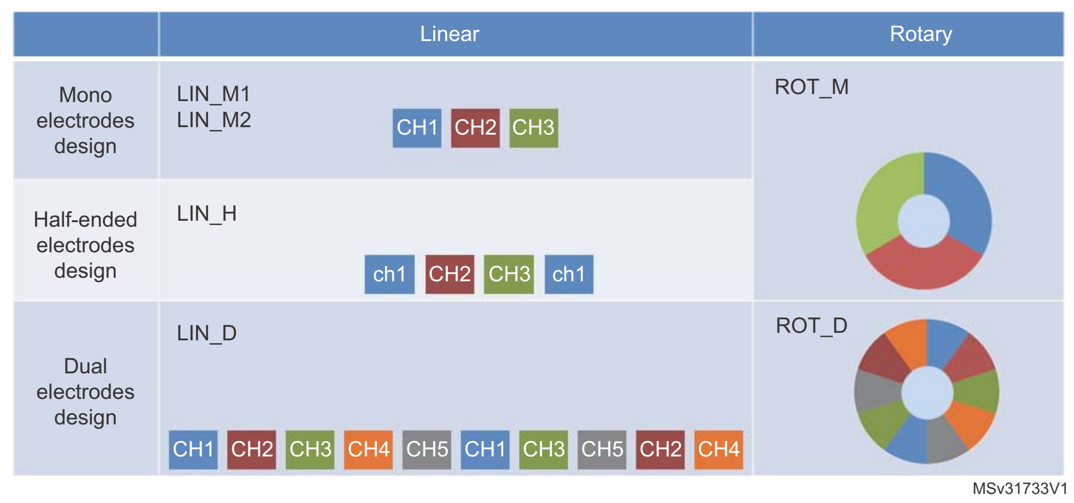
   Abbreviation: **LIN_H**

   Examples:

   - ch1 CH2 CH3 ch1
   - ch1 CH2 CH3 CH4 ch1
   - ch1 CH2 CH3 CH4 CH5 ch1

The following figure summarizes the different electrodes designs we can have on linear and rotary touch sensors:

Figure 8. Electrodes designs



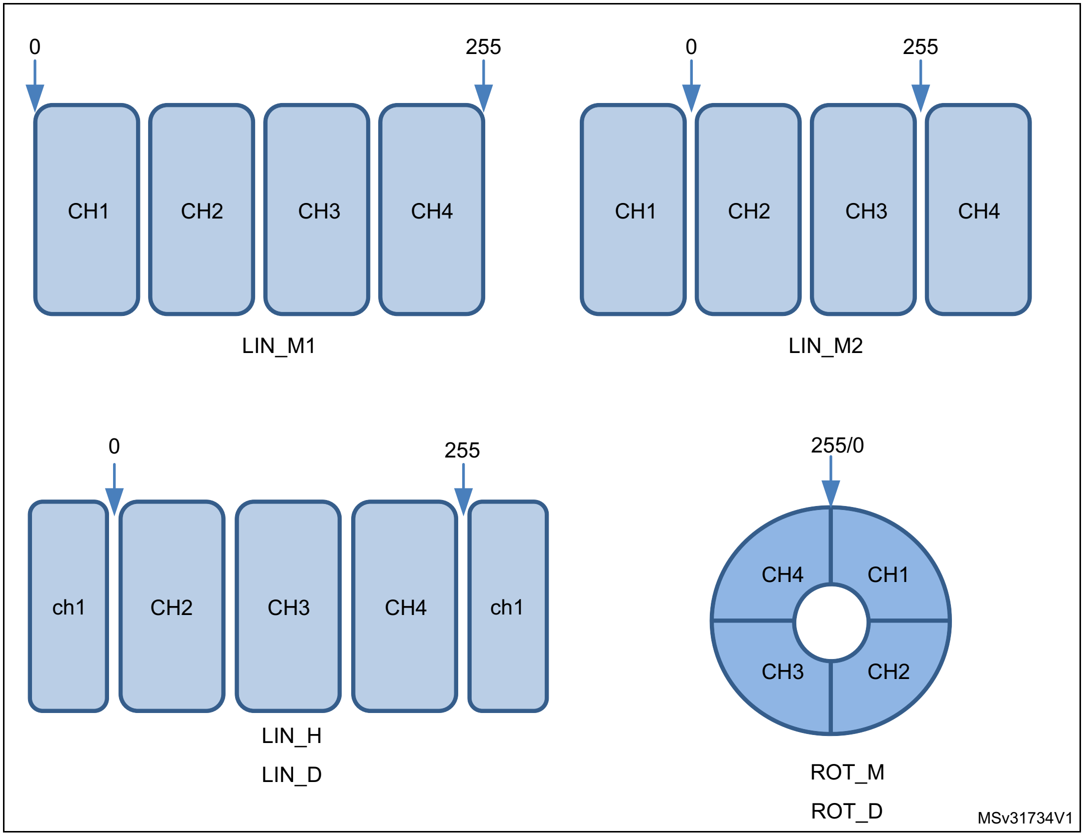
Positions 0 and 255

Special care must be taken for the 0 and 255 positions on linear sensors. These positions are placed differently depending on the electrodes design used:

- **LIN_M1**: the 0 and 255 positions are placed completely at the sensor's **extremities.** These positions can be obtain with difficulty if the electrodes are too big or if they are separated by an important space.
- **LIN_M2**, **LIN_H** and **LIN_D**: the 0 position is placed **between the first and second electrodes**. The 255 position is placed **between the last two electrodes**.
- **ROT_M** and **ROT_D**: the 0 and 255 positions are always placed **between the first and the last electrodes**.

The following figures summarizes the different placements of the 0 and 255 positions with 4 channels sensors:

Figure 9. Positions 0 and 255



The following table summarizes the different linear and rotary touch sensors electrodes designs supported by the STMTouch touch sensing library:

Table 3. Supported linear and rotary touch sensors

Number of Channels	LIN_M1	LIN_M2	LIN_H	LIN_D	ROT_M	ROT_D
3	Yes	Yes	Yes	No	Yes	No
4	Yes	Yes	Yes	No	Yes	No
5	Yes	Yes	Yes	No	Yes	Yes
6	Yes	Yes	Yes	No	Yes	No

Each supported electrode design is described by 3 fields in the **TSL_LinRot_T** or **TSL_LinRotB_T** structures:

- Position offset table
- Sector computation parameter
- Position correction parameter for linear sensor

These 3 fields are defined in the **tsl_linrot.c** and **tsl_linrot.h** files and follow the naming convention:

Position offset table: TSL_POSOFF_nCH_[LIN|ROT]_[M1|M2|H|D]

Sector computation parameter: TSL_SCTCOMP_nCH_[LIN|ROT]_[M1|M2|H|D]

Position correction parameter for linear sensor: TSL_POSCORR_nCH_LIN_[M1|M2|H|D]

With:

- n = number of channels
- LIN = linear sensor
- ROT = rotary sensor
- M1 = mono electrodes design with 0/255 position at extremities
- M2 = mono electrodes design
- H = half-ended electrodes design
- D = dual electrodes design

In order to gain memory space, each table is only compiled if its corresponding parameter is set in the configuration file:

TSLPRM_USE_nCH_[LIN|ROT]_[M1|M2|H|D]

2.9.5 Resources

All the functions related to this sensor are described in the files:

- tsl_linrot.c
- tsl_linrot.h

Two types of linear and rotary sensor are available:

- basic: defined by the **TSL_LinRotB_T** structure
- extended: defined by the **TSL_LinRot_T** structure

The difference between “basic” and “extended” is the same as for the touchkey sensor.

Three functions (called methods) are used to initialized the sensor parameters, run the sensor state machine and calculate the position.

- TSL_linrot_Init()
- TSL_linrot_Process()
- TSL_linrot_CalcPos()

2.9.6 Parameters

- TSLPRM_TOTAL_LINROTS

2.9.7 Usage example

The user must declared these methods in the application code.

Note: One can also use one’s own initialization and process functions instead:

```
CONST TSL_LinRotMethods_T MyLinRots_Methods =
{
  TSL_linrot_Init,
  TSL_linrot_Process,
  TSL_linrot_CalcPos
};
```

The declaration of the linear and rotary sensor is done by the user in the application code in the same manner as for touchkey sensor.

Example with 2 “basic” linear touch sensors, one with 3 channels half-ended and the other with 5 channels mono electrodes design:

```
CONST TSL_LinRotB_T MyLinRots[2] =
{
  // LinRot sensor 0
  &MyLinRots_Data[0],
  &MyLinRots_Param[0],
  &MyChannels_Data[CHANNEL_9_DEST],
  3, // Number of channels
  MyLinRot0_DeltaCoeff, // Delta coefficient table
  (TSL_tsignPosition_T *)TSL_POSOFF_3CH_LIN_H, // Position table
  TSL_SCTCOMP_3CH_LIN_H, // Sector compensation
  TSL_POSCORR_3CH_LIN_H, // Position correction
  // LinRot sensor 1
  &MyLinRots_Data[1],
  &MyLinRots_Param[1],
  &MyChannels_Data[CHANNEL_12_DEST],
  5, // Number of channels
  MyLinRot1_DeltaCoeff, // Delta coefficient table
  (TSL_tsignPosition_T *)TSL_POSOFF_5CH_LIN_M2, // Position table
  TSL_SCTCOMP_5CH_LIN_M2, // Sector compensation
  TSL_POSCORR_5CH_LIN_M2  // Position correction
};
```

Example of one “extended” (i.e. having its own state machine and methods) linear touch sensor with 3 channels half-ended:

```
CONST TSL_LinRot_T MyLinRots[1] =
{
  // LinRot sensor 0
  &MyLinRots_Data[0],
  &MyLinRots_Param[0],
  &MyChannels_Data[CHANNEL_0_DEST],
  3, // Number of channels
  MyLinRot0_DeltaCoeff,
  (TSL_tsignPosition_T *)TSL_POSOFF_3CH_LIN_H,
  TSL_SCTCOMP_3CH_LIN_H,
  TSL_POSCORR_3CH_LIN_H,
  MyLinRots_StateMachine, // Specific state machine
  &MyLinRots_Methods // Specific methods
};
```

Example of one “extended” rotary touch sensor with 3 channels mono electrode design:

```
CONST TSL_LinRot_T MyLinRots[0] =
{
  // LinRot sensor 0
  &MyLinRots_Data[0],
  &MyLinRots_Param[0],
  &MyChannels_Data[CHANNEL_0_DEST],
  3, // Number of channels
  MyLinRot0_DeltaCoeff,
  (TSL_tsignPosition_T *)TSL_POSOFF_3CH_ROT_M,
  TSL_SCTCOMP_3CH_ROT_M,
  0, // No position correction needed on a Rotary sensor
  MyLinRots_StateMachine, // Specific state machine
  &MyLinRots_Methods // Specific methods
};
```

2.10 Main state machine

The main state machine is managed by the user in the application layer. A set of functions are available to accomplish this task. The main state machine can be defined with polling or with interrupt modes, using one or several banks. The modularity of the STMTouch touch sensing library allows also the application code to be inserted between acquisition and processing tasks. Several examples are given below.

The functions to use for the acquisition are:

- TSL_acq_BankConfig()
- TSL_acq_BankStartAcq()
- TSL_acq_BankWaitEOC()
- TSL_acq_BankGetResult()

These functions are device dependent and are described in the **tsl_acq_<XXX>.c** files.

The functions to use for the processing are:

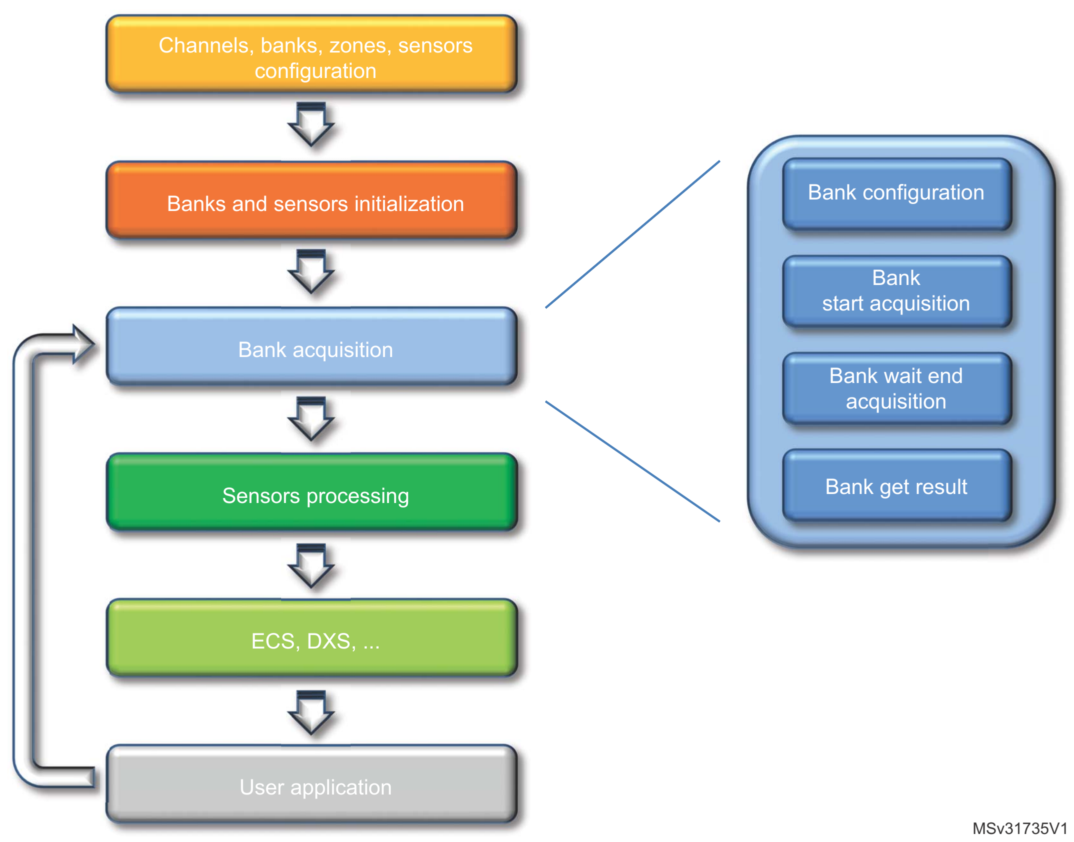
- TSL_obj_GroupProcess()
- TSL_ecs_Process()
- TSL_dxs_FirstObj()

Other functions that can be used during the processing:

- TSL_tim_CheckDelay_ms()
- TSL_obj_SetGlobalObj()
- TSL_tkey_GetStateId()
- TSL_tkey_GetStateMask()
- TSL_linrot_SetStateOff()
- TSL_linrot_SetStateCalibration()

The main state machine principle is illustrated by the figure below:

Figure 10. Main state machine



The main state machine steps are:

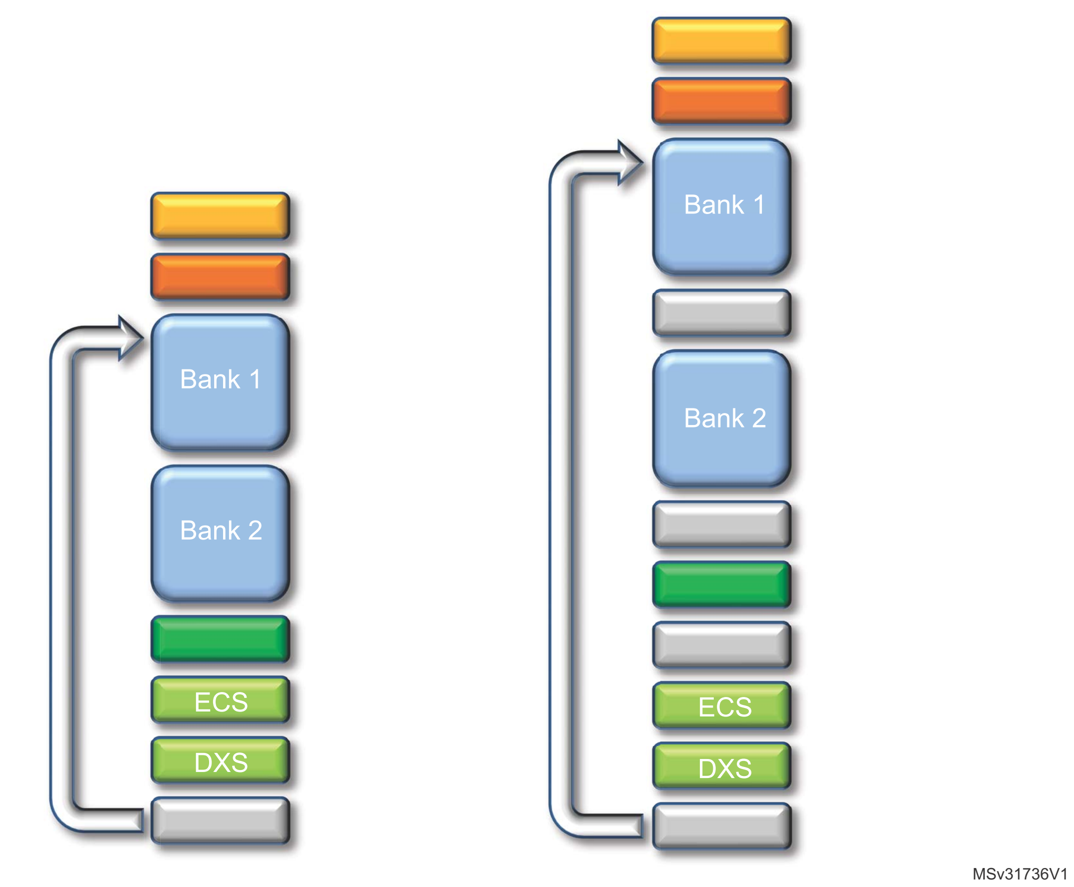
1. The **channels, banks and sensors configuration** step are used to declare all the different elements. This is done in the global declaration section in the main application file. See the section associated to each element for more details.
2. The **banks and sensors initialization** step is used to initialize the STMTouch touch sensing library modules. The sensors parameters are initialized with their default value defined in the configuration files.
3. **The banks acquisition** step is used to perform the acquisition of the banks. It is composed of 4 sub-steps:
   - **configuration**: used to configure all channels of the bank
   - **start acquisition**: used to launch the measurement on all channels of the bank
   - **wait end acquisition**: used to wait the end of acquisition of all channels of the bank
   - **get result**: used to read all the channels measurements and to store them in the channel data layer.
4. **The sensors processing** step is used to execute the state machine of the sensors.

Note: The debouncing, Detection Time Out and re-calibration are automatically performed inside this step.

5. The **ECS, DXS** step is used to execute other algorithms that are not performed in the sensor state machine like the ECS, DXS, other filters, etc. This step is optional and it can be executed at certain time intervals (mainly for ECS).
6. The **user application** step is used to execute the application layer (read the sensors state, decide which actions to perform, manage ERROR states, etc.). The user application can also be placed between other steps, for example it can be done between the “sensors processing” step and the "ECS/DXS".

There are multiple manners to perform the main state machine. The following figures show some examples with two banks.

Figure 11. Example of main state machine



2.11 Sensors state machine

2.11.1 Overview

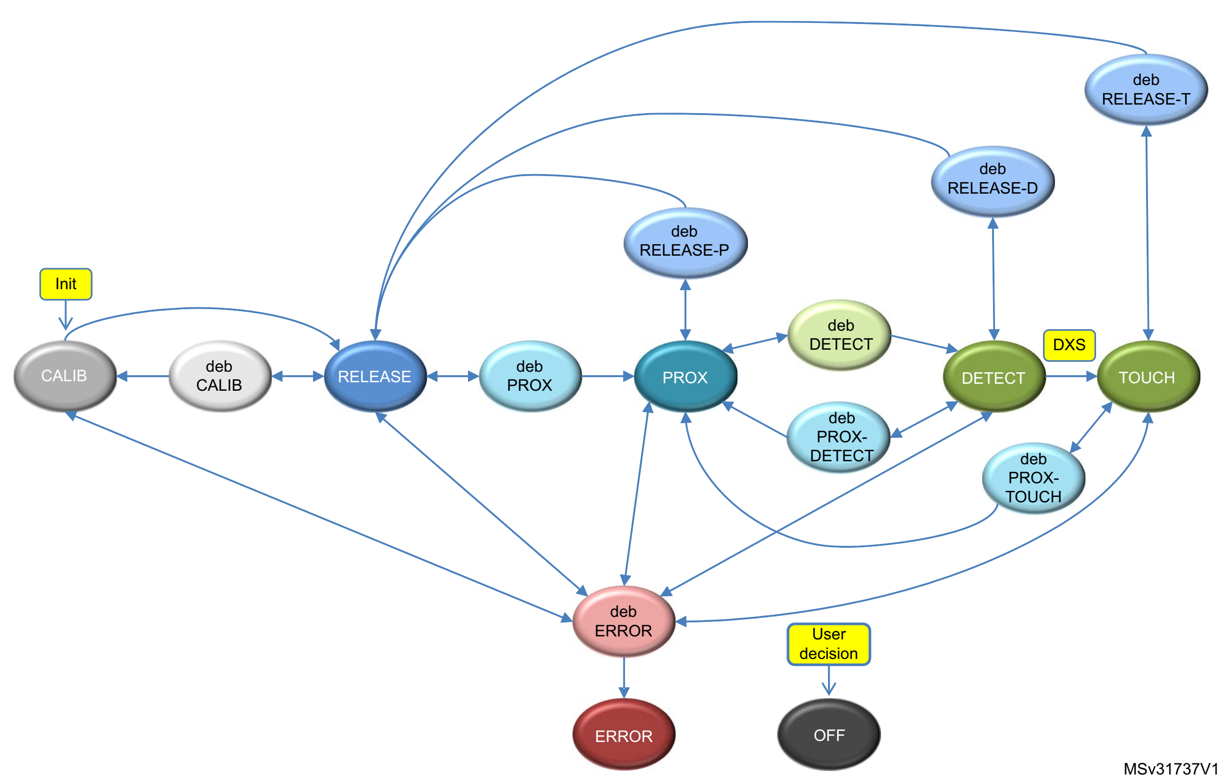
The state machine is managed in the files:

- **tsl_touchkey.c** and **tsl_touchkey.h** for the touchkey sensors
- **tsl_linrot.c** and **tsl_linrot.h** for the linear and rotary touch sensors

There is a total of **20 states** defined in the **TSL_StateId_enum_T** structure.

The following figure shows the simplified state machine used by any sensor (for clarity not all the connections between states are shown).

Figure 12. Simplified sensors state machine



2.11.2 States constant table

Each state ID is associated to a mask and a function. The association STATE_ID-mask-function is made in the user application code using a constant table of the **TSL_State_T** type. The name of this table is free and user can give any name he wants. If no function is needed simply put a zero instead of the function name.

Here below an example of touchkey sensors state machine:

```
// Touchkeys state machine
const TSL_State_T MyTKeys_StateMachine[] =
{
//----------------------------------------------------------------------
// ID      MASK                                 FUNCTION
//----------------------------------------------------------------------
// Calibration states
/*  0 */ { TSL_STATEMASK_CALIB, TSL_tkey_CalibrationStateProcess },
/*  1 */ { TSL_STATEMASK_DEB_CALIB, TSL_tkey_DebCalibrationStateProcess },
// RELEASE states
/*  2 */ { TSL_STATEMASK_RELEASE,           TSL_tkey_ReleaseStateProcess },
#if TSLPRM_USE_PROX > 0
/*  3 */ { TSL_STATEMASK_DEB_RELEASE_PROX,
TSL_tkey_DebReleaseProxStateProcess },
#else
/*  3 */ { TSL_STATEMASK_DEB_RELEASE_PROX, 0 },
#endif
/*  4 */ { TSL_STATEMASK_DEB_RELEASE_DETECT,
TSL_tkey_DebReleaseDetectStateProcess },
/*  5 */ { TSL_STATEMASK_DEB_RELEASE_TOUCH,
TSL_tkey_DebReleaseTouchStateProcess },
#if TSLPRM_USE_PROX > 0
// Proximity states
/*  6 */ { TSL_STATEMASK_PROX,               TSL_tkey_ProxStateProcess },
/*  7 */ { TSL_STATEMASK_DEB_PROX,          TSL_tkey_DebProxStateProcess },
/*  8 */ { TSL_STATEMASK_DEB_PROX_DETECT,
TSL_tkey_DebProxDetectStateProcess },
/*  9 */ { TSL_STATEMASK_DEB_PROX_TOUCH,
TSL_tkey_DebProxTouchStateProcess },
#else
/*  6 */ { TSL_STATEMASK_PROX,               0 },
/*  7 */ { TSL_STATEMASK_DEB_PROX,           0 },
/*  8 */ { TSL_STATEMASK_DEB_PROX_DETECT,    0 },
/*  9 */ { TSL_STATEMASK_DEB_PROX_TOUCH,     0 },
#endif
// DETECT states
/* 10 */ { TSL_STATEMASK_DETECT,             TSL_tkey_DetectStateProcess },
/* 11 */ { TSL_STATEMASK_DEB_DETECT,        TSL_tkey_DebDetectStateProcess
},
// TOUCH state
/* 12 */ { TSL_STATEMASK_TOUCH,              TSL_tkey_TouchStateProcess },
// ERROR states
/* 13 */ { TSL_STATEMASK_ERROR,              MyTKeys_ErrorStateProcess },
/* 14 */ { TSL_STATEMASK_DEB_ERROR_CALIB,    TSL_tkey_DebErrorStateProcess
},
/* 15 */ { TSL_STATEMASK_DEB_ERROR_RELEASE,  TSL_tkey_DebErrorStateProcess
},
/* 16 */ { TSL_STATEMASK_DEB_ERROR_PROX,     TSL_tkey_DebErrorStateProcess
},
/* 17 */ { TSL_STATEMASK_DEB_ERROR_DETECT,   TSL_tkey_DebErrorStateProcess
},
/* 18 */ { TSL_STATEMASK_DEB_ERROR_TOUCH,    TSL_tkey_DebErrorStateProcess
},
// Other states
/* 19 */ { TSL_STATEMASK_OFF,                MyTKeys_OffStateProcess }
};
```

The STMTouch touch sensing library contains all the functions needed to manage each state. However the user can copy and adapt one or several functions to fit the requirements of his application.

Example:

```
/*  0 */ { TSL_STATEMASK_CALIB, MyTkeys_CalibrationStateProcess },
```

Note: The two functions used to manage the ERROR and OFF states are not part of the STMTouch touch sensing library. These functions are managed by the application.

For linear and rotary sensor state machine, it is the same principle. The functions used to manage each state start with the prefix “TSL_linrot_”:

```
CONST TSL_State_T MyLinRots_StateMachine[] =
{
// Calibration states
/*  0 */ { TSL_STATEMASK_CALIB, TSL_linrot_CalibrationStateProcess },
```

2.11.3 States detail

The two tables below show the detail of how each state is entered following the thresholds measured.

Table 4. Detailed sensors states 1/2

Previous state	**all excepted 13**	**all excepted 13**	**2p,10p,12p,3, 4p,5p,7,8,9, 11p**	**2,4,11**	**2p,6,4p,7,8,1 1p**	**DXS,5**	**DXS,5p,9**	**2,2p,1**	**2,2p,6,10, 10p,12,12p ,0,14..18**
state nb	**2**	**2p**	**6**	**10**	**10p**	**12**	**12p**	**0**	**13**
Current state	**RELEASE**	**RELEASE with PROX**	**PROX**	**DETECT**	**DETECT with PROX**	**TOUCH**	**TOUCH with PROX**	**CALIB**	**ERROR**
Delta									
DETECT IN Th	deb DETECT or DETECT+DTO	deb DETECT or DETECT+DTO	deb DETECT or DETECT+DTO	same or CALIB if DTO	same or CALIB if DTO	same or CALIB if DTO	same or CALIB if DTO	RELEASE or ERROR	same
	same	deb PROX or PROX+DTO	same or CALIB if DTO						
DETECT OUT Th PROX IN Th				deb RELEASE-DETECT or RELEASE	deb PROX-DETECT or PROX+DTO	deb RELEASE-TOUCH or RELEASE	deb PROX-TOUCH or PROX+DTO		
		same							
PROX OUT Th			deb RELEASE-PROX or RELEASE		deb RELEASE-DETECT or RELEASE		deb RELEASE-TOUCH or RELEASE		
CALIB Th	deb CALIB or CALIB	deb CALIB or CALIB							
if ACQ ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	deb ERROR or ERROR	same

Table 5. Detailed sensors states 2/2

Previous state	6	10	10p,8	12	12p,9	2p,11p	10p	12p	2	2p,6,7	2,2p	2,2p,6,10,10p,12,12p,0
state nb	**3**	**4**	**4p**	**5**	**5p**	**7**	**8**	**9**	**11**	**11p**	**1**	**14..18**
Current state	**deb RELEASE-PROX**	**deb RELEASE-DETECT**	**deb RELEASE-DETECT with PROX**	**deb RELEASE-TOUCH**	**deb RELEASE-TOUCH with PROX**	**deb PROX**	**deb PROX-DETECT**	**deb PROX-TOUCH**	**deb DETECT**	**deb DETECT with PROX**	**deb CALIB**	**deb ERROR**
Delta												
DETECT IN Th	PROX	DETECT	DETECT	TOUCH	TOUCH	deb DETECT or DETECT+ DTO	DETECT	TOUCH	same or DETECT+ DTO	same or DETECT+ DTO	RELEASE	RE LEASE PROX DETECT TOUCH CALIB
						same or PROX+ DTO			RELEASE	deb PROX or PROX+ DTO		
DETECT OUT Th PROX IN Th		same or RELEASE	PROX	same or RELEASE	PROX		same or PROX+ DTO	same or PROX+ DTO				
						RELEASE				RELEASE		
PROX OUT Th	same or RELEASE		same or RELEASE		same or RELEASE		deb RELEASE-DETECT or RELEASE	deb RELEASE-TOUCH or RELEASE				
CALIB Th											same or CALIB	
if ACQ ERROR	PROX	DETECT	DETECT	TOUCH	TOUCH	RELEASE	DETECT	TOUCH	RELEASE	RELEASE	RELEASE	ERROR

2.11.4 CALIBRATION state

It consists in calculating the reference for all the channels of a sensor. An average of a certain number of measurements is done.

The number of measurement samples to use for the calibration is defined by the **TSLPRM_CALIB_SAMPLES** parameter.

After reset the initialization method of each object is called. This method initializes the sensor parameters and then goes in the CALIBRATION state. After the calibration is done, the sensor goes in the RELEASE state or ERROR state if an error occurred.

Related functions:

- TSL_tkey_CalibrationStateProcess()
- TSL_linrot_CalibrationStateProcess()
- TSL_tkey_SetStateCalibration()
- TSL_linrot_SetStateCalibration()

Calibration delay

If a noise filter is used it should be necessary to wait a certain amount of measurement samples before to start the reference calculation. This number of samples to wait is defined by the **TSLPRM_CALIB_DELAY** parameter.

Re-calibration

If the calibration threshold is reached while in RELEASE state, a new calibration is performed. This “re-calibration” prevents the application to get stuck if something touches permanently the sensor like a drop of water for example or if the sensor is touched upon power-on.

2.11.5 RELEASE state

Corresponds to the “idle” state of the sensor when no presence is detected.

Related functions:

- TSL_tkey_ReleaseStateProcess()
- TSL_linrot_ReleaseStateProcess()

2.11.6 PROXIMITY state

This state is optional and is enabled or disabled using the **TSLPRM_USE_PROX** parameter.

Related functions:

- TSL_tkey_ProxStateProcess()
- TSL_linrot_ProxStateProcess()

2.11.7 DETECT state

It is the “normal” state when the sensor is touched.

Related functions:

- TSL_tkey_DetectStateProcess()
- TSL_linrot_DetectStateProcess()

2.11.8 TOUCH state

Same as DETECT state excepted that it is entered only by the DXS processing. If the DXS is not used this state is never entered.

Related functions:

- TSL_tkey_TouchStateProcess()
- TSL_linrot_TouchStateProcess()

2.11.9 ERROR state

It is used to catch all acquisition errors detected in the other states.

The management of this state must be performed at application level.

2.11.10 OFF state

It is used to inform the acquisition module to stop the burst and/or acquisition on the sensor's channels.

The management of this state must be performed at application level.

2.11.11 DEBOUNCE states

The debounce is optional and is enabled/disabled using the different debounce counters parameters: **TSLPRM_DEBOUNCE_PROX**, **TSLPRM_DEBOUNCE_DETECT**, **TSLPRM_DEBOUNCE_RELEASE**, **TSLPRM_DEBOUNCE_CALIB**, **TSLPRM_DEBOUNCE_ERROR**

The debounce is off if the corresponding parameter is equal to zero.

2.11.12 Reading the current state

The current state can be obtained by using the functions:

For touchkey sensor:

- TSL_tkey_GetStateId()
- TSL_tkey_GetStateMask()

For linear and rotary sensor:

- TSL_linrot_GetStateId()
- TSL_linrot_GetStateMask()

The functions **TSL_tkey_IsChanged()** or **TSL_linrot_IsChanged()** allows to check if a sensor state has changed.

You can also directly read the state inside the sensor data structure:

```
if MyTKeys[0].p_Data->StateId == TSL_STATEID_DETECT)
```

2.11.13 Enabling a specific state

It is possible to enter directly in the calibration, OFF and OFF with “burst only” states. The OFF with “burst only” state consists in only bursting the electrode without performing acquisition on it. It can be used in specific cases to improve the robustness against noise or to keep optimum sensor sensitivity.

This is done by using the following functions:

For touchkey sensor:

- TSL_tkey_SetStateCalibration()
- TSL_tkey_SetStateOff()
- TSL_tkey_SetStateBurstOnly()

For linear and rotary sensor:

- TSL_linrot_SetStateCalibration()
- TSL_linrot_SetStateOff()
- TSL_linrot_SetStateBurstOnly()

2.12 Environment Change System (ECS)

2.12.1 Principle

Power supply voltage, temperature and air humidity may induce a slow variation of the measured signal. The Environment Change System (ECS) is used to adapt the reference to these environment changes.

The ECS processing is based on an infinite response digital low pass filter of the first order (IIR filter):

$$Y(n) = K \times X(n) + (1 - K) \times Y(n - 1)$$

with:

Y = reference

X = acquisition value (last measurement)

K = coefficient.

The higher value is K, the faster is the response time. Two default K coefficients are available to obtain fast and slow responses.

The sampling frequency is programmable using a timing utility routine (see example below).

If the sensor is in PROX, DETECT or TOUCH states, the ECS is disabled for the duration of the detection timeout or for the duration of the touch (whichever ends first).

When the ECS is disabled, Yn=Yn-1

As soon as the recalibration times out or the detection ends, the filter is set active again.

2.12.2 Resources

The ECS functions are provided in the files:

- tsl_ecs.c
- tsl_ecs.h

The functions are:

- **TSL_ecs_Process()**: main function to be used by the user
- **TSL_ecs_CalcK()**: additional function
- **TSL_ecs_ProcessK()**: additional function

2.12.3 Parameters

- TSLPRM_ECS_K_FAST
- TSLPRM_ECS_K_SLOW
- TSLPRM_ECS_DELAY

2.12.4 Usage example

The ECS processing is usually performed in the main state machine at regular time intervals defined by the user. But it can be done also in interrupt routines. It must be performed after the sensors state machine is processed.

The ECS is activated only when all the sensors are in RELEASE, ERROR or OFF states, with at least one sensor in RELEASE state. It can also be delayed from milli-seconds to few seconds.

The ECS processing is performed on a group of sensors defined by the user. Different groups can be created and ECS can be applied on these groups with different K coefficients.

It is user’s choice to decide the best thing to do for his application.

The simplest way is to call the **TSL_ecs_Process()** function in the main application loop using the default K coefficients defined in the configuration file:

```
TSL_ecs_Process(&MyObjGroup);
```

To call this functions at regular time intervals you can use the provide timing routine **TSL_tim_CheckDelay_ms()**.

Example with ECS executed every 100ms:

```
TSL_tTick_ms_T time_ECS_tick;
int main(void) {
  while (1) {
    ...
    // ECS every 100 ms
    if (TSL_tim_CheckDelay_ms(100, &time_ECS_tick) == TSL_STATUS_OK)
    {
      TSL_ecs_Process(&MyObjGroup);
    }
    ...
  }
}
```

The **TSL_ecs_ProcessK()** function allows to use a K coefficient different than the default value:

```
if (TSL_tim_CheckDelay_ms(100, &time_ECS_tick) == TSL_STATUS_OK)
{
  if ((MyObjGroup->StateMask & TSL_STATE_RELEASE_BIT_MASK) &&
      !(MyObjGroup->StateMask & TSL_STATEMASK_ACTIVE))
  {
    TSL_ecs_ProcessK(&MyObjGroup, 120);
  }
}
```

2.13 Detection Exclusion System (DXS)

2.13.1 Principle

The DXS processing is used to prevent several sensors to be in the DETECT state at the same time. This could happen if the sensors are closed to each other or if their sensitivity is too high. This can be useful also in some applications to prevent the user to touch at the same time several sensors with “opposite” meaning (volume up and volume down for example).

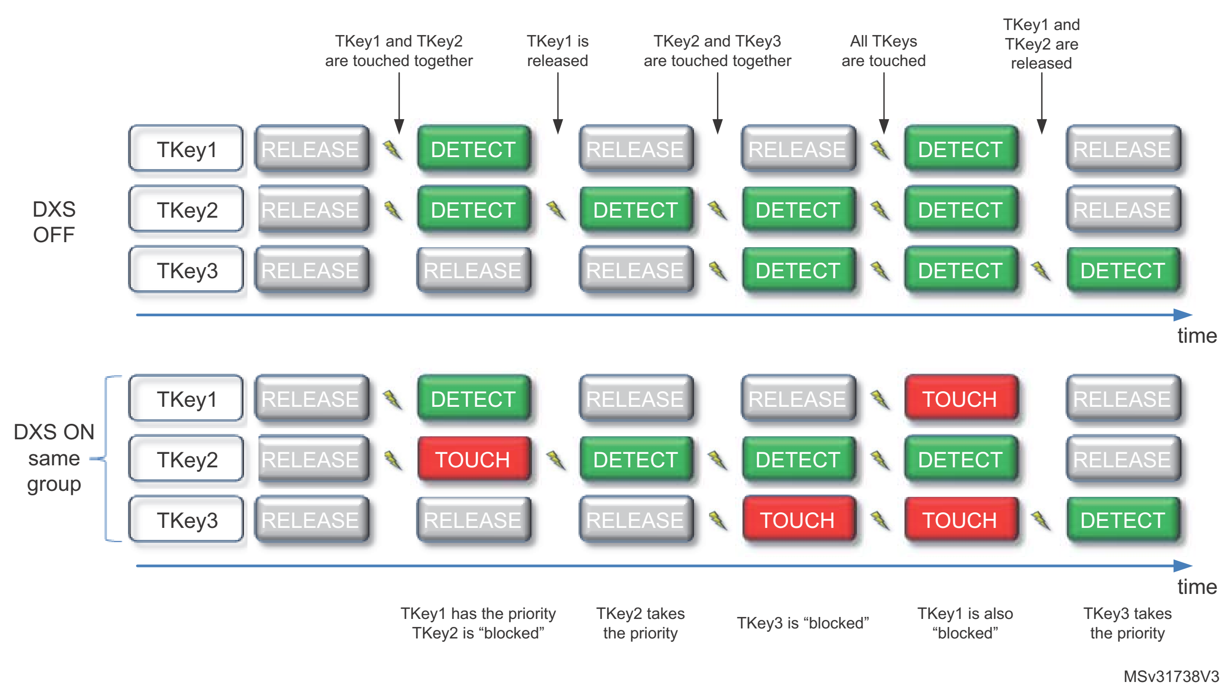
The first sensor in the group of sensors has the priority and enters in the DETECT state (with the DxSLock flag set). The other sensors are “blocked” and enter instead in the TOUCH state.

Note: A particular care must be taken when designing sensors that are shared between multiple DXS groups. The sensor that will be assigned in the DETECT state depends on the sensors position in the DXS groups and also on the order of the DXS groups processing. See the examples 1 and 2 for more detail.

The figure below illustrates the difference in behavior for a group of 3 sensors (touchkeys) when the DXS is OFF and ON. The three touchkeys are part of the same DXS group.

Note: The touchkeys can be replaced by a linear or a rotary sensor.

Figure 13. DXS principle



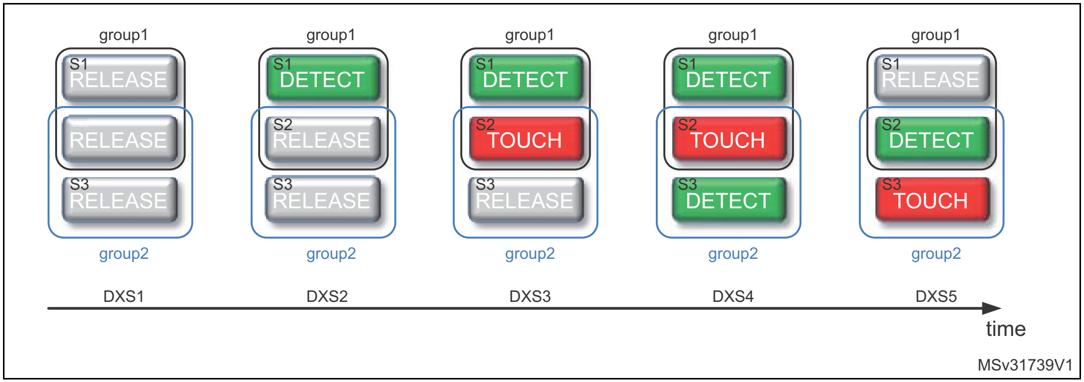
Example 1: 3 sensors with one shared between two groups.

In this example the group1 is composed of the two sensors s1 and s2 in this order and the group2 of the two sensors s2 and s3 in this order.

The DXS groups are processed in this order: group1 first and then group2.

We can see in the step DXS5 that the sensor 2 (s2) goes in DETECT state instead of the sensor 3 (s3). This is simply because s2 is placed first in the group2.

Figure 14. DXS example 1



Example 2: 4 sensors with one shared between three groups.

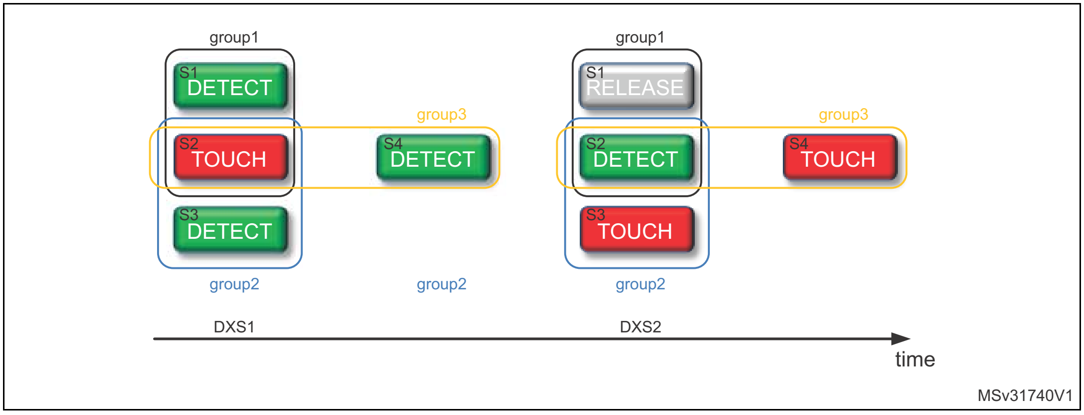
In this example the group1 is composed of the two sensors s1 and s2 in this order, the group2 of the two sensors s2 and s3 in this order and the group3 of the two sensors s2 and s4 in this order.

The DXS groups are processed in this order: group1 first, then group2 and finally group3.

We can see in the step DXS2 that the sensor 2 takes the priority over the sensors 3 and 4.

To summarize, the decision to be in DETECT state depends on the sensors placement inside the group and also on the order of the groups processing.

Figure 15. DXS example 2



2.13.2 Resources

The DXS functions are provided in the files:

- tsl_dxs.c
- tsl_dxs.h

The functions to use are:

- TSL_dxs_FirstObj()

2.13.3 Parameters

- TSLPRM_USE_DXS

2.13.4 Usage example

The DXS processing is performed usually in the main state machine but it can also be done in interrupt routines.

Warning: The DXS must be absolutely performed after the sensors state machine is processed, that is after the call to the TSL_obj_GroupProcess() function (see the main state machine for more details).

The DXS processing is performed on a **group of sensors** defined by the user. Different groups of DXS can be created.

It's up to the user to decide the best partitioning for his application.

Example:

```
int main(void) {
  while (1) {
    ...
    TSL_obj_GroupProcess(&MyObjGroup1);
    TSL_obj_GroupProcess(&MyObjGroup2);
    TSL_dxs_FirstObj(&MyObjGroup1);
    TSL_dxs_FirstObj(&MyObjGroup2);
    ...
  }
}
```

2.14 Detection Time Out (DTO)

2.14.1 Principle

The Detection Time Out (DTO) introduces a simple way to cope with water film and any obstacle that may come in contact with a sensor. It introduces a maximum duration for the 'detected' state of any sensor called the Detection Time Out (DTO).

After this period of time, the sensor is automatically recalibrated. This allows to make the sensor touch sensitive again, even if the obstacle or the liquid film is still present on the application front panel.

This feature is application dependent and the time out must be tuned according to the user interface specifications.

The DTO is applied on the PROX, DETECT and TOUCH states and can be disabled.

2.14.2 Resources

The DTO functions are provided in the files:

- tsl_touchkey.c
- tsl_touchkey.h
- tsl_linrot.c
- tsl_linrot.h

The functions used by the DTO are:

- TSL_tkey_DTOGetTime()
- TSL_linrot_DTOGetTime()
- TSL_tim_CheckDelay_sec()

Note: *The user doesn't need to call these functions to perform the DTO.*

2.14.3 Parameters

- TSLPRM_DTO

2.14.4 Usage

The DTO is automatically performed inside the sensor state machine. The user doesn't need to call any function in the application code.

The DTO is disabled by writing zero in the **TSLPRM_DTO** parameter.

2.15 Noise filters

2.15.1 Principle

The STMTouch touch sensing library has been designed to facilitate the implementation of different noise filters. These filters can be used for many purpose and can range from very simple design to very complicated.

2.15.2 Resources

The filters are defined in the files:

- tsl_filter.c
- tsl_filter.h

Each filter is described by a function:

- **TSL_filt_MeasFilter()**: filter on measurement values
- **TSL_filt_DeltaFilter()**: filter on delta values

2.15.3 Parameters

There is no parameter for the filter module.

2.15.4 Usage

The filter functions can be called at anytime in the main application. In order to speed-up the execution time and to gain RAM space, the measure and delta filters are called by the **TSL_acq_BankGetResult()** function.

Examples:

```
// Apply a filter on the measures only
TSL_acq_BankGetResult(0, TSL_filt_MeasFilter, 0);
// Get the measures without applying any filter
TSL_acq_BankGetResult(0, 0, 0);
```

Note: The user can also create his own filter functions.

2.16 Timing management

2.16.1 Principle

The STMTouch touch sensing library needs an internal clock (“timing”), in particular for the ECS and DTO processing.

The timing process consists to increment a global variable at a regular interval. Different functions are then used to compare the current “time” and to check if a certain delay has elapsed.

The Systick is used as timebase for the STMTouch touch sensing library. Its initialization must be done in the user code layer. Usually it is already done by the HAL_Init function. The TSLPRM_TICK_FREQ parameter must be set accordingly.

2.16.2 Resources

The common timing routines are described in the files:

- tsl_time.c
- tsl_time.h

Functions:

- TSL_tim_ProcessIT()
- TSL_tim_CheckDelay_ms()
- TSL_tim_CheckDelay_sec()

2.16.3 Parameters

- TSLPRM_TICK_FREQ: the value must be in line with the Systick frequency that is initialized in the user code.

2.16.4 Usage

The function **TSL_tim_CheckDelay_ms()** can be used in the main application code to execute some code (for example the ECS) at a regular interval.

Example:

```
TSL_tTick_ms_T time_ECS_tick;
TSL_tTick_ms_T time_LED_tick;
int main(void) {
  TSL_Init(MyBanks); // The timing starts...
  while (1) {
    ...
    // Launch the ECS every 100 ms
    if (TSL_tim_CheckDelay_ms(100, &time_ECS_tick) == TSL_STATUS_OK)
    {
      TSL_ecs_Process(&MyObjGroup);
    }
    // Toggle LED every 500 ms
    if (TSL_tim_CheckDelay_ms(500, &time_LED_tick) == TSL_STATUS_OK)
    {
      ToggleLED();
    }
    ...
  }
}
```

2.17 Parameters

All the parameters are described in the **tsl_conf.h** file.

*Note: The **tsl_conf_<XXX>_template.h** file present in the STM32_TouchSensing_Library/inc folder must be copied in the application project inc/tsl_conf.h and adapted to your application (number of channels, banks, debounce, DTO, etc.).*

The structure **TSL_Params_T** is used to hold certain parameters that are common to all sensors. These parameters can be changed by the user while the application is running.

Parameters checking

All common parameters are verified (presence and value range) in the file:

- tsl_check_config.h

All device specific parameters are verified in the **tsl_check_config_<XXX>.h** file.

3 Devices with TSC peripheral

This section concerns all STM32 microcontrollers that include the touch sensing controller peripheral (TSC).

3.1 Acquisition

The acquisition is done in the files:

- tsl_acq_tsc.c
- tsl_acq_tsc.h

Functions used by the application layer and that are device dependent:

- TSL_acq_BankConfig()
- TSL_acq_BankStartAcq()
- TSL_acq_BankWaitEOC()
- TSL_acq_GetMeas()

The other functions in this file are for internal use and the user doesn't need to call them directly.

The device selection must be done at the end of the tsl_conf.h file:

```
#include "stm32f0xx.h" /* Select the file corresponding to the device in use
(i.e. stm32f3xx.h, stm32f0xx.h, ...) */
```

3.2 Timings

The timing management is done in the files:

- tsl_time.c
- tsl_time.h

The **Systick** is used to generate a timebase for the ECS and DTO modules. It must be initialized in the user code (already done by the HAL_init function).

3.3 Parameters

The parameters are described in the file:

- tsl_conf_tsc_template.h (to be copied in the project and rename in **tsl_conf.h**)

and are checked in the file:

- tsl_check_config_tsc.h

3.4 MCU resources

The table below shows the peripherals that are used by the STMTouch touch sensing library on any STM32 microcontroller with the touch sensing controller. Care must be taken when using them to avoid any unwanted behavior.

Table 6. STM32F0 series MCU resources used

Peripheral	Function
GPIOs	Acquisition
Systick	Time base for ECS and DTO
Touch sensing controller (TSC)	Acquisition

3.5 STM32F0 series microcontrollers

3.5.1 Memory footprint

Conditions

- IAR ANSI C/C++ compiler/linker V7.40.3.8902 for ARM®
- Compiler optimization: high size
- Counted files: tsl*.o
- STM32 TouchSensing library options: ECS=ON, DTO=ON, DXS=OFF, PROX=OFF
- Each sensor has its own parameters placed in RAM

The following table summarizes the memory footprint with different configurations:

Table 7. STM32F0 series memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	3.0	100
2	1	2 TKeys	3.0	120
2	2	2 TKeys	3.0	120
24	3	24 TKeys	4.0	620
3	1	1 Linear-3ch	4.1	130
15	3	12 TKeys + 1 Linear-3ch	6.2	420
24	3	18 TKeys + 2 Linear-3ch	6.5	610

[1] The content of this table is provided for information purposes only.

3.5.2 Available touch sensing channels

The tables below provide an overview of the available touch sensing channels for the STM32F0 series microcontrollers.

Note: The following tables are not restrictive in term of part numbers supported by the STMTouch touch sensing library. The STMTouch touch sensing library can be used on any new device that may become available as part of ST microcontrollers portfolio. Please contact your ST representative for support.

Note: For n available pins in an I/O group, one pin is used as sampling capacitor and n-1 pins are used as channels.

The I/O group cannot be used if the number of available pins in less or equal to one.

Table 8. Available touch sensing channels for STM32F098xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F098Vx				STM32F098Rx						STM32F098Cx	
			UFBGA100		LQFP100		UFBGA64		LQFP64		WLCSP64		LQFP48 UFQFPN48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	2	x	2	x	2	x	2	x	2	-	1
	TSC_G3_IO2	PB0	x		x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x		x	
	TSC_G3_IO4	-	-		-		-		-		-		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x		x	

Table 8. Available touch sensing channels for STM32F098xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F098Vx				STM32F098Rx						STM32F098Cx	
			UFBGA100		LQFP100		UFBGA64		LQFP64		WLCSP64		LQFP48 UFQFPN48	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	0	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		x		-		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	x	0	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			23		23		17		17		17		16	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 9. Available touch sensing channels for STM32F091xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F091Vx				STM32F091Rx						STM32F091Cx	
			UFBGA100		LQFP100		UFBGA64		LQFP64		WLCSP64		LQFP48 UFQFPN48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x		x	
G2	TSC_G2_IO1	PA4(1)	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5(1)	x		x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	x	3	x	3	x	3	-	2
	TSC_G3_IO2	PB0	x		x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x		x	

Table 9. Available touch sensing channels for STM32F091xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F091Vx				STM32F091Rx						STM32F091Cx	
			UFBGA100		LQFP100		UFBGA64		LQFP64		WLCSP64		LQFP48 UFQFPN48	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	0	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		x		-		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	x	0	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		18		18		18		17	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 10. Available touch sensing channels for STM32F078xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F078Vx				STM32F078Rx		STM32F078Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x	
G2	TSC_G2_IO1	PA4(1)	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5(1)	x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	2	x	2	x	2	-	1	-	1
	TSC_G3_IO2	PB0	x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x	
	TSC_G3_IO4	-	-		-		-		-		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x	

Table 10. Available touch sensing channels for STM32F078xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F078Vx				STM32F078Rx		STM32F078Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		-		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			23		23		17		16		16	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 11. Available touch sensing channels for STM32F072xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F072Vx				STM32F072Rx		STM32F072Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	x	3	-	2	-	2
	TSC_G3_IO2	PB0	x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x	

Table 11. Available touch sensing channels for STM32F072xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F072Vx				STM32F072Rx		STM32F072Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		-		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		18		17		17	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 12. Available touch sensing channels for STM32F071xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F071Vx				STM32F071Rx		STM32F071Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	x	3	-	2	-	2
	TSC_G3_IO2	PB0	x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x	

Table 12. Available touch sensing channels for STM32F071xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F071Vx				STM32F071Rx		STM32F071Cx			
			UFBGA100		LQFP100		LQFP64		LQFP48 UFQFPN48		WLCSP49	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		-		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		18		17		17	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 13. Available touch sensing channels for STM32F058xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F058Rx				STM32F058Cx	
			LQFP64		UFBGA64		UFQFPN48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	

Table 13. Available touch sensing channels for STM32F058xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F058Rx				STM32F058Cx	
			LQFP64		UFBGA64		UFQFPN48	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	2	x	2	-	1
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1	x		x		x	
	TSC_G3_IO4	-	-		-		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA11	x		x		x	
	TSC_G4_IO4	PA12	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x	
	TSC_G6_IO3	PB13	x		x		x	
	TSC_G6_IO4	PB14	x		x		x	

Table 13. Available touch sensing channels for STM32F058xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F058Rx				STM32F058Cx	
			LQFP64		UFBGA64		UFQFPN48	
G7	TSC_G7_IO1	-	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-	
	TSC_G7_IO3	-	-		-		-	
	TSC_G7_IO4	-	-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-	
	TSC_G8_IO3	-	-		-		-	
	TSC_G8_IO4	-	-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			17		17		16	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 14. Available touch sensing channels for STM32F051xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F051Rx				STM32F051Cx		STM32F051Kx			
			LQFP64		UFBGA64		LQFP48 UFQFPN48		LQFP32		UFQFPN32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x	

Table 14. Available touch sensing channels for STM32F051xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F051Rx				STM32F051Cx		STM32F051Kx			
			LQFP64		UFBGA64		LQFP48 UFQFPN48		LQFP32		UFQFPN32	
G3	TSC_G3_IO1	PC5	x	3	x	3	-	2	-	1	-	2
	TSC_G3_IO2	PB0	x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		-		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x	
G6	TSC_G6_IO1	PB11	-	3	-	3	-	3	-	0	-	0
	TSC_G6_IO2	PB12	-		-		-		-		-	
	TSC_G6_IO3	PB13	-		-		-		-		-	
	TSC_G6_IO4	PB14	-		-		-		-		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-		-		-	
	TSC_G7_IO3	-	-		-		-		-		-	
	TSC_G7_IO4	-	-		-		-		-		-	

Table 14. Available touch sensing channels for STM32F051xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F051Rx				STM32F051Cx		STM32F051Kx			
			LQFP64		UFBGA64		LQFP48 UFQFPN48		LQFP32		UFQFPN32	
G8	TSC_G8_IO1	-	-	0	-	0	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-		-		-	
	TSC_G8_IO3	-	-		-		-		-		-	
	TSC_G8_IO4	-	-		-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			18		18		17		13		14	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 15. Available touch sensing channels for STM32F048xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F048Cx		STM32F048Tx		STM32F048Gx	
			UFQFPN48		WLCSP36		UFQFPN28	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	-	-	1	-	1	-	0
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1	x		x		-	
	TSC_G3_IO4	-	-		-		-	

Table 15. Available touch sensing channels for STM32F048xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F048Cx		STM32F048Tx		STM32F048Gx	
			UFQFPN48		WLCSP36		UFQFPN28	
G4	TSC_G4_IO1	PA9[2]	x	3	x	3	x[2]	1
	TSC_G4_IO2	PA10[2]	x		x		x[2]	
	TSC_G4_IO3	PA11[2]	x		x		x[2]	
	TSC_G4_IO4	PA12[2]	x		x		x[2]	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	-	-	0	-	0	-	0
	TSC_G6_IO2	-	-		-		-	
	TSC_G6_IO3	-	-		-		-	
	TSC_G6_IO4	-	-		-		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-	
	TSC_G7_IO3	-	-		-		-	
	TSC_G7_IO4	-	-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-	
	TSC_G8_IO3	-	-		-		-	
	TSC_G8_IO4	-	-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			13		13		10	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.
2. Pin pair PA11/PA12 can be remapped instead of pin pair PA9/PA10 using SYS_CTRL register (28 and 20 pins packages only).

Table 16. Available touch sensing channels for STM32F042xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F042Cx		STM32F042Tx		STM32F042Kx				STM32F042Gx		STM32F042Fx	
			LQFP48 UFQFPN48		WLCSP36		LQFP32		UFQFPN32		UFQFPN28		TSSOP20	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x		x	
G2	TSC_G2_IO1	PA4(1)	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5(1)	x		x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x		x	
G3	TSC_G3_IO1	-	-	2	-	2	-	1	-	2	-	1	-	0
	TSC_G3_IO2	PB0	x		x		x		x		x		-	
	TSC_G3_IO3	PB1	x		x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		-		x		-		-	
G4	TSC_G4_IO1	PA9(2)	x	3	x	3	x	3	x	3	x(2)	1	x(2)	1
	TSC_G4_IO2	PA10(2)	x		x		x		x		x(2)		x(2)	
	TSC_G4_IO3	PA11(2)	x		x		x		x		x(2)		x(2)	
	TSC_G4_IO4	PA12(2)	x		x		x		x		x(2)		x(2)	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3	-	0
	TSC_G5_IO2	PB4	x		x		x		x		x		-	
	TSC_G5_IO3	PB6	x		x		x		x		x		-	
	TSC_G5_IO4	PB7	x		x		x		x		x		-	

Table 16. Available touch sensing channels for STM32F042xx (continued)

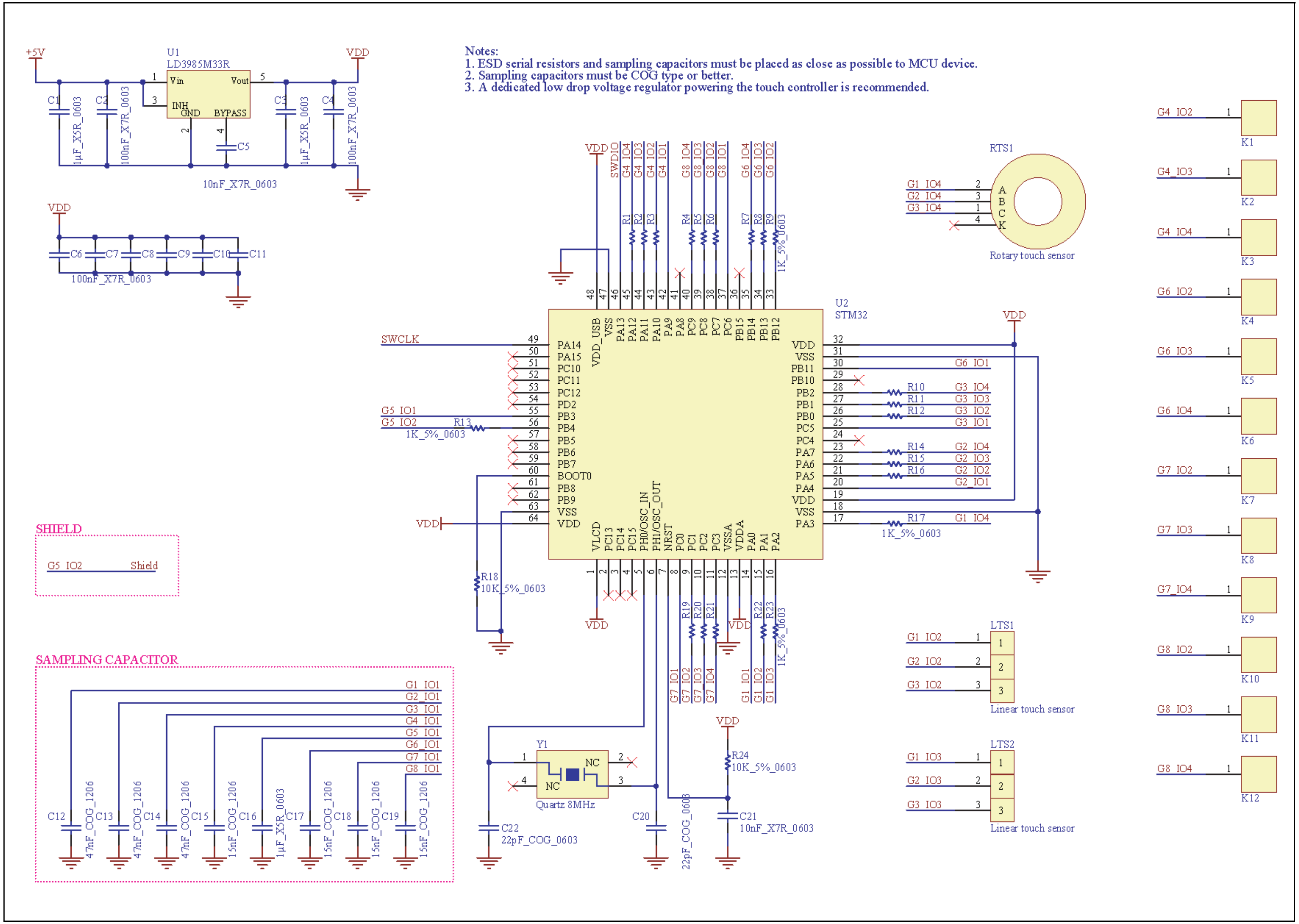
Analog I/O group	Capacitive sensing signal name	Pin name	STM32F042Cx		STM32F042Tx		STM32F042Kx				STM32F042Gx		STM32F042Fx	
			LQFP48 UFQFPN48		WLCSP36		LQFP32		UFQFPN32		UFQFPN28		TSSOP20	
G6	TSC_G6_IO1	-	-	0	-	0	-	0	-	0	-	0	-	0
	TSC_G6_IO2	-	-		-		-		-		-		-	
	TSC_G6_IO3	-	-		-		-		-		-		-	
	TSC_G6_IO4	-	-		-		-		-		-		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-		-		-		-	
	TSC_G7_IO3	-	-		-		-		-		-		-	
	TSC_G7_IO4	-	-		-		-		-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-		-		-		-	
	TSC_G8_IO3	-	-		-		-		-		-		-	
	TSC_G8_IO4	-	-		-		-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			14		14		13		14		11		7	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.
2. Pin pair PA11/PA12 can be remapped instead of pin pair PA9/PA10 using SYS_CTRL register (28 and 20 pins packages only).

3.5.3 Hardware implementation example

Figure 16 shows an example of hardware implementation on STM32F0 series microcontrollers.

Figure 16. STM32F0 series hardware implementation example



Notes:
1. ESD serial resistors and sampling capacitors must be placed as close as possible to MCU device.
2. Sampling capacitors must be COG type or better.
3. A dedicated low drop voltage regulator powering the touch controller is recommended.

3.6 STM32F3 series microcontrollers

3.6.1 Memory footprint

Conditions

- IAR ANSI C/C++ compiler/linker V7.40.3.8902 for ARM®
- Compiler optimization: high size
- Counted files: tsl*.o
- STM32 TouchSensing library options: ECS=ON, DTO=ON, DXS=OFF, PROX=OFF
- Each sensor has its own parameters placed in RAM

The following tables summarize the memory footprint with different configurations:

Table 17. STM32F3 series memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	2.8	100
2	1	2 TKeys	2.8	120
2	2	2 TKeys	2.8	120
24	3	24 TKeys	3.8	620
3	1	1 Linear-3ch	3.8	130
15	3	12 TKeys + 1 Linear-3ch	5.7	420
24	3	18 TKeys + 2 Linear-3ch	6.0	610

1. The content of this table is provided for information purposes only.

3.6.2 Available touch sensing channels

The tables below provide an overview of the available touch sensing channels for the STM32F3 series microcontrollers.

Note: The following tables are not restrictive in term of part numbers supported by the STMTouch touch sensing library. The STMTouch touch sensing library can be used on any new device that may become available as part of ST microcontrollers portfolio. Please contact your ST representative for support.

Note: For n available pins in an I/O group, one pin is used as sampling capacitor and n-1 pins are used as channels.

The I/O group cannot be used if the number of available pins in less or equal to one.

Table 18. Available touch sensing channels for STM32F398VE

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F398VE	
			LQFP100	
G1	TSC_G1_IO1	PA0	x	3
	TSC_G1_IO2	PA1	x	
	TSC_G1_IO3	PA2[1]	x	
	TSC_G1_IO4	PA3	x	
G2	TSC_G2_IO1	PA4[1]	x	3
	TSC_G2_IO2	PA5[1]	x	
	TSC_G2_IO3	PA6[1]	x	
	TSC_G2_IO4	PA7	x	
G3	TSC_G3_IO1	PC5	x	2
	TSC_G3_IO2	PB0	x	
	TSC_G3_IO3	PB1[1]	x	
	TSC_G3_IO4	-	-	
G4	TSC_G4_IO1	PA9	x	3
	TSC_G4_IO2	PA10	x	
	TSC_G4_IO3	PA13	x	
	TSC_G4_IO4	PA14	x	
G5	TSC_G5_IO1	PB3	x	3
	TSC_G5_IO2	PB4	x	
	TSC_G5_IO3	PB6	x	
	TSC_G5_IO4	PB7	x	
G6	TSC_G6_IO1	PB11	x	3
	TSC_G6_IO2	PB12[1]	x	
	TSC_G6_IO3	PB13	x	
	TSC_G6_IO4	PB14	x	
G7	TSC_G7_IO1	PE2	x	3
	TSC_G7_IO2	PE3	x	
	TSC_G7_IO3	PE4	x	
	TSC_G7_IO4	PE5	x	
G8	TSC_G8_IO1	PD12	x	3
	TSC_G8_IO2	PD13	x	
	TSC_G8_IO3	PD14	x	
	TSC_G8_IO4	PD15	x	
Number of capacitive sensing channels (sampling I/Os not counted)			23	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 19. Available touch sensing channels for STM32F378xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F378Vx				STM32F378Rx				STM32F378Cx	
			LQFP100		BGA100		LQFP64		WLCSP66		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x	
G2	TSC_G2_IO1	PA4[(1)]	x	3	x	3	x	3	x	3	x	2
	TSC_G2_IO2	PA5[(1)]	x		x		x		x		x	
	TSC_G2_IO3	PA6[(1)]	x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		-	
G3	TSC_G3_IO1	PC4	x	3	x	3	x	3	x	3	-	1
	TSC_G3_IO2	PC5	x		x		x		x		-	
	TSC_G3_IO3	PB0	x		x		x		x		x	
	TSC_G3_IO4	PB1	x		x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x	
G6	TSC_G6_IO1	PB14	x	3	x	3	x	2	x	2	x	2
	TSC_G6_IO2	PB15	x		x		x		x		x	
	TSC_G6_IO3	PD8	x		x		x		x		x	
	TSC_G6_IO4	PD9	x		x		-		-		-	
G7	TSC_G7_IO1	PE2	x	3	x	3	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		-		-		-	

Table 19. Available touch sensing channels for STM32F378xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F378Vx				STM32F378Rx				STM32F378Cx	
			LQFP100		BGA100		LQFP64		WLCSP66		LQFP48	
G8	TSC_G8_IO1	PD12	x	3	x	3	-	0	-	0	-	0
	TSC_G8_IO2	PD13	x		x		-		-		-	
	TSC_G8_IO3	PD14	x		x		-		-		-	
	TSC_G8_IO4	PD15	x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		17		17		14	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 20. Available touch sensing channels for STM32F373xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F373Vx				STM32F373Rx		STM32F373Cx	
			LQFP100		BGA100		LQFP64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	2
	TSC_G2_IO2	PA5[1]	x		x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		-	
G3	TSC_G3_IO1	PC4	x	3	x	3	x	3	-	1
	TSC_G3_IO2	PC5	x		x		x		-	
	TSC_G3_IO3	PB0	x		x		x		x	
	TSC_G3_IO4	PB1	x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x	

Table 20. Available touch sensing channels for STM32F373xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F373Vx				STM32F373Rx		STM32F373Cx	
			LQFP100		BGA100		LQFP64		LQFP48	
G6	TSC_G6_IO1	PB14	x	3	x	3	x	2	x	2
	TSC_G6_IO2	PB15	x		x		x		x	
	TSC_G6_IO3	PD8	x		x		x		x	
	TSC_G6_IO4	PD9	x		x		-		-	
G7	TSC_G7_IO1	PE2	x	3	x	3	-	0	-	0
	TSC_G7_IO2	PE3	x		x		-		-	
	TSC_G7_IO3	PE4	x		x		-		-	
	TSC_G7_IO4	PE5	x		x		-		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	-	0	-	0
	TSC_G8_IO2	PD13	x		x		-		-	
	TSC_G8_IO3	PD14	x		x		-		-	
	TSC_G8_IO4	PD15	x		x		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		17		14	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 21. Available touch sensing channels for STM32F358xC

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F358Vx		STM32F358Rx		STM32F358Cx	
			LQFP100		LQFP64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	2	x	2	-	1
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x	
	TSC_G3_IO4	-	-		-		-	

Table 21. Available touch sensing channels for STM32F358xC (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F358Vx		STM32F358Rx		STM32F358Cx	
			LQFP100		LQFP64		LQFP48	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA13	x		x		x	
	TSC_G4_IO4	PA14	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3
	TSC_G6_IO2	PB12[1]	x		x		x	
	TSC_G6_IO3	PB13	x		x		x	
	TSC_G6_IO4	PB14	x		x		x	
G7	TSC_G7_IO1	PE2	x	3	-	0	-	0
	TSC_G7_IO2	PE3	x		-		-	
	TSC_G7_IO3	PE4	x		-		-	
	TSC_G7_IO4	PE5	x		-		-	
G8	TSC_G8_IO1	PD12	x	3	-	0	-	0
	TSC_G8_IO2	PD13	x		-		-	
	TSC_G8_IO3	PD14	x		-		-	
	TSC_G8_IO4	PD15	x		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			23		17		16	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 22. Available touch sensing channels for STM32F334x4/x6/x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F334Rx		STM32F334Cx		STM32F334Kx	
			LQFP64		LQFP48		LQFP32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	

Table 22. Available touch sensing channels for STM32F334x4/x6/x8 (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F334Rx		STM32F334Cx		STM32F334Kx	
			LQFP64		LQFP48		LQFP32	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	3	-	2	-	1
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1	x		x		x	
	TSC_G3_IO4	PB2	x		x		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA13	x		x		x	
	TSC_G4_IO4	PA14	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	-	0
	TSC_G6_IO2	PB12	x		x		-	
	TSC_G6_IO3	PB13	x		x		-	
	TSC_G6_IO4	PB14	x		x		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-	
	TSC_G7_IO3	-	-		-		-	
	TSC_G7_IO4	-	-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-	
	TSC_G8_IO3	-	-		-		-	
	TSC_G8_IO4	-	-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			18		17		13	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 23. Available touch sensing channels for STM32F328C8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F328C8	
			LQFP48	
G1	TSC_G1_IO1	PA0	x	3
	TSC_G1_IO2	PA1	x	
	TSC_G1_IO3	PA2	x	
	TSC_G1_IO4	PA3	x	
G2	TSC_G2_IO1	PA4[1]	x	3
	TSC_G2_IO2	PA5[1]	x	
	TSC_G2_IO3	PA6[1]	x	
	TSC_G2_IO4	PA7	x	
G3	TSC_G3_IO1	-	-	1
	TSC_G3_IO2	PB0	x	
	TSC_G3_IO3	PB1	x	
	TSC_G3_IO4	-	-	
G4	TSC_G4_IO1	PA9	x	3
	TSC_G4_IO2	PA10	x	
	TSC_G4_IO3	PA13	x	
	TSC_G4_IO4	PA14	x	
G5	TSC_G5_IO1	PB3	x	3
	TSC_G5_IO2	PB4	x	
	TSC_G5_IO3	PB6	x	
	TSC_G5_IO4	PB7	x	
G6	TSC_G6_IO1	PB11	x	3
	TSC_G6_IO2	PB12	x	
	TSC_G6_IO3	PB13	x	
	TSC_G6_IO4	PB14	x	
G7	TSC_G7_IO1	-	-	0
	TSC_G7_IO2	-	-	
	TSC_G7_IO3	-	-	
	TSC_G7_IO4	-	-	
G8	TSC_G8_IO1	-	-	0
	TSC_G8_IO2	-	-	
	TSC_G8_IO3	-	-	
	TSC_G8_IO4	-	-	
Number of capacitive sensing channels (sampling I/Os not counted)			16	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 24. Available touch sensing channels for STM32F318x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F318C8		STM32F318K8	
			WLCSP49		UQFN32	
G1	TSC_G1_IO1	PA0	x	3	x	3
	TSC_G1_IO2	PA1	x		x	
	TSC_G1_IO3	PA2[1]	x		x	
	TSC_G1_IO4	PA3	x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x	
	TSC_G2_IO3	PA6[1]	x		x	
	TSC_G2_IO4	PA7	x		x	
G3	TSC_G3_IO1	-	-	1	-	0
	TSC_G3_IO2	PB0	x		x	
	TSC_G3_IO3	PB1	x		-	
	TSC_G3_IO4	-	-		-	
G4	TSC_G4_IO1	PA9	x	3	x	3
	TSC_G4_IO2	PA10	x		x	
	TSC_G4_IO3	PA13	x		x	
	TSC_G4_IO4	PA14	x		x	
G5	TSC_G5_IO1	PB3	x	3	x	2
	TSC_G5_IO2	PB4	x		x	
	TSC_G5_IO3	PB6	x		x	
	TSC_G5_IO4	PB7	x		-	
G6	TSC_G6_IO1	PB11	x	3	-	0
	TSC_G6_IO2	PB12	x		-	
	TSC_G6_IO3	PB13	x		-	
	TSC_G6_IO4	PB14	x		-	
G7	TSC_G7_IO1	-	-	0	-	0
	TSC_G7_IO2	-	-		-	
	TSC_G7_IO3	-	-		-	
	TSC_G7_IO4	-	-		-	
G8	TSC_G8_IO1	-	-	0	-	0
	TSC_G8_IO2	-	-		-	
	TSC_G8_IO3	-	-		-	
	TSC_G8_IO4	-	-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			16		11	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 25. Available touch sensing channels for STM32F303xD/xE

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Zx		STM32F303Vx				STM32F303Rx	
			LQFP144		LQFP100		UFBGA100		LQFP64	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	x	3	x	3
	TSC_G3_IO2	PB0	x		x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12[1]	x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	3	-	0
	TSC_G7_IO2	PE3	x		x		x		-	
	TSC_G7_IO3	PE4	x		x		x		-	
	TSC_G7_IO4	PE5	x		x		x		-	

Table 25. Available touch sensing channels for STM32F303xD/xE (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Zx		STM32F303Vx				STM32F303Rx	
			LQFP144		LQFP100		UFBGA100		LQFP64	
G8	TSC_G8_IO1	PD12	x	3	x	3	x	3	-	0
	TSC_G8_IO2	PD13	x		x		x		-	
	TSC_G8_IO3	PD14	x		x		x		-	
	TSC_G8_IO4	PD15	x		x		x		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		24		18	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 26. Available touch sensing channels for STM32F303xB/xC

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Vx		STM32F303Rx		STM32F303Cx	
			LQFP100		LQFP64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	-	2
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x	
	TSC_G3_IO4	PB2	x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA13	x		x		x	
	TSC_G4_IO4	PA14	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	

Table 26. Available touch sensing channels for STM32F303xB/xC (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Vx		STM32F303Rx		STM32F303Cx	
			LQFP100		LQFP64		LQFP48	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3
	TSC_G6_IO2	PB12(1)	x		x		x	
	TSC_G6_IO3	PB13	x		x		x	
	TSC_G6_IO4	PB14	x		x		x	
G7	TSC_G7_IO1	PE2	x	3	-	0	-	0
	TSC_G7_IO2	PE3	x		-		-	
	TSC_G7_IO3	PE4	x		-		-	
	TSC_G7_IO4	PE5	x		-		-	
G8	TSC_G8_IO1	PD12	x	3	-	0	-	0
	TSC_G8_IO2	PD13	x		-		-	
	TSC_G8_IO3	PD14	x		-		-	
	TSC_G8_IO4	PD15	x		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		18		17	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 27. Available touch sensing channels for STM32F303x6/x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Rx		STM32F303Cx		STM32F303Kx	
			LQFP64		LQFP48		LQFP32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2(1)	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4(1)	x	3	x	3	x	3
	TSC_G2_IO2	PA5(1)	x		x		x	
	TSC_G2_IO3	PA6(1)	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	3	-	2	-	0
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1(1)	x		x		-	
	TSC_G3_IO4	PB2	x		x		-	

Table 27. Available touch sensing channels for STM32F303x6/x8 (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F303Rx		STM32F303Cx		STM32F303Kx	
			LQFP64		LQFP48		LQFP32	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA13	x		x		x	
	TSC_G4_IO4	PA14	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	-	0
	TSC_G6_IO2	PB12[1]	x		x		-	
	TSC_G6_IO3	PB13	x		x		-	
	TSC_G6_IO4	PB14	x		x		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-	
	TSC_G7_IO3	-	-		-		-	
	TSC_G7_IO4	-	-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-	
	TSC_G8_IO3	-	-		-		-	
	TSC_G8_IO4	-	-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			18		17		12	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 28. Available touch sensing channels for STM32F302xD/xE

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F302Zx		STM32F302Vx				STM32F302Rx	
			LQFP144		LQFP100		UFBGA100		LQFP64	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x	

Table 28. Available touch sensing channels for STM32F302xD/xE (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F302Zx		STM32F302Vx				STM32F302Rx	
			LQFP144		LQFP100		UFBGA100		LQFP64	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	x	3	x	3
	TSC_G3_IO2	PB0	x		x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	3
	TSC_G6_IO2	PB12[1]	x		x		x		x	
	TSC_G6_IO3	PB13	x		x		x		x	
	TSC_G6_IO4	PB14	x		x		x		x	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	3	-	0
	TSC_G7_IO2	PE3	x		x		x		-	
	TSC_G7_IO3	PE4	x		x		x		-	
	TSC_G7_IO4	PE5	x		x		x		-	
G8	TSC_G8_IO1	PD12	x	3	x	3	x	3	-	0
	TSC_G8_IO2	PD13	x		x		x		-	
	TSC_G8_IO3	PD14	x		x		x		-	
	TSC_G8_IO4	PD15	x		x		x		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		24		18	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 29. Available touch sensing channels for STM32F302xB/xC

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F302Vx		STM32F302Rx		STM32F302Cx	
			LQFP100		LQFP64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	-	2
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x	
	TSC_G3_IO4	PB2	x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA13	x		x		x	
	TSC_G4_IO4	PA14	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3
	TSC_G6_IO2	PB12[1]	x		x		x	
	TSC_G6_IO3	PB13	x		x		x	
	TSC_G6_IO4	PB14	x		x		x	
G7	TSC_G7_IO1	PE2	x	3	-	0	-	0
	TSC_G7_IO2	PE3	x		-		-	
	TSC_G7_IO3	PE4	x		-		-	
	TSC_G7_IO4	PE5	x		-		-	
G8	TSC_G8_IO1	PD12	x	3	-	0	-	0
	TSC_G8_IO2	PD13	x		-		-	
	TSC_G8_IO3	PD14	x		-		-	
	TSC_G8_IO4	PD15	x		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		18		17	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 30. Available touch sensing channels for STM32F302x6/x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32F302Rx		STM32F302Cx				STM32F302Kx	
			LQFP64		LQFP48		WLCSP49		UQFN32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	-	2	-	2	-	0
	TSC_G3_IO2	PB0	x		x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x		-	
	TSC_G3_IO4	PB2	x		x		x		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	-	3	-	0
	TSC_G6_IO2	PB12[1]	x		x		-		-	
	TSC_G6_IO3	PB13	x		x		-		-	
	TSC_G6_IO4	PB14	x		x		-		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-		-	
	TSC_G7_IO3	-	-		-		-		-	
	TSC_G7_IO4	-	-		-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-		-	
	TSC_G8_IO3	-	-		-		-		-	
	TSC_G8_IO4	-	-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			18		17		17		12	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 31. Available touch sensing channels for STM32F301x6/x8

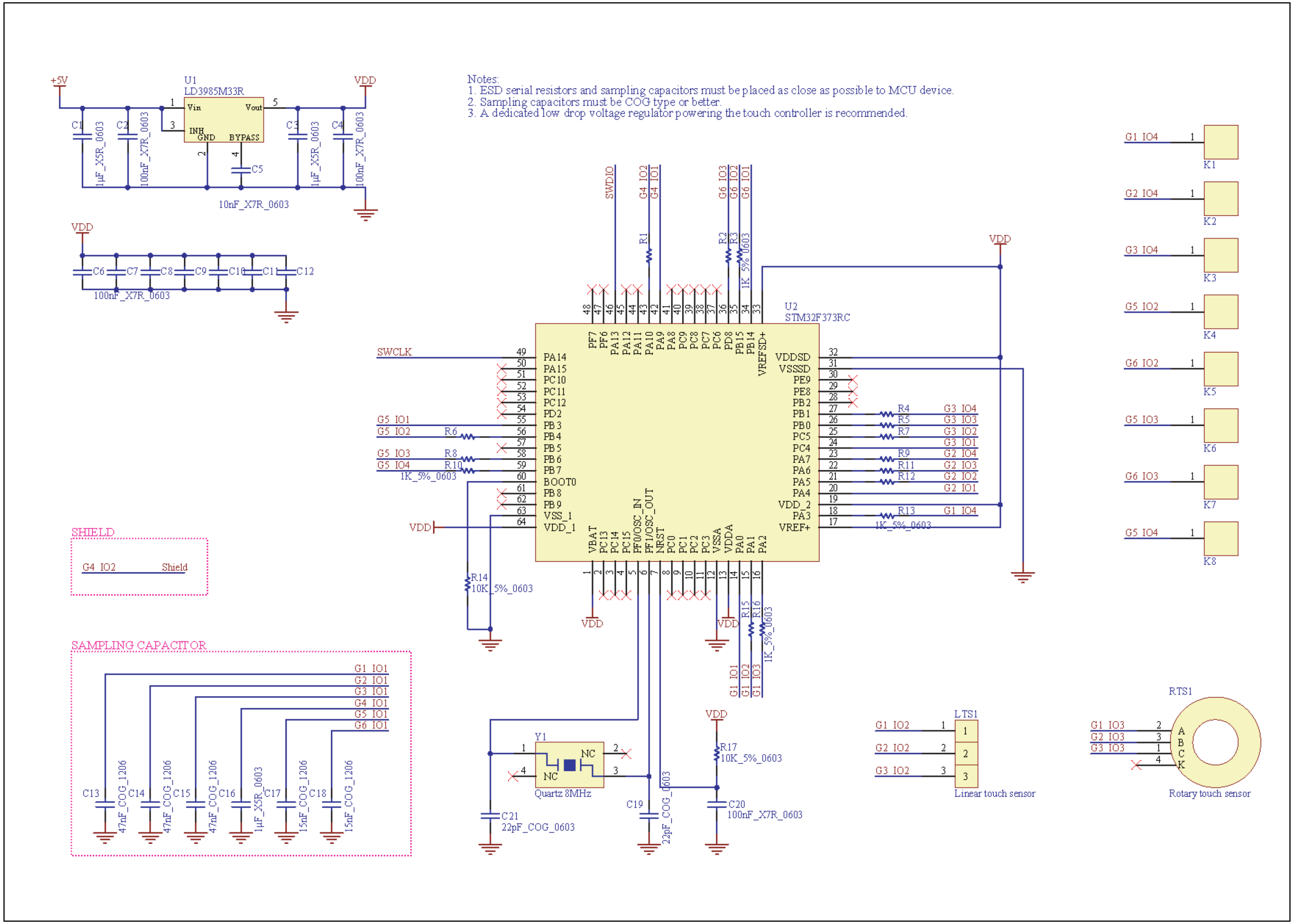
Analog I/O group	Capacitive sensing signal name	Pin name	STM32F301Rx		STM32F301Cx				STM32F301Kx	
			LQFP64		LQFP48		WLCSP49		UQFN32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x	
	TSC_G1_IO3	PA2[1]	x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5[1]	x		x		x		x	
	TSC_G2_IO3	PA6[1]	x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	-	2	-	2	-	0
	TSC_G3_IO2	PB0	x		x		x		x	
	TSC_G3_IO3	PB1[1]	x		x		x		-	
	TSC_G3_IO4	PB2	x		x		x		-	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x	
	TSC_G4_IO3	PA13	x		x		x		x	
	TSC_G4_IO4	PA14	x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	-	3	-	0
	TSC_G6_IO2	PB12[1]	x		x		-		-	
	TSC_G6_IO3	PB13	x		x		-		-	
	TSC_G6_IO4	PB14	x		x		-		-	
G7	TSC_G7_IO1	-	-	0	-	0	-	0	-	0
	TSC_G7_IO2	-	-		-		-		-	
	TSC_G7_IO3	-	-		-		-		-	
	TSC_G7_IO4	-	-		-		-		-	
G8	TSC_G8_IO1	-	-	0	-	0	-	0	-	0
	TSC_G8_IO2	-	-		-		-		-	
	TSC_G8_IO3	-	-		-		-		-	
	TSC_G8_IO4	-	-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			18		17		17		12	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

3.6.3 Hardware implementation example

Figure 17 shows an example of hardware implementation on STM32F3 series microcontrollers.

Figure 17. STM32F3 series hardware implementation example



Notes:
1. ESD serial resistors and sampling capacitors must be placed as close as possible to MCU device.
2. Sampling capacitors must be COG type or better.
3. A dedicated low drop voltage regulator powering the touch controller is recommended.

3.7 STM32L0 series microcontrollers

3.7.1 Memory footprint

Conditions

- IAR ANSI C/C++ compiler/linker V7.40.3.8902 for ARM®
- Compiler optimization: high size
- Counted files: tsl*.o
- STM32 TouchSensing library options: ECS=ON, DTO=ON, DXS=OFF, PROX=OFF
- Each sensor has its own parameters placed in RAM

The following table summarize the memory footprint with different configurations:

Table 32. STM32L0 series memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	3.0	100
2	1	2 TKeys	3.0	120
2	2	2 TKeys	3.0	120
24	3	24 TKeys	4.0	620
3	1	1 Linear-3ch	4.1	130
15	3	12 TKeys + 1 Linear-3ch	6.2	420
24	3	18 TKeys + 2 Linear-3ch	6.5	610

[1] The content of this table is provided for information purposes only.

3.7.2 Available touch sensing channels

The tables below provide an overview of the available touch sensing channels for the STM32L0 series microcontrollers.

Note: *The following tables are not restrictive in term of part numbers supported by the STMTouch touch sensing library. The STMTouch touch sensing library can be used on any new device that may become available as part of ST microcontrollers portfolio. Please contact your ST representative for support.*

Note: *For n available pins in an I/O group, one pin is used as sampling capacitor and n-1 pins are used as channels.*

The I/O group cannot be used if the number of available pins in less or equal to one.

Table 33. Available touch sensing channels for STM32L063x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L063R8		STM32L063C8	
			LQFP64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3
	TSC_G1_IO2	PA1	x		x	
	TSC_G1_IO3	PA2	x		x	
	TSC_G1_IO4	PA3	x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3
	TSC_G2_IO2	PA5	x		x	
	TSC_G2_IO3	PA6	x		x	
	TSC_G2_IO4	PA7	x		x	
G3	TSC_G3_IO1	PC5	x	3	-	2
	TSC_G3_IO2	PB0	x		x	
	TSC_G3_IO3	PB1	x		x	
	TSC_G3_IO4	PB2	x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3
	TSC_G4_IO2	PA10	x		x	
	TSC_G4_IO3	PA11	x		x	
	TSC_G4_IO4	PA12	x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3
	TSC_G5_IO2	PB4	x		x	
	TSC_G5_IO3	PB6	x		x	
	TSC_G5_IO4	PB7	x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3
	TSC_G6_IO2	PB12	x		x	
	TSC_G6_IO3	PB13	x		x	
	TSC_G6_IO4	PB14	x		x	
G7	TSC_G7_IO1	PC0	x	3	-	0
	TSC_G7_IO2	PC1	x		-	
	TSC_G7_IO3	PC2	x		-	
	TSC_G7_IO4	PC3	x		-	
G8	TSC_G8_IO1	PC6	x	3	-	0
	TSC_G8_IO2	PC7	x		-	
	TSC_G8_IO3	PC8	x		-	
	TSC_G8_IO4	PC9	x		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		17	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 34. Available touch sensing channels for STM32L062K8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L062K8	
			UFQFPN32	
G1	TSC_G1_IO1	PA0	x	3
	TSC_G1_IO2	PA1	x	
	TSC_G1_IO3	PA2	x	
	TSC_G1_IO4	PA3	x	
G2	TSC_G2_IO1	PA4[1]	x	3
	TSC_G2_IO2	PA5	x	
	TSC_G2_IO3	PA6	x	
	TSC_G2_IO4	PA7	x	
G3	TSC_G3_IO1	PC5	-	2
	TSC_G3_IO2	PB0	x	
	TSC_G3_IO3	PB1	x	
	TSC_G3_IO4	PB2	x	
G4	TSC_G4_IO1	PA9	x	3
	TSC_G4_IO2	PA10	x	
	TSC_G4_IO3	PA11	x	
	TSC_G4_IO4	PA12	x	
G5	TSC_G5_IO1	PB3	x	3
	TSC_G5_IO2	PB4	x	
	TSC_G5_IO3	PB6	x	
	TSC_G5_IO4	PB7	x	
G6	TSC_G6_IO1	PB11	-	0
	TSC_G6_IO2	PB12	-	
	TSC_G6_IO3	PB13	-	
	TSC_G6_IO4	PB14	-	
G7	TSC_G7_IO1	PC0	-	0
	TSC_G7_IO2	PC1	-	
	TSC_G7_IO3	PC2	-	
	TSC_G7_IO4	PC3	-	
G8	TSC_G8_IO1	PC6	-	0
	TSC_G8_IO2	PC7	-	
	TSC_G8_IO3	PC8	-	
	TSC_G8_IO4	PC9	-	
Number of capacitive sensing channels (sampling I/Os not counted)			14	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 35. Available touch sensing channels for STM32L053x6/x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L053Rx		STM32L053Rx		STM32L053Cx	
			LQFP64		TFBGA64		LQFP48	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x	
	TSC_G1_IO3	PA2	x		x		x	
	TSC_G1_IO4	PA3	x		x		x	
G2	TSC_G2_IO1	PA4[1]	x	3	x	3	x	3
	TSC_G2_IO2	PA5	x		x		x	
	TSC_G2_IO3	PA6	x		x		x	
	TSC_G2_IO4	PA7	x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	-	2
	TSC_G3_IO2	PB0	x		x		x	
	TSC_G3_IO3	PB1	x		x		x	
	TSC_G3_IO4	PB2	x		x		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x	
	TSC_G4_IO3	PA11	x		x		x	
	TSC_G4_IO4	PA12	x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x	
	TSC_G5_IO3	PB6	x		x		x	
	TSC_G5_IO4	PB7	x		x		x	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3
	TSC_G6_IO2	PB12	x		x		x	
	TSC_G6_IO3	PB13	x		x		x	
	TSC_G6_IO4	PB14	x		x		x	
G7	TSC_G7_IO1	PC0	x	3	x	2	-	0
	TSC_G7_IO2	PC1	x		x		-	
	TSC_G7_IO3	PC2	x		x		-	
	TSC_G7_IO4	PC3	x		-		-	
G8	TSC_G8_IO1	PC6	x	3	x	3	-	0
	TSC_G8_IO2	PC7	x		x		-	
	TSC_G8_IO3	PC8	x		x		-	
	TSC_G8_IO4	PC9	x		x		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		23		17	

[1] This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

Table 36. Available touch sensing channels for STM32L052x6/x8

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L052Rx		STM32L052Rx		STM32L052Cx		STM32L052Tx		STM32L052Kx			
			LQFP64		TFBGA64		LQFP48		WLCSP36		LQFP32		UFQFPN32	
G1	TSC_G1_IO1	PA0	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PA1	x		x		x		x		x		x	
	TSC_G1_IO3	PA2	x		x		x		x		x		x	
	TSC_G1_IO4	PA3	x		x		x		x		x		x	
G2	TSC_G2_IO1	PA4(1)	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PA5	x		x		x		x		x		x	
	TSC_G2_IO3	PA6	x		x		x		x		x		x	
	TSC_G2_IO4	PA7	x		x		x		x		x		x	
G3	TSC_G3_IO1	PC5	x	3	x	3	-	2	-	2	-	1	-	2
	TSC_G3_IO2	PB0	x		x		x		x		x		x	
	TSC_G3_IO3	PB1	x		x		x		x		x		x	
	TSC_G3_IO4	PB2	x		x		x		x		-		x	
G4	TSC_G4_IO1	PA9	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PA10	x		x		x		x		x		x	
	TSC_G4_IO3	PA11	x		x		x		x		x		x	
	TSC_G4_IO4	PA12	x		x		x		x		x		x	
G5	TSC_G5_IO1	PB3	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G5_IO2	PB4	x		x		x		x		x		x	
	TSC_G5_IO3	PB6	x		x		x		x		x		x	
	TSC_G5_IO4	PB7	x		x		x		x		x		x	

Table 36. Available touch sensing channels for STM32L052x6/x8 (continued)

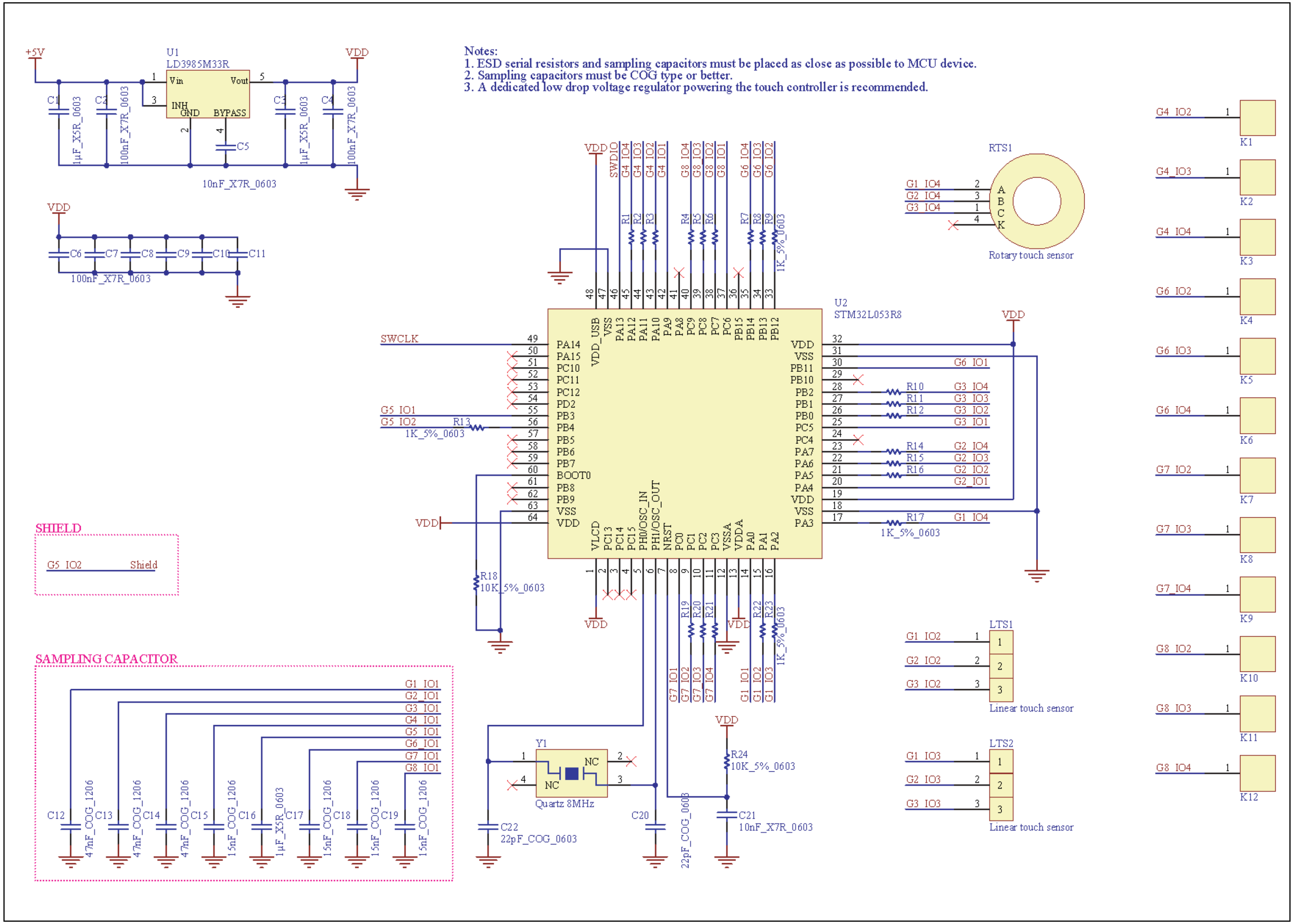
Analog I/O group	Capacitive sensing signal name	Pin name	STM32L052Rx		STM32L052Rx		STM32L052Cx		STM32L052Tx		STM32L052Kx			
			LQFP64		TFBGA64		LQFP48		WLCSP36		LQFP32		UFQFPN32	
G6	TSC_G6_IO1	PB11	x	3	x	3	x	3	x	0	-	0	-	0
	TSC_G6_IO2	PB12	x		x		x		-		-		-	
	TSC_G6_IO3	PB13	x		x		x		-		-		-	
	TSC_G6_IO4	PB14	x		x		x		-		-		-	
G7	TSC_G7_IO1	PC0	x	3	x	2	-	0	-	0	-	0	-	0
	TSC_G7_IO2	PC1	x		x		-		-		-		-	
	TSC_G7_IO3	PC2	x		x		-		-		-		-	
	TSC_G7_IO4	PC3	x		-		-		-		-		-	
G8	TSC_G8_IO1	PC6	x	3	x	3	-	0	-	0	-	0	-	0
	TSC_G8_IO2	PC7	x		x		-		-		-		-	
	TSC_G8_IO3	PC8	x		x		-		-		-		-	
	TSC_G8_IO4	PC9	x		x		-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		23		17		14		13		14	

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

3.7.3 Hardware implementation example

Figure 18 shows an example of hardware implementation on STM32L0 series microcontrollers.

Figure 18. STM32L0 series hardware implementation example



Notes:
1. ESD serial resistors and sampling capacitors must be placed as close as possible to MCU device.
2. Sampling capacitors must be COG type or better.
3. A dedicated low drop voltage regulator powering the touch controller is recommended.

3.8 STM32L4 series microcontrollers

3.8.1 Memory footprint

Conditions

- IAR ANSI C/C++ compiler/linker V7.40.3.8902 for ARM®
- Compiler optimization: high size
- Counted files: tsl*.o
- STM32 TouchSensing library options: ECS=ON, DTO=ON, DXS=OFF, PROX=OFF
- Each sensor has its own parameters placed in RAM

The following table summarizes the memory footprint with different configurations:

Table 37. STM32L4 series memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	2.8	100
2	1	2 TKeys	2.8	120
2	2	2 TKeys	2.8	120
24	3	24 TKeys	3.8	620
3	1	1 Linear-3ch	3.8	130
15	3	12 TKeys + 1 Linear-3ch	5.7	420
24	3	18 TKeys + 2 Linear-3ch	6.0	610

[1] The content of this table is provided for information purposes only.

3.8.2 Available touch sensing channels

The tables below provide an overview of the available touch sensing channels for the STM32L4 series microcontrollers.

Note: The following tables are not restrictive in term of part numbers supported by the STMTouch touch sensing library. The STMTouch touch sensing library can be used on any new device that may become available as part of ST microcontrollers portfolio. Please contact your ST representative for support.

Note: For n available pins in an I/O group, one pin is used as sampling capacitor and n-1 pins are used as channels.

The I/O group cannot be used if the number of available pins in less or equal to one.

Table 38. Available touch sensing channels for STM32L486xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L486Zx		STM32L486Qx		STM32L486Vx		STM32L486Jx		STM32L486Rx	
			LQFP144		UFBGA132		LQFP100		WLCSP72		LQFP64	
G1	TSC_G1_IO1	PB12	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PB13	x		x		x		x		x	
	TSC_G1_IO3	PB14	x		x		x		x		x	
	TSC_G1_IO4	PB15	x		x		x		x		x	
G2	TSC_G2_IO1	PB4	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PB5	x		x		x		x		x	
	TSC_G2_IO3	PB6	x		x		x		x		x	
	TSC_G2_IO4	PB7	x		x		x		x		x	
G3	TSC_G3_IO1	PA15	x	3	x	3	x	3	x	3	x	3
	TSC_G3_IO2	PC10	x		x		x		x		x	
	TSC_G3_IO3	PC11	x		x		x		x		x	
	TSC_G3_IO4	PC12	x		x		x		x		x	
G4	TSC_G4_IO1	PC6	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PC7	x		x		x		x		x	
	TSC_G4_IO3	PC8	x		x		x		x		x	
	TSC_G4_IO4	PC9	x		x		x		x		x	
G5	TSC_G5_IO1	PE10	x	3	x	3	x	3	-	0	-	0
	TSC_G5_IO2	PE11	x		x		x		-		-	
	TSC_G5_IO3	PE12	x		x		x		-		-	
	TSC_G5_IO4	PE13	x		x		x		-		-	

Table 38. Available touch sensing channels for STM32L486xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L486Zx		STM32L486Qx		STM32L486Vx		STM32L486Jx		STM32L486Rx	
			LQFP144		UFBGA132		LQFP100		WLCSP72		LQFP64	
G6	TSC_G6_IO1	PD10	x	3	x	3	x	3	-	0	-	0
	TSC_G6_IO2	PD11	x		x		x		-		-	
	TSC_G6_IO3	PD12	x		x		x		-		-	
	TSC_G6_IO4	PD13	x		x		x		-		-	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	3	-	0	-	0
	TSC_G7_IO2	PE3	x		x		x		-		-	
	TSC_G7_IO3	PE4	x		x		x		-		-	
	TSC_G7_IO4	PE5	x		x		x		-		-	
G8	TSC_G8_IO1	PF14	x	3	x	3	-	0	-	0	-	0
	TSC_G8_IO2	PF15	x		x		-		-		-	
	TSC_G8_IO3	PG0	x		x		-		-		-	
	TSC_G8_IO4	PG1	x		x		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		21		12		12	

Table 39. Available touch sensing channels for STM32L476xx

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L476Zx		STM32L476Qx		STM32L476Vx		STM32L476Mx		STM32L476Jx		STM32L476Rx	
			LQFP144		UFBGA132		LQFP100		WLCSP81		WLCSP72		LQFP64	
G1	TSC_G1_IO1	PB12	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G1_IO2	PB13	x		x		x		x		x		x	
	TSC_G1_IO3	PB14	x		x		x		x		x		x	
	TSC_G1_IO4	PB15	x		x		x		x		x		x	

Table 39. Available touch sensing channels for STM32L476xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L476Zx		STM32L476Qx		STM32L476Vx		STM32L476Mx		STM32L476Jx		STM32L476Rx	
			LQFP144		UFBGA132		LQFP100		WLCSP81		WLCSP72		LQFP64	
G2	TSC_G2_IO1	PB4	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G2_IO2	PB5	x		x		x		x		x		x	
	TSC_G2_IO3	PB6	x		x		x		x		x		x	
	TSC_G2_IO4	PB7	x		x		x		x		x		x	
G3	TSC_G3_IO1	PA15	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G3_IO2	PC10	x		x		x		x		x		x	
	TSC_G3_IO3	PC11	x		x		x		x		x		x	
	TSC_G3_IO4	PC12	x		x		x		x		x		x	
G4	TSC_G4_IO1	PC6	x	3	x	3	x	3	x	3	x	3	x	3
	TSC_G4_IO2	PC7	x		x		x		x		x		x	
	TSC_G4_IO3	PC8	x		x		x		x		x		x	
	TSC_G4_IO4	PC9	x		x		x		x		x		x	
G5	TSC_G5_IO1	PE10	x	3	x	3	x	3	-	0	-	0	-	0
	TSC_G5_IO2	PE11	x		x		x		-		-		-	
	TSC_G5_IO3	PE12	x		x		x		-		-		-	
	TSC_G5_IO4	PE13	x		x		x		-		-		-	
G6	TSC_G6_IO1	PD10	x	3	x	3	x	3	-	0	-	0	-	0
	TSC_G6_IO2	PD11	x		x		x		-		-		-	
	TSC_G6_IO3	PD12	x		x		x		-		-		-	
	TSC_G6_IO4	PD13	x		x		x		-		-		-	
G7	TSC_G7_IO1	PE2	x	3	x	3	x	3	-	0	-	0	-	0
	TSC_G7_IO2	PE3	x		x		x		-		-		-	
	TSC_G7_IO3	PE4	x		x		x		-		-		-	
	TSC_G7_IO4	PE5	x		x		x		-		-		-	

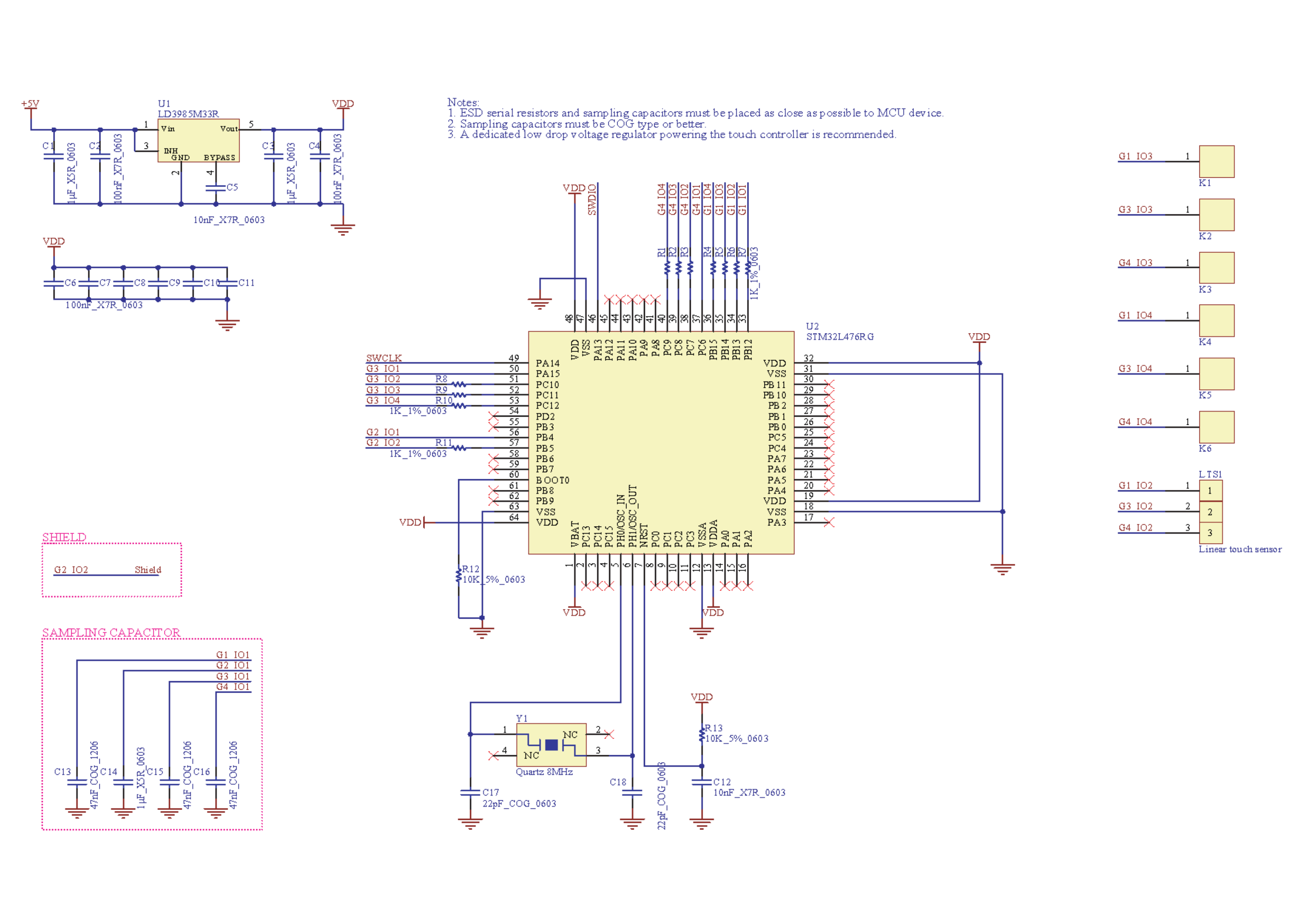
Table 39. Available touch sensing channels for STM32L476xx (continued)

Analog I/O group	Capacitive sensing signal name	Pin name	STM32L476Zx		STM32L476Qx		STM32L476Vx		STM32L476Mx		STM32L476Jx		STM32L476Rx	
			LQFP144		UFBGA132		LQFP100		WLCSP81		WLCSP72		LQFP64	
G8	TSC_G8_IO1	PF14	x	3	x	3	-	0	-	0	-	0	-	0
	TSC_G8_IO2	PF15	x		x		-		-		-		-	
	TSC_G8_IO3	PG0	x		x		-		-		-		-	
	TSC_G8_IO4	PG1	x		x		-		-		-		-	
Number of capacitive sensing channels (sampling I/Os not counted)			24		24		21		12		12		12	

3.8.3 Hardware implementation example

Figure 19 shows an example of hardware implementation on STM32L4 series microcontrollers.

Figure 19. STM32L4 series hardware implementation example



Notes:
1. ESD serial resistors and sampling capacitors must be placed as close as possible to MCU device.
2. Sampling capacitors must be COG type or better.
3. A dedicated low drop voltage regulator powering the touch controller is recommended.

4 STM32L1 series microcontrollers

These microcontrollers support two different acquisition modes: hardware and software.

4.1 Acquisition

The STM32L1 series microcontrollers **hardware acquisition mode** (using two timers) is done in the files:

- tsl_acq_stm32l1xx_hw.c
- tsl_acq_stm32l1xx_hw.h

Warning: This acquisition mode is only available for the STM32L1 series microcontrollers featuring a minimum of 384 K of Flash.

The STM32L1 series microcontrollers **software acquisition mode** is done in the files:

- tsl_acq_stm32l1xx_sw.c
- tsl_acq_stm32l1xx_sw.h

This acquisition is available for all STM32L1 series microcontrollers.

Note: The hardware acquisition mode is selected per default for the STM32L1 series microcontrollers featuring a minimum of 384 K of Flash. If you want to use the software acquisition mode you must add the following constant in the toolchain compiler preprocessor:

- TSLPRM_STM32L1XX_SW_ACQ

Functions used by the application layer and that are device dependent:

- TSL_acq_BankConfig()
- TSL_acq_BankStartAcq()
- TSL_acq_BankWaitEOC()
- TSL_acq_GetMeas()

The other functions in this file are for internal use and the user doesn't need to call them directly.

4.2 Timings

The timing management is done in the files:

- tsl_time.c
- tsl_time.h

The **Systick** is used to generate a timebase for the ECS and DTO modules. It must be initialized in the user code (already done by the HAL_init function).

4.3 Parameters

The parameters specific to the STM32L1 series microcontrollers are described in the file:

- tsl_conf_stm32l1xx_template.h (to be copied in project and rename in **tsl_conf.h**)

and are checked in the file:

- tsl_check_config_stm32l1xx.h

4.4 Memory footprint

Conditions

- IAR ANSI C/C++ compiler/linker V7.40.3.8902 for ARM®
- Compiler optimization: high size
- Counted files: tsl*.o
- STM32 TouchSensing library options: ECS=ON, DTO=ON, DXS=OFF, PROX=OFF
- Each sensor has its own parameters placed in RAM

The following tables summarize the memory footprint with different configurations

Table 40. STM32L1 series with hardware acquisition mode memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	5.3	340
2	1	2 TKeys	5.3	360
2	2	2 TKeys	5.5	360
24	3	24 TKeys	6.2	870
3	1	1 Linear-3ch	6.3	370
15	3	12 TKeys + 1 Linear-3ch	8.3	660
24	3	18 TKeys + 2 Linear-3ch	8.5	850

[1] The content of this table is provided for information purposes only.

Table 41. STM32L1 series with software acquisition mode memory footprint[1]

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
1	1	1 TKey	5.5	410
2	1	2 TKeys	5.5	430
2	2	2 TKeys	5.5	430
24	3	24 TKeys	6.2	930
3	1	1 Linear-3ch	6.5	440

Table 41. STM32L1 series with software acquisition mode memory footprint[1] (continued)

Channels	Banks	Sensors	ROM (~ Kbyte)	RAM (~ byte)
15	3	12 TKeys + 1 Linear-3ch	8.2	730
24	3	18 TKeys + 2 Linear-3ch	8.5	920

1. The content of this table is provided for information purposes only.

4.5 MCU resources

The tables below show the peripherals that are used by the STMTouch touch sensing library on STM32L1 series microcontrollers. Care must be taken when using them to avoid any unwanted behavior.

Table 42. MCU resources used on STM32L1 series with hardware acquisition

Peripheral	Function
GPIOs	Acquisition
Systick	Time base for ECS and DTO
2 Timers (TIM9, TIM11)	Acquisition
Routing interface	Acquisition

Table 43. MCU resources used on STM32L1 series with software acquisition

Peripheral	Function
GPIOs	Acquisition
Systick	Time base for ECS and DTO
Routing interface	Acquisition

4.6 Available touch sensing channels

The tables below provide an overview of the available touch sensing channels for the STM32L1 series microcontrollers.

Note: *The following tables are not restrictive in term of part numbers supported by the STMTouch touch sensing library. The STMTouch touch sensing library can be used on any new device that may become available as part of ST microcontrollers portfolio. Please contact your ST representative for support.*

Note: *For n available pins in an I/O group, one pin is used as sampling capacitor and n-1 pins are used as channels. The I/O group cannot be used if the number of available pins in less or equal to one.*

Table 44. Available touch sensing channels for STM32L1 Series 512K

Subfamily			STM32L1 series 512K												
Packages			LQFP64			LQFP100 / WLCSP104				UFBGA132			LQFP144		
Part numbers			STM32L151RE STM32L152RE STM32L162RE			STM32L151VE STM32L152VE STM32L162VE				STM32L151QE STM32L152QE STM32L162QE			STM32L151ZE STM32L152ZE STM32L162ZE		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	Number of available pins	Usage	LQFP Pin	WLCSP ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 1	G1_IO1	PA0	14	4	3 channels with 1 sampling capacitor	23	K9	4	3 channels with 1 sampling capacitor	L2	4	3 channels with 1 sampling capacitor	34	4	3 channels with 1 sampling capacitor
	G1_IO2	PA1	15			24	L9			M2			35		
	G1_IO3	PA2	16			25	J8			K3			36		
	G1_IO4	PA3[1]	17			26	H7			L3			37		
Group 2	G2_IO1	PA6	22	2	1 channel with 1 sampling capacitor	31	H6	2	1 channel with 1 sampling capacitor	L4	4[2]	3 channels with 1 sampling capacitor	42	4[2]	3 channels with 1 sampling capacitor
	G2_IO2	PA7	23			32	K7			J5			43		
	G2_IO3	PF15	-			-	-			J9			55		
	G2_IO4	PG0[3]	-			-	-			H9			56		
	G2_IO5	PG1[3]	-			-	-			G9			57		
Group 3	G3_IO1	PB0[1]	26	3	2 channels with 1 sampling capacitor	35	J6	3	2 channels with 1 sampling capacitor	M5	5	4 channels with 1 sampling capacitor	46	5	4 channels with 1 sampling capacitor
	G3_IO2	PB1	27			36	K6			M6			47		
	G3_IO3	PB2	28			37	M6			L6			48		
	G3_IO4	PF11	-			-	-			K6			49		
	G3_IO5	PF12	-			-	-			J7			50		

Table 44. Available touch sensing channels for STM32L1 Series 512K (continued)

Subfamily			STM32L1 series 512K												
Packages			LQFP64			LQFP100 / WLCSP104				UFBGA132			LQFP144		
Part numbers			STM32L151RE STM32L152RE STM32L162RE			STM32L151VE STM32L152VE STM32L162VE				STM32L151QE STM32L152QE STM32L162QE			STM32L151ZE STM32L152ZE STM32L162ZE		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	Number of available pins	Usage	LQFP Pin	WLCSP ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 4	G4_IO1	PA8	41	3	2 channels with 1 sampling capacitor	67	F3	3	2 channels with 1 sampling capacitor	D11	3	2 channels with 1 sampling capacitor	100	3	2 channels with 1 sampling capacitor
	G4_IO2	PA9	42			68	F1			D10			101		
	G4_IO3	PA10	43			69	F2			C12			102		
Group 5	G5_IO1	PA13	46	3	2 channels with 1 sampling capacitor	72	E3	3	2 channels with 1 sampling capacitor	A11	3	2 channels with 1 sampling capacitor	105	3	2 channels with 1 sampling capacitor
	G5_IO2	PA14	49			76	D3			A10			109		
	G5_IO3	PA15	50			77	B1			A9			110		
Group 6	G6_IO1	PB4	56	4	3 channels with 1 sampling capacitor	90	A5	4	3 channels with 1 sampling capacitor	A7	4	3 channels with 1 sampling capacitor	134	4	3 channels with 1 sampling capacitor
	G6_IO2	PB5	57			91	A6			C5			135		
	G6_IO3	PB6	58			92	C5			B5			136		
	G6_IO4	PB7	59			93	C7			B4			137		

Table 44. Available touch sensing channels for STM32L1 Series 512K (continued)

Subfamily			STM32L1 series 512K												
Packages			LQFP64			LQFP100 / WLCSP104				UFBGA132			LQFP144		
Part numbers			STM32L151RE STM32L152RE STM32L162RE			STM32L151VE STM32L152VE STM32L162VE				STM32L151QE STM32L152QE STM32L162QE			STM32L151ZE STM32L152ZE STM32L162ZE		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	Number of available pins	Usage	LQFP Pin	WLCSP ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 7	G7_IO1	PB12	33	4	3 channels with 1 sampling capacitor	51	J4	4	3 channels with 1 sampling capacitor	L12	5[2]	4 channels with 1 sampling capacitor	73	5[2]	4 channels with 1 sampling capacitor
	G7_IO2	PB13	34			52	J3			K12			74		
	G7_IO3	PB14	35			53	L1			K11			75		
	G7_IO4	PB15	36			54	K2			K10			76		
	G7_IO5	PG2[3]	-			-	-			G10			87		
	G7_IO6	PG3[3]	-			-	-			F9			88		
	G7_IO7	PG4[3]	-			-	-			F10			89		
Group 8	G8_IO1	PC0	8	4	3 channels with 1 sampling capacitor	15	F6	4	3 channels with 1 sampling capacitor	H1	4	3 channels with 1 sampling capacitor	26	4	3 channels with 1 sampling capacitor
	G8_IO2	PC1	9			16	H9			J2			27		
	G8_IO3	PC2	10			17	G9			J3			28		
	G8_IO4	PC3	11			18	G8			K2			29		
Group 9	G9_IO1	PC4	24	2	1 channel with 1 sampling capacitor	33	L7	2	1 channel with 1 sampling capacitor	K5	4	3 channels with 1 sampling capacitor	44	4	3 channels with 1 sampling capacitor
	G9_IO2	PC5	25			34	M7			L5			45		
	G9_IO3	PF13	-			-	-			K7			53		
	G9_IO4	PF14	-			-	-			J8			54		

Table 44. Available touch sensing channels for STM32L1 Series 512K (continued)

Subfamily			STM32L1 series 512K												
Packages			LQFP64			LQFP100 / WLCSP104				UFBGA132			LQFP144		
Part numbers			STM32L151RE STM32L152RE STM32L162RE			STM32L151VE STM32L152VE STM32L162VE				STM32L151QE STM32L152QE STM32L162QE			STM32L151ZE STM32L152ZE STM32L162ZE		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	Number of available pins	Usage	LQFP Pin	WLCSP ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 10	G10_IO1	PC6	37	4	3 channels with 1 sampling capacitor	63	H1	4	3 channels with 1 sampling capacitor	E12	4	3 channels with 1 sampling capacitor	96	4	3 channels with 1 sampling capacitor
	G10_IO2	PC7	38			64	G1			E11			97		
	G10_IO3	PC8	39			65	G2			E10			98		
	G10_IO4	PC9	40			66	F4			D12			99		
Group 11	G11_IO1	PF6	-	0	Cannot be used for touch sensing	-	-	0	Cannot be used for touch sensing	G3	4	3 channels with 1 sampling capacitor	18	5	4 channels with 1 sampling capacitor
	G11_IO2	PF7	-			-	-			G4			19		
	G11_IO3	PF8	-			-	-			H4			20		
	G11_IO4	PF9	-			-	-			J6			21		
	G11_IO5	PF10	-			-	-			-			22		
Maximum number of channels			23 channels with 10 sampling capacitors			23 channels with 10 sampling capacitors				33 channels with 11 sampling capacitors			34 channels with 11 sampling capacitors		

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.

2. Not all the pins are available simultaneously on this group.

3. This GPIO can only be configured as sampling capacitor I/O when using HW acquisition mode and as channel I/O when using SW acquisition mode.

Table 45. Available touch sensing channels for STM32L1 Series 384K

Subfamily			STM32L1 series 384K												
Packages			LQFP64 / WLCSP64				LQFP100			UFBGA132			LQFP144		
Part numbers			STM32L151RD STM32L152RD STM32L162RD				STM32L151VD STM32L152VD STM32L162VD			STM32L151QD STM32L152QD STM32L162QD			STM32L151ZD STM32L152ZD STM32L162ZD		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	WLCSP ball	Number of available pins	Usage	LQFP Pin	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 1	G1_IO1	PA0	14	F6	4	3 channels with 1 sampling capacitor	23	4	3 channels with 1 sampling capacitor	L2	4	3 channels with 1 sampling capacitor	34	4	3 channels with 1 sampling capacitor
	G1_IO2	PA1	15	E6			24			M2			35		
	G1_IO3	PA2	16	H8			25			K3			36		
	G1_IO4	PA3[1]	17	G7			26			L3			37		
Group 2	G2_IO1	PA6	22	G5	2	1 channel with 1 sampling capacitor	31	2	1 channel with 1 sampling capacitor	L4	4[2]	3 channels with 1 sampling capacitor	42	4[2]	3 channels with 1 sampling capacitor
	G2_IO2	PA7	23	G4			32			J5			43		
	G2_IO3	PF15	-	-			-			J9			55		
	G2_IO4	PG0[3]	-	-			-			H9			56		
	G2_IO5	PG1[3]	-	-			-			G9			57		
Group 3	G3_IO1	PB0[1]	26	H4	3	2 channels with 1 sampling capacitor	35	3	2 channels with 1 sampling capacitor	M5	5	4 channels with 1 sampling capacitor	46	5	4 channels with 1 sampling capacitor
	G3_IO2	PB1	27	F4			36			M6			47		
	G3_IO3	PB2	28	H3			37			L6			48		
	G3_IO4	PF11	-	-			-			K6			49		
	G3_IO5	PF12	-	-			-			J7			50		

Table 45. Available touch sensing channels for STM32L1 Series 384K (continued)

Subfamily			STM32L1 series 384K												
Packages			LQFP64 / WLCSP64				LQFP100			UFBGA132			LQFP144		
Part numbers			STM32L151RD STM32L152RD STM32L162RD				STM32L151VD STM32L152VD STM32L162VD			STM32L151QD STM32L152QD STM32L162QD			STM32L151ZD STM32L152ZD STM32L162ZD		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	WLCSP ball	Number of available pins	Usage	LQFP Pin	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 4	G4_IO1	PA8	41	E4	3	2 channels with 1 sampling capacitor	67	3	2 channels with 1 sampling capacitor	D11	3	2 channels with 1 sampling capacitor	100	3	2 channels with 1 sampling capacitor
	G4_IO2	PA9	42	D2			68			D10			101		
	G4_IO3	PA10	43	D3			69			C12			102		
Group 5	G5_IO1	PA13	46	D4	3	2 channels with 1 sampling capacitor	72	3	2 channels with 1 sampling capacitor	A11	3	2 channels with 1 sampling capacitor	105	3	2 channels with 1 sampling capacitor
	G5_IO2	PA14	49	B2			76			A10			109		
	G5_IO3	PA15	50	C3			77			A9			110		
Group 6	G6_IO1	PB4	56	B4	4	3 channels with 1 sampling capacitor	90	4	3 channels with 1 sampling capacitor	A7	4	3 channels with 1 sampling capacitor	134	4	3 channels with 1 sampling capacitor
	G6_IO2	PB5	57	A5			91			C5			135		
	G6_IO3	PB6	58	B5			92			B5			136		
	G6_IO4	PB7	59	C5			93			B4			137		

Table 45. Available touch sensing channels for STM32L1 Series 384K (continued)

Subfamily			STM32L1 series 384K												
Packages			LQFP64 / WLCSP64				LQFP100			UFBGA132			LQFP144		
Part numbers			STM32L151RD STM32L152RD STM32L162RD				STM32L151VD STM32L152VD STM32L162VD			STM32L151QD STM32L152QD STM32L162QD			STM32L151ZD STM32L152ZD STM32L162ZD		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	WLCSP ball	Number of available pins	Usage	LQFP Pin	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 7	G7_IO1	PB12	33	G2	4	3 channels with 1 sampling capacitor	51	4	3 channels with 1 sampling capacitor	L12	5[2]	4 channels with 1 sampling capacitor	73	5[2]	4 channels with 1 sampling capacitor
	G7_IO2	PB13	34	G1			52			K12			74		
	G7_IO3	PB14	35	F2			53			K11			75		
	G7_IO4	PB15	36	F1			54			K10			76		
	G7_IO5	PG2[3]	-	-			-			G10			87		
	G7_IO6	PG3[3]	-	-			-			F9			88		
	G7_IO7	PG4[3]	-	-			-			F10			89		
Group 8	G8_IO1	PC0	8	E8	4	3 channels with 1 sampling capacitor	15	4	3 channels with 1 sampling capacitor	H1	4	3 channels with 1 sampling capacitor	26	4	3 channels with 1 sampling capacitor
	G8_IO2	PC1	9	F8			16			J2			27		
	G8_IO3	PC2	10	D6			17			J3			28		
	G8_IO4	PC3[1]	11	F7			18			K2			29		
Group 9	G9_IO1	PC4	24	H6	2	1 channel with 1 sampling capacitor	33	2	1 channel with 1 sampling capacitor	K5	4	3 channels with 1 sampling capacitor	44	4	3 channels with 1 sampling capacitor
	G9_IO2	PC5	25	H5			34			L5			45		
	G9_IO3	PF13	-	-			-			K7			53		
	G9_IO4	PF14	-	-			-			J8			54		

Table 45. Available touch sensing channels for STM32L1 Series 384K (continued)

Subfamily			STM32L1 series 384K												
Packages			LQFP64 / WLCSP64				LQFP100			UFBGA132			LQFP144		
Part numbers			STM32L151RD STM32L152RD STM32L162RD				STM32L151VD STM32L152VD STM32L162VD			STM32L151QD STM32L152QD STM32L162QD			STM32L151ZD STM32L152ZD STM32L162ZD		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	WLCSP ball	Number of available pins	Usage	LQFP Pin	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 10	G10_IO1	PC6	37	E1	4	3 channels with 1 sampling capacitor	63	4	3 channels with 1 sampling capacitor	E12	4	3 channels with 1 sampling capacitor	96	4	3 channels with 1 sampling capacitor
	G10_IO2	PC7	38	E2			64			E11			97		
	G10_IO3	PC8	39	E3			65			E10			98		
	G10_IO4	PC9	40	D1			66			D12			99		
Group 11	G11_IO1	PF6	-	-	0	Cannot be used for touch sensing	-	0	Cannot be used for touch sensing	G3	4	3 channels with 1 sampling capacitor	18	5	4 channels with 1 sampling capacitor
	G11_IO2	PF7	-	-			-			G4			19		
	G11_IO3	PF8	-	-			-			H4			20		
	G11_IO4	PF9	-	-			-			J6			21		
	G11_IO5	PF10	-	-			-			-			22		
Maximum number of channels			23 channels with 10 sampling capacitors				23 channels with 10 sampling capacitors			33 channels with 11 sampling capacitors			34 channels with 11 sampling capacitors		

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.
2. Not all the pins are available simultaneously on this group.
3. This GPIO can only be configured as sampling capacitor I/O when using HW acquisition mode and as channel I/O when using SW acquisition mode.

Table 46. Available touch sensing channels for STM32L1 Series 256K (table 1/2)

Subfamily			STM32L1 series 256K									
Packages			LQFP48 or UFQFPN48			WLCSP63			LQFP64 / WLCSP64			
Part numbers			STM32L152CC			STM32L151UC			STM32L151RC STM32L152RC STM32L162RC			
Analog I/O group	Gx_IOy	GPIO	Pin	Number of available pins	Usage	WLCSP ball	Number of available pins	Usage	LQFP pin	WLCSP ball	Number of available pins	Usage
Group 1	G1_IO1	PA0	10	4	3 channels with 1 sampling capacitor	E4	4	3 channels with 1 sampling capacitor	14	F6	4	3 channels with 1 sampling capacitor
	G1_IO2	PA1	11			G5			15	E6		
	G1_IO3	PA2	12			H6			16	H8		
	G1_IO4	PA3(1)	13			J7			17	G7		
Group 2	G2_IO1	PA6	16	2	1 channel with 1 sampling capacitor	G4	2	1 channel with 1 sampling capacitor	22	G5	2	1 channel with 1 sampling capacitor
	G2_IO2	PA7	17			J5			23	G4		
	G2_IO3	PF15	-			-			-	-		
	G2_IO4	PG0(2)	-			-			-	-		
	G2_IO5	PG1(2)	-			-			-	-		
Group 3	G3_IO1	PB0(1)	18	3	2 channels with 1 sampling capacitor	J3	3	2 channels with 1 sampling capacitor	26	H4	3	2 channels with 1 sampling capacitor
	G3_IO2	PB1	19			H3			27	F4		
	G3_IO3	PB2	20			G3			28	H3		
	G3_IO4	PF11	-			-			-	-		
	G3_IO5	PF12	-			-			-	-		
Group 4	G4_IO1	PA8	29	3	2 channels with 1 sampling capacitor	E3	3	2 channels with 1 sampling capacitor	41	E4	3	2 channels with 1 sampling capacitor
	G4_IO2	PA9	30			C1			42	D2		
	G4_IO3	PA10	31			D2			43	D3		

Table 46. Available touch sensing channels for STM32L1 Series 256K (table 1/2) (continued)

Subfamily			STM32L1 series 256K									
Packages			LQFP48 or UFQFPN48			WLCSP63			LQFP64 / WLCSP64			
Part numbers			STM32L152CC			STM32L151UC			STM32L151RC STM32L152RC STM32L162RC			
Analog I/O group	Gx_IOy	GPIO	Pin	Number of available pins	Usage	WLCSP ball	Number of available pins	Usage	LQFP pin	WLCSP ball	Number of available pins	Usage
Group 5	G5_IO1	PA13	34	3	2 channels with 1 sampling capacitor	C2	3	2 channels with 1 sampling capacitor	46	D4	3	2 channels with 1 sampling capacitor
	G5_IO2	PA14	37			C3			49	B2		
	G5_IO3	PA15	38			A2			50	C3		
Group 6	G6_IO1	PB4	40	4	3 channels with 1 sampling capacitor	D4	4	3 channels with 1 sampling capacitor	56	B4	4	3 channels with 1 sampling capacitor
	G6_IO2	PB5	41			A5			57	A5		
	G6_IO3	PB6	42			B5			58	B5		
	G6_IO4	PB7	43			C5			59	C5		
Group 7	G7_IO1	PB12	25	4	3 channels with 1 sampling capacitor	G2	4	3 channels with 1 sampling capacitor	33	G2	4	3 channels with 1 sampling capacitor
	G7_IO2	PB13	26			G1			34	G1		
	G7_IO3	PB14	27			F3			35	F2		
	G7_IO4	PB15	28			F2			36	F1		
	G7_IO5	PG2(2)	-			-			-	-		
	G7_IO6	PG3(2)	-			-			-	-		
	G7_IO7	PG4(2)	-			-			-	-		

Table 46. Available touch sensing channels for STM32L1 Series 256K (table 1/2) (continued)

Subfamily			STM32L1 series 256K									
Packages			LQFP48 or UFQFPN48			WLCSP63			LQFP64 / WLCSP64			
Part numbers			STM32L152CC			STM32L151UC			STM32L151RC STM32L152RC STM32L162RC			
Analog I/O group	Gx_IOy	GPIO	Pin	Number of available pins	Usage	WLCSP ball	Number of available pins	Usage	LQFP pin	WLCSP ball	Number of available pins	Usage
Group 8	G8_IO1	PC0	-	0	Cannot be used for touch sensing	E6	4	3 channels with 1 sampling capacitor	8	E8	4	3 channels with 1 sampling capacitor
	G8_IO2	PC1	-			E5			9	F8		
	G8_IO3	PC2	-			G7			10	D6		
	G8_IO4	PC3	-			G6			11	F7		
Group 9	G9_IO1	PC4	-	0		F4	2	1 channel with 1 sampling capacitor	24	H6	2	1 channel with 1 sampling capacitor
	G9_IO2	PC5	-			J4			25	H5		
	G9_IO3	PF13	-			-			-	-		
	G9_IO4	PF14	-			-			-	-		
Group 10	G10_IO1	PC6	-	0		F1	4	3 channels with 1 sampling capacitor	37	E1	4	3 channels with 1 sampling capacitor
	G10_IO2	PC7	-			E1			38	E2		
	G10_IO3	PC8	-			D1			39	E3		
	G10_IO4	PC9	-			E2			40	D1		
Group11	G11_IO1	PF6	-	0		-	0	Cannot be used for touch sensing	-	-	0	Cannot be used for touch sensing
	G11_IO2	PF7	-			-			-	-		
	G11_IO3	PF8	-			-			-	-		
	G11_IO4	PF9	-			-			-	-		
	G11_IO5	PF10	-			-			-	-		

Table 46. Available touch sensing channels for STM32L1 Series 256K (table 1/2) (continued)

Subfamily			STM32L1 series 256K									
Packages			LQFP48 or UFQFPN48			WLCSP63			LQFP64 / WLCSP64			
Part numbers			STM32L152CC			STM32L151UC			STM32L151RC STM32L152RC STM32L162RC			
Analog I/O group	Gx_IOy	GPIO	Pin	Number of available pins	Usage	WLCSP ball	Number of available pins	Usage	LQFP pin	WLCSP ball	Number of available pins	Usage
Maximum number of channels			16 channels with 7 sampling capacitors			23 channels with 10 sampling capacitors			23 channels with 10 sampling capacitors			

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.
2. This GPIO can only be configured as sampling capacitor I/O when using HW acquisition mode and as channel I/O when using SW acquisition mode.

Table 47. Available touch sensing channels for STM32L1 Series 256K (table 2/2)

Subfamily			STM32L1 series 256K									
Packages			LQFP100 / UFBGA100				UFBGA132			LQFP144		
Part numbers			STM32L151VC STM32L152VC STM32L162VC				STM32L151QC STM32L152QC STM32L162QC			STM32L151ZC STM32L152ZC STM32L162ZC		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	BGA ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 1	G1_IO1	PA0	23	L2	4	3 channels with 1 sampling capacitor	L2	4	3 channels with 1 sampling capacitor	34	4	3 channels with 1 sampling capacitor
	G1_IO2	PA1	24	M2			M2			35		
	G1_IO3	PA2	25	K3			K3			36		
	G1_IO4	PA3(1)	26	L3			L3			37		
Group 2	G2_IO1	PA6	31	L4	2	1 channel with 1 sampling capacitor	L4	4(2)	3 channels with 1 sampling capacitor	42	4(2)	3 channels with 1 sampling capacitor
	G2_IO2	PA7	32	M4			J5			43		
	G2_IO3	PF15	-	-			J9			55		
	G2_IO4	PG0(3)	-	-			H9			56		
	G2_IO5	PG1(3)	-	-			G9			57		
Group 3	G3_IO1	PB0(1)	35	M5	3	2 channels with 1 sampling capacitor	M5	5	4 channels with 1 sampling capacitor	46	5	4 channels with 1 sampling capacitor
	G3_IO2	PB1	36	M6			M6			47		
	G3_IO3	PB2	37	L6			L6			48		
	G3_IO4	PF11	-	-			K6			49		
	G3_IO5	PF12	-	-			J7			50		
Group 4	G4_IO1	PA8	67	D11	3	2 channels with 1 sampling capacitor	D11	3	2 channels with 1 sampling capacitor	100	3	2 channels with 1 sampling capacitor
	G4_IO2	PA9	68	D10			D10			101		
	G4_IO3	PA10	69	C12			C12			102		

Table 47. Available touch sensing channels for STM32L1 Series 256K (table 2/2) (continued)

Subfamily			STM32L1 series 256K									
Packages			LQFP100 / UFBGA100				UFBGA132			LQFP144		
Part numbers			STM32L151VC STM32L152VC STM32L162VC				STM32L151QC STM32L152QC STM32L162QC			STM32L151ZC STM32L152ZC STM32L162ZC		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	BGA ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 5	G5_IO1	PA13	72	A11	3	2 channels with 1 sampling capacitor	A11	3	2 channels with 1 sampling capacitor	105	3	2 channels with 1 sampling capacitor
	G5_IO2	PA14	76	A10			A10			109		
	G5_IO3	PA15	77	A9			A9			110		
Group 6	G6_IO1	PB4	90	A7	4	3 channels with 1 sampling capacitor	A7	4	3 channels with 1 sampling capacitor	134	4	3 channels with 1 sampling capacitor
	G6_IO2	PB5	91	C5			C5			135		
	G6_IO3	PB6	92	B5			B5			136		
	G6_IO4	PB7	93	B4			B4			137		
Group 7	G7_IO1	PB12	51	L12	4	3 channels with 1 sampling capacitor	L12	5(2)	4 channels with 1 sampling capacitor	73	5(2)	4 channels with 1 sampling capacitor
	G7_IO2	PB13	52	K12			K12			74		
	G7_IO3	PB14	53	K11			K11			75		
	G7_IO4	PB15	54	K10			K10			76		
	G7_IO5	PG2(3)	-	-			G10			87		
	G7_IO6	PG3(3)	-	-			F9			88		
	G7_IO7	PG4(3)	-	-			F10			89		
Group 8	G8_IO1	PC0	15	H1	4	3 channels with 1 sampling capacitor	H1	4	3 channels with 1 sampling capacitor	26	4	3 channels with 1 sampling capacitor
	G8_IO2	PC1	16	J2			J2			27		
	G8_IO3	PC2	17	J3			J3			28		
	G8_IO4	PC3	18	K2			K2(3)			29(3)		

Table 47. Available touch sensing channels for STM32L1 Series 256K (table 2/2) (continued)

Subfamily			STM32L1 series 256K									
Packages			LQFP100 / UFBGA100				UFBGA132			LQFP144		
Part numbers			STM32L151VC STM32L152VC STM32L162VC				STM32L151QC STM32L152QC STM32L162QC			STM32L151ZC STM32L152ZC STM32L162ZC		
Analog I/O group	Gx_IOy	GPIO	LQFP pin	BGA ball	Number of available pins	Usage	BGA ball	Number of available pins	Usage	LQFP pin	Number of available pins	Usage
Group 9	G9_IO1	PC4	33	K5	2	1 channel with 1 sampling capacitor	K5	4	3 channels with 1 sampling capacitor	44	4	3 channels with 1 sampling capacitor
	G9_IO2	PC5	34	L5			L5			45		
	G9_IO3	PF13	-	-			K7			53		
	G9_IO4	PF14	-	-			J8			54		
Group 10	G10_IO1	PC6	63	E12	4	3 channels with 1 sampling capacitor	E12	4	3 channels with 1 sampling capacitor	96	4	3 channels with 1 sampling capacitor
	G10_IO2	PC7	64	E11			E11			97		
	G10_IO3	PC8	65	E10			E10			98		
	G10_IO4	PC9	66	D12			D12			99		
Group11	G11_IO1	PF6	-	-	0	Cannot be used for touch sensing	G3	4	3 channels with 1 sampling capacitor	18	5	4 channels with 1 sampling capacitor
	G11_IO2	PF7	-	-			G4			19		
	G11_IO3	PF8	-	-			H4			20		
	G11_IO4	PF9	-	-			J6			21		
	G11_IO5	PF10	-	-			-			22		
Maximum number of channels			23 channels with 10 sampling capacitors				33 channels with 11 sampling capacitors			34 channels with 11 sampling capacitors		

1. This GPIO offers a reduced touch sensing sensitivity. It is thus recommended to use it as sampling capacitor I/O.
2. Not all the pins are available simultaneously on this group.
3. This GPIO can only be configured as sampling capacitor I/O when using HW acquisition mode and as channel I/O when using SW acquisition mode.

Table 48. Available touch sensing channels for STM32L15x 32K to 128K

Subfamily			**STM32L15x 32K to 128K**										
Packages			**LQFP48 / VFQFPN48**			**LQFP64 / BGA64**				**LQFP100 / BGA100**			
Part numbers			**STM32L151C6 STM32L151C8 STM32L151CB STM32L152C6 STM32L152C8 STM32L152CB**			**STM32L151R6 STM32L151R8 STM32L151RB STM32L152R6 STM32L152R8 STM32L152RB**				**STM32L151V8 STM32L151VB STM32L152V8 STM32L152VB**			
Analog I/O group	**Gx_IOy**	**GPIO**	**Pin**	**Number of available pins**	**Usage**	**LQFP pin**	**BGA ball**	**Number of available pins**	**Usage**	**LQFP pin**	**BGA ball**	**Number of available pins**	**Usage**
Group 1	G1_IO1	PA0	10	4	3 channels with 1 sampling capacitor	14	G2	4	3 channels with 1 sampling capacitor	23	L2	4	3 channels with 1 sampling capacitor
	G1_IO2	PA1	11			15	H2			24	M2		
	G1_IO3	PA2	12			16	F3			25	K3		
	G1_IO4	PA3	13			17	G3			26	L3		
Group 2	G2_IO1	PA6	16	2	1 channel with 1 sampling capacitor	22	G4	2	1 channel with 1 sampling capacitor	31	L4	2	1 channel with 1 sampling capacitor
	G2_IO2	PA7	17			23	H4			32	M4		
Group 3	G3_IO1	PB0	18	2	1 channel with 1 sampling capacitor	26	F5	2	1 channel with 1 sampling capacitor	35	M5	2	1 channel with 1 sampling capacitor
	G3_IO2	PB1	19			27	G5			36	M6		
Group 4	G4_IO1	PA8	29	3	2 channels with 1 sampling capacitor	41	D7	3	2 channels with 1 sampling capacitor	67	D11	3	2 channels with 1 sampling capacitor
	G4_IO2	PA9	30			42	C7			68	D10		
	G4_IO3	PA10	31			43	C6			69	C12		
Group 5	G5_IO1	PA13	34	3	2 channels with 1 sampling capacitor	46	A8	3	2 channels with 1 sampling capacitor	72	A11	3	2 channels with 1 sampling capacitor
	G5_IO2	PA14	37			49	A7			76	A10		
	G5_IO3	PA15	38			50	A6			77	A9		

Table 48. Available touch sensing channels for STM32L15x 32K to 128K (continued)

Subfamily			STM32L15x 32K to 128K										
Packages			LQFP48 / VFQFPN48			LQFP64 / BGA64				LQFP100 / BGA100			
Part numbers			STM32L151C6 STM32L151C8 STM32L151CB STM32L152C6 STM32L152C8 STM32L152CB			STM32L151R6 STM32L151R8 STM32L151RB STM32L152R6 STM32L152R8 STM32L152RB				STM32L151V8 STM32L151VB STM32L152V8 STM32L152VB			
Analog I/O group	Gx_IOy	GPIO	Pin	Number of available pins	Usage	LQFP pin	BGA ball	Number of available pins	Usage	LQFP pin	BGA ball	Number of available pins	Usage
Group 6	G6_IO1	PB4	40	2	1 channel with 1 sampling capacitor	56	A4	2	1 channel with 1 sampling capacitor	90	A7	2	1 channel with 1 sampling capacitor
	G6_IO2	PB5	41			57	C4			91	C5		
Group 7	G7_IO1	PB12	25	4	3 channels with 1 sampling capacitor	33	H8	4	3 channels with 1 sampling capacitor	51	L12	4	3 channels with 1 sampling capacitor
	G7_IO2	PB13	26			34	G8			52	K12		
	G7_IO3	PB14	27			35	F8			53	K11		
	G7_IO4	PB15	28			36	F7			54	K10		

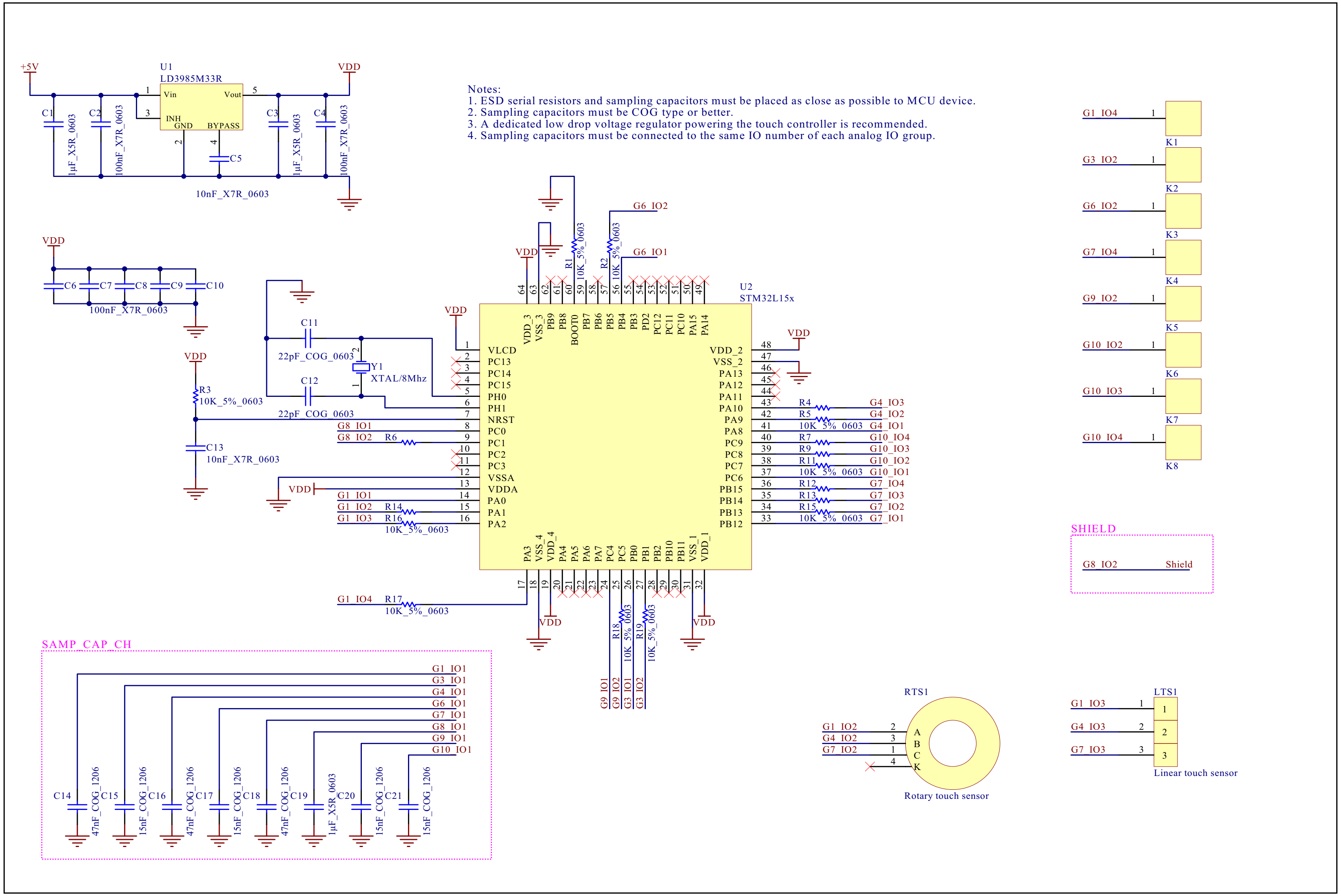
Table 48. Available touch sensing channels for STM32L15x 32K to 128K (continued)

Subfamily				STM32L15x 32K to 128K									
Packages				LQFP48 / VFQFPN48		LQFP64 / BGA64				LQFP100 / BGA100			
Part numbers				STM32L151C6 STM32L151C8 STM32L151CB STM32L152C6 STM32L152C8 STM32L152CB		STM32L151R6 STM32L151R8 STM32L151RB STM32L152R6 STM32L152R8 STM32L152RB				STM32L151V8 STM32L151VB STM32L152V8 STM32L152VB			
Analog I/O group	**Gx_IOy**	**GPIO**	**Pin**	**Number of available pins**	**Usage**	**LQFP pin**	**BGA ball**	**Number of available pins**	**Usage**	**LQFP pin**	**BGA ball**	**Number of available pins**	**Usage**
Group 8	G8_IO1	PC0	-	0	Cannot be used for touch sensing	8	E3	4/3	3/2 channels with 1 sampling capacitor	15	H1	4	3 channels with 1 sampling capacitor
	G8_IO2	PC1	-			9	E2			16	J2		
	G8_IO3	PC2	-			10	F2			17	J3		
	G8_IO4	PC3	-			11	-			18	K2		
Group 9	G9_IO1	PC4	-	0		24	H5	2	1 channel with 1 sampling capacitor	33	K5	2	1 channel with 1 sampling capacitor
	G9_IO2	PC5	-			25	H6			34	L5		
Group 10	G10_IO1	PC6	-	0		37	F6	4	3 channels with 1 sampling capacitor	63	E12	4	3 channels with 1 sampling capacitor
	G10_IO2	PC7	-			38	E7			64	E11		
	G10_IO3	PC8	-			39	E8			65	E10		
	G10_IO4	PC9	-			40	D8			66	D12		
Maximum number of channels				13 channels with 7 sampling capacitors		20/19 channels with 10 sampling capacitors				20 channels with 10 sampling capacitors			

4.7 Hardware implementation example

Figure 20 shows an example of hardware implementation on STM32L1 series microcontrollers.

Figure 20. STM32L1 series hardware implementation example



Notes:
1. ESD serial resistors and sampling capacitors must be placed as close as possible to MCU device.
2. Sampling capacitors must be COG type or better.
3. A dedicated low drop voltage regulator powering the touch controller is recommended.
4. Sampling capacitors must be connected to the same IO number of each analog IO group.

5 Getting started

5.1 Create your application

Start with an application example present in the STM32Cube package of the device you intend to use. Take an example that is close in term of number of channels/sensors with your target application. Copy and paste the example in the same parent folder and rename it according your target application. Then modify the files as described below.

The following sections describe the necessary steps to create a new application project.

5.1.1 Toolchain compiler preprocessor section

The device that you intend to use must be written in the **toolchain compiler preprocessor section** of your project.

These defines are the same as those used by the STM32Cube. Please see the stm<xxx>.h map file to have the list of the microcontrollers definition**.**

Note: The hardware acquisition mode is selected per default for the STM32L1 series microcontrollers with a minimum of 384 K of Flash. If you want to use the software acquisition mode you must add the following constant in the toolchain compiler preprocessor:

- TSLPRM_STM32L1XX_SW_ACQ

5.1.2 The tsl_conf.h file

The **tsl_conf.h** file contains all the STMTouch touch sensing library parameters. The following edits must be done:

1. Change the number of channels, banks, sensors according your application.
2. Change the common parameters: thresholds, debounce, ECS, DTO, etc.
3. Change the parameters specific to the device.

5.1.3 The main file

The **main.c** and **main.h** files contain the application code itself (LEDs and LCD management, etc.) and the call to the STMTouch touch sensing library initialization and action functions.

5.1.4 The tsl_user file

The **tsl_user.c** and **tsl_user.h** files contain the STMTouch touch sensing library configuration (definition of channels, banks, sensors, etc.) and the STMTouch touch sensing library initialization (**TSL_user_Init**) and action (**TSL_user_Action**) functions.

Create the channels variables using the structures (**mandatory**):

- TSL_ChannelSrc_T
- TSL_ChannelDest_T
- TSL_ChannelData_T

Create the Banks variables using the structures (**mandatory**):

- TSL_Bank_T

Create the touchkeys variables using the structures (optional):

- TSL_TouchKeyData_T
- TSL_TouchKeyParam_T
- TSL_State_T
- TSL_TouchKeyMethods_T
- TSL_TouchKeyB_T
- TSL_TouchKey_T

Create the linear and rotary touch sensors variables using the structures (optional):

- TSL_LinRotData_T
- TSL_LinRotParam_T
- TSL_State_T
- TSL_LinRotMethods_T
- TSL_LinRotB_T
- TSL_LinRot_T

Create the generic sensors (objects) variables using the structures (**mandatory**):

- TSL_Object_T
- TSL_ObjectGroup_T

The **TSL_user_Init()** function contains the initialization of the STMTouch touch sensing library. Modify this function to take into account your bank array name and object groups names.

The **TSL_user_Action()** function contains the main state machine. Modify it also if you have several object groups to process or to change the ECS period, etc.

5.2 Debug with STM Studio

The STM Studio software is very useful to observe variables of the STMTouch touch sensing library. Its powerful features allow to better understand how the sensors behave and to find the better parameters to apply.

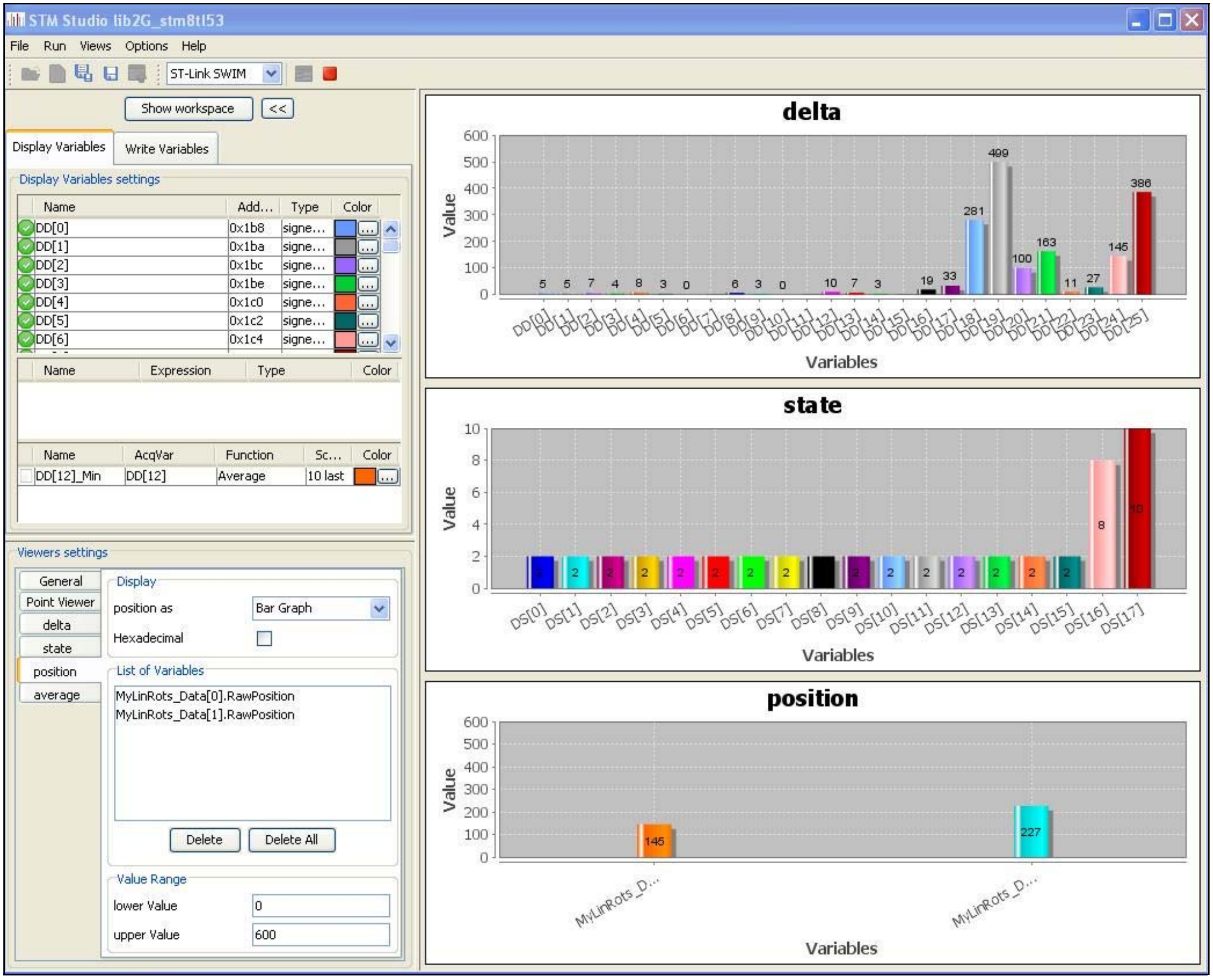
This section does not intend to explain how to use this tool, but give some advice to better understand and debug user's application.

This is a non-exhaustive list of the STMTouch touch sensing library variables to observe:

- The **channels measure**, **reference** and **delta**. These variables are present inside the **TSL_ChannelData_T** structure. This is useful to adjust the **thresholds** parameters.
- The **sensors state** present in the **TSL_TouchKeyData_T** and **TSL_LinRotData_T** structures. This is useful to adjust the **Debounce**, **ECS** and **DTO** parameters.
- The linear and rotary touch sensors **position** in the **TSL_LinRotData_T** structure.

The following snapshot is an example of data visualization on STM Studio:

Figure 21. STM Studio snapshot



Name	Add...	Type	Color
DD[0]	0x1b8	signe...	🟦 ...
DD[1]	0x1ba	signe...	⬜ ...
DD[2]	0x1bc	signe...	🟪 ...
DD[3]	0x1be	signe...	🟩 ...
DD[4]	0x1c0	signe...	🟧 ...
DD[5]	0x1c2	signe...	🟩 ...
DD[6]	0x1c4	signe...	🟥 ...

Name	Expression	Type	Color

Name	AcqVar	Function	Sc...	Color
□ DD[12]_Min	DD[12]	Average	10 last	🟧 ...

Display

position as: Bar Graph

Hexadecimal □

List of Variables

MyLinRots_Data[0].RawPosition
MyLinRots_Data[1].RawPosition

Value Range

lower Value: 0

upper Value: 600

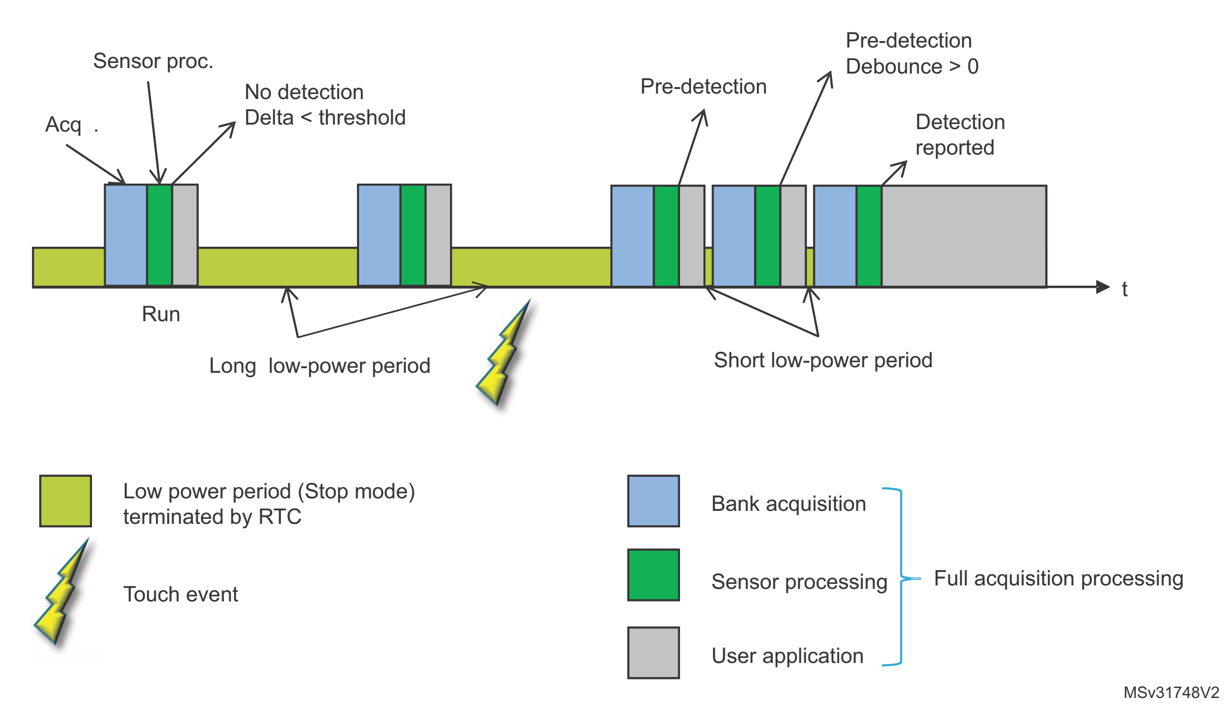
5.3 Low-power strategy

The following figure shows the acquisition sequencing for a single bank acquisition to optimize the power consumption of the device.

To reduce the power consumption, the acquisitions are sequenced with a long delay in between. During this delay, the MPU can be in low-power mode (i.e. Stop mode). This delay can be shortened or even removed between two consecutive acquisitions when the delta becomes greater than a detection threshold (proximity or touch). The long delay is restored if all the sensors return in RELEASE state.

For optimum power consumption, the acquisition should be performed with the MCU in Sleep or Low-power sleep mode and with the optimum TSC peripheral clock frequency (not too low or too high). The sensor processing should be performed at the highest possible frequency in order to minimize the processing duration. The user application processing should be done at the optimum CPU frequency to offer the best trend between task duration and power consumption.

Figure 22. Low-power strategy



This approach allows to save power consumption without increasing the response time. The maximum response time is obtained when a touch occurs during the sensor processing. It can be expressed as followed:

> Max Response Time = long low-power period + (n) x short low-power period + (n+2) x full acquisition processing - bank acquisition

with n being the debounce value.

5.4 Tips and tricks

5.4.1 Recommendations to increase the noise immunity on the PCB

To ensure a correct operation in noisy environment, the floating nets must be avoided (tracks, copper elements, conductive frames, etc.).

As a consequence:

- All unused touch controller I/Os must be either configured to output push-pull low or externally tied to GND.
- The parameter TSLPRM_TSC_IODEF should also be configured to the output push-pull low state.
- We recommend to drive the sampling capacitor common node using a standard I/O of the touch controller configured in output push-pull low mode.
- It may also be required to add a capacitor-input filter (pi filter) on each channel line.

5.4.2 Bank definition

For optimum sensitivity and position reporting, all the channels composing a linear or a rotary touch sensor must be acquired **simultaneously**. This means that all the channels must belong to the **same bank**.

Note: The library allows to define a linear or a rotary touch sensor with channels belonging to different banks. A such configuration induces a ***loss of sensitivity*** *and a* ***higher noise level****. Moreover, depending on the acquisition time, it is also possible to observe a position change when removing the finger from the sensor.*

5.4.3 Channel assignment

It is recommended to assign GPIOs offering the same sensitivity level to all the channels composing a linear or a rotary touch sensor. Moreover, it is not recommended to use GPIOs offering a reduced sensitivity.

5.4.4 IO Default state parameter

For optimum acquisition noise level, it is recommended to set the **TSLPRM_TSC_IODEF**

or **TSLPRM_IODEF** parameter to **output push-pull low**.

However, if your application is using a linear or a rotary touch sensor with channels belonging to different banks, this parameter must be set to **input floating**. This will ensure optimum sensitivity.

5.5 Related documents

AN4312 - Guidelines for designing touch sensing applications with surface sensors.

AN4299 - Guidelines to improve conducted noise robustness on STM32F0/F3, STM32L0/L4 series touch sensing applications.

AN4316 - Tuning a STMTouch-based application.

6 Revision history

Table 49. Document revision history

Date	Revision	Changes
05-Jan-2016	1	Initial release.
29-Feb-2016	2	Updated *Section 2.4.3: Acquisition and processing layers*. Removed former *Section 2.7: Zone*.
19-May-2016	3	Updated document title. Updated *Section : Introduction*. Updated line TSC_G3_IO4 in *Table 22: Available touch sensing channels for STM32F334x4/x6/x8*.
24-Nov-2016	4	Updated *Section 5.5: Related documents*.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2016 STMicroelectronics – All rights reserved