Getting started with touch sensing control on STM32 microcontrollers

Introduction

This document helps customers to quickly locate information regarding touch sensing on STM32 microcontrollers.

This application note lists all the existing application notes and user manuals covering touch sensing. It indicates where the key aspects of touch sensing are documented.

It also explains how to build touch sensing applications on STM32L0538-DISCO and STM32F072B-DISCO discovery boards using the STM32CubeMX graphical interface.

Table 1. **Applicable products**

Type	Product series
STM32 microcontrollers	STM32F0 series
	STM32F3 series
	STM32L0 series
	STM32L1 series
	STM32L4 series
	STM32L4+ series
	STM32L5 series
	STM32U5 series
	STM32WB series
	STM32WBA series

1 General information

This document applies to Arm®-based devices.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

2 Terminology and principle

2.1 Terminology

The touch sensing most relevant acronyms are described below:

- Acquisition mode
  - CT: Charge-Transfer acquisition principle. This mode is used on STM32 microcontrollers.
- Touch sensing STM32 peripheral
  - TSC: touch sensing controller peripheral.
  - Bank: set of channels acquired simultaneously.
  - Channel: elementary acquisition item.
  - Group: set of 1..3 channels plus one sampling capacitor (C_S).
- Sensors
  - Touchkey or TKey: single channel sensor.
  - Linear sensor: multi-channels sensor with the electrodes positioned in a linear way.
  - Rotary sensor: multi-channels sensor with the electrodes positioned in a circular way.
  - Active shield: track running along or copper plane surrounding the sensor track and/or sensor itself. Active shield is driven similarly to the sensor. Improve noise robustness without decreasing the sensitivity.
- STM32 software
  - TSL: touch sensing library.
  - Delta: difference between the measure and the reference.
  - Measure or meas: current signal measured on a channel.
  - Reference or ref: reference signal based on the average of a sample of measures.
  - DTO: detection time out. Time out is defined by TSLPRM_DTO. See TSLPRM_DTO in tsl_conf.h file.
  - DXS: detection exclusion system. Exclusion system is defined by TSLPRM_USE_DXS. See TSLPRM_USE_DXS in tsl_conf.h file.
  - ECS: environment change system. See TSLPRM_ECS_DELAY in tsl_conf.h file.
- Hardware involved
  - C_X: sensor capacitance (typical value is few pF).
  - C_P: parasitic capacitance (typical value few pF).
  - C_t: equivalent touch capacitance.
  - $C_s/C_{skey}/C_{sshield}$: sampling capacitor (typical value from 2.2 to 100nF).
  - $R_s/R_{skey}/R_{sshield}$: serial resistor, ESD protection (typical value from 100Ohms to 10K).

2.2 Principle

The STM32 touch sensing feature is based on charge transfer.

The surface charge transfer acquisition principle consists in charging a sensor capacitance (C_X) and in transferring the accumulated charge into a sampling capacitor (C_S).

This sequence is repeated until the voltage across C_s reaches V_{IH}.

The number of charge transfers required to reach the threshold is a direct representation of the size of the electrode capacitance. When the sensor is touched, the sensor capacitance to the earth is increased. This means that the C voltage reaches V_{IH} with less count and the measurement value decreases. When this measurement goes below a threshold, a detection is reported by the TSL. The schematic below does not take into account the parasitic capacitor.

Figure 1. Change transfer principle

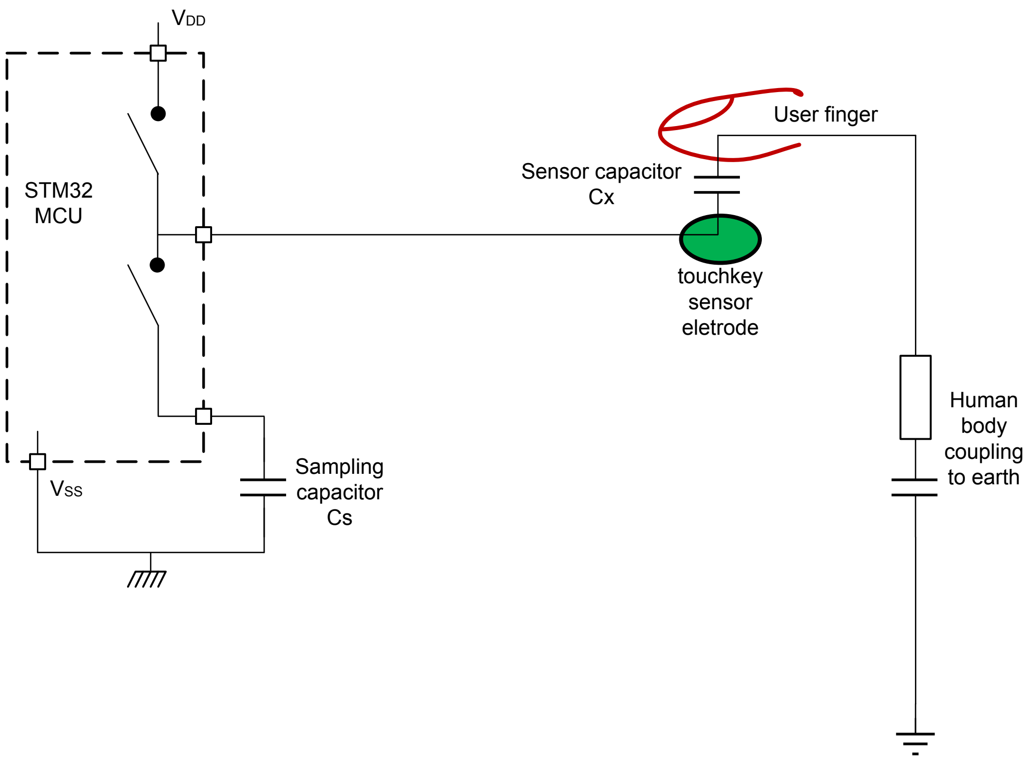


Table 2 gives a list of documents containing information about the change transfer principle.

Table 2. Change transfer principle documentation

Id	Title	Chapters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	Surface charge transfer acquisition principle overview
AN4310	Sampling capacitor selection guide for touch sensing applications on MCUs	Charge transfer acquisition principle overview
AN4312	Design with surface sensors for touch sensing applications on MCUs	Capacitive sensing technology
AN4316	Tuning a touch sensing application on MCUs	Charge transfer period tuning
OLT	STM32L4 online training	Touch sensing controller (TSC)

3 Document reference

Figure 2 shows the main documentation tree related to TSC and TSL.

Figure 2. Main documentation tree

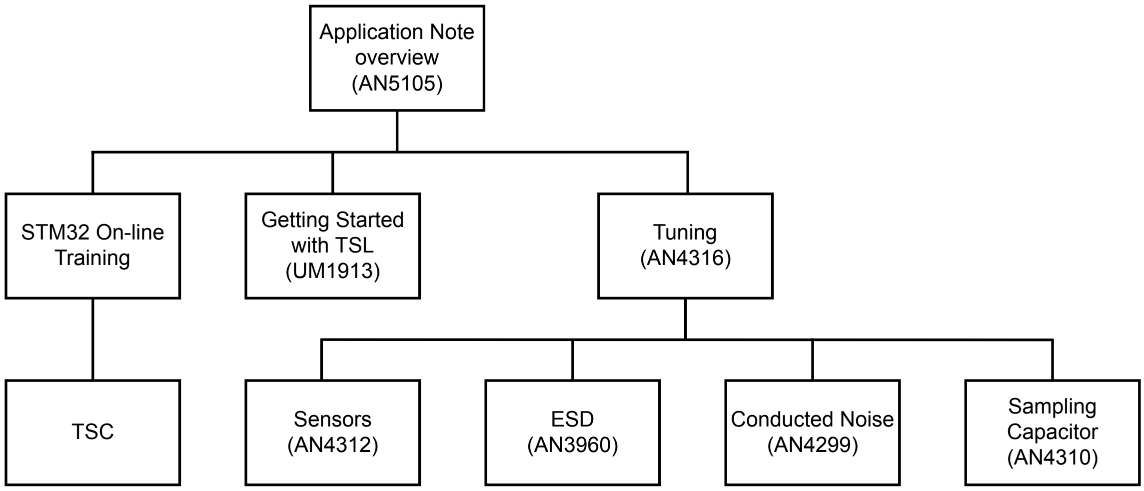


Table 3. Reference documentation

Document name	Document title
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library
AN3960	ESD considerations for touch sensing applications on MCUs
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs
AN4310	Sampling capacitor selection guide for touch sensing applications on MCUs
AN4312	Design with surface sensors for touch sensing applications on MCUs
AN4316	Tuning a touch sensing application on MCUs

4 STM32L4 touch sensing controller online presentation

An online training is available from the STMicroelectronics website www.st.com.

Insert the "STM32L4 Online Training" string in the "Search" function and press enter.

To find it use the function "Search" and insert the strings “STM32L4 Online Training”. The figure below shows the online page available.

Figure 3. STM32L4 online training

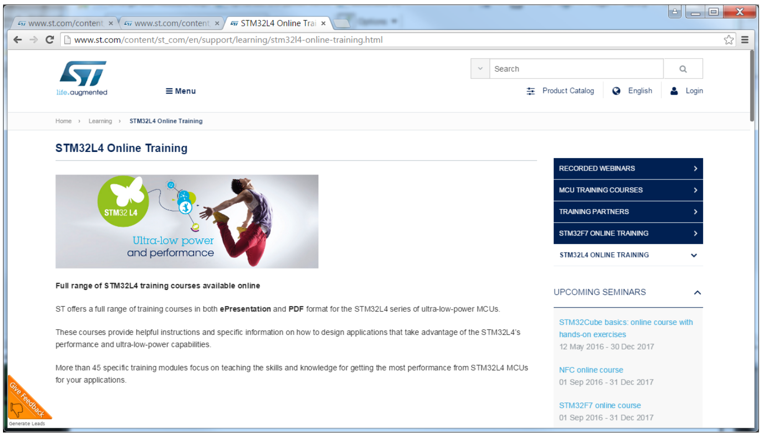
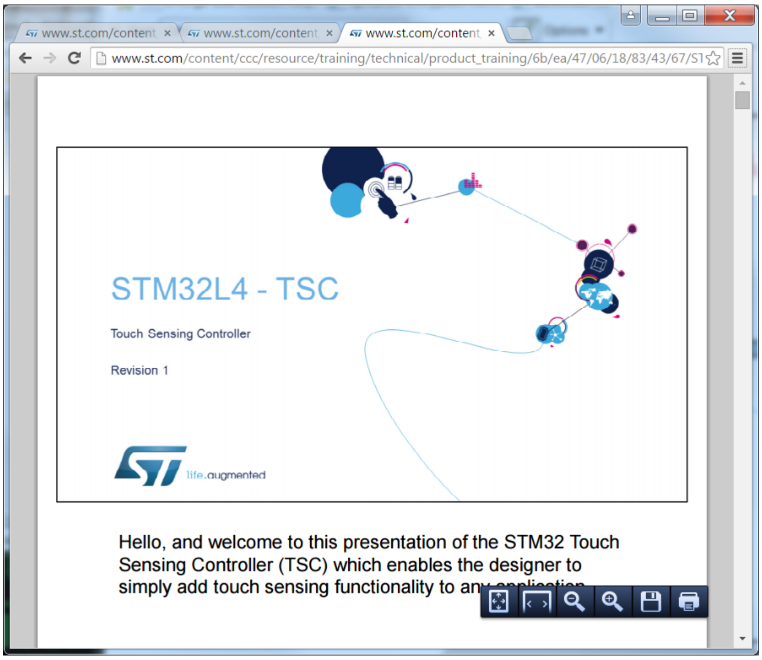


Figure 4. STM32L4 Touch Sensing Controller online training



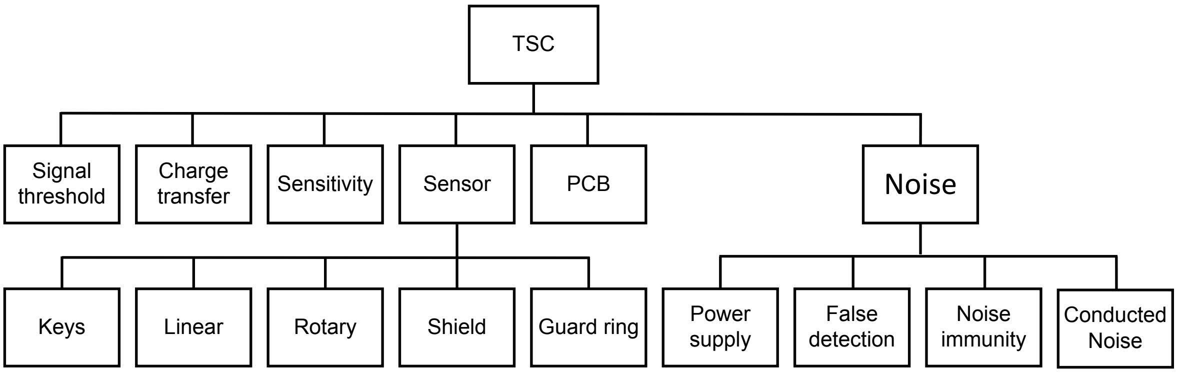
5 Main TSC characteristics

5.1 Description

The following Figure 5 shows all touch sensing controller (TSC) characteristics and their correlation.

The TSC main characteristics are described in the following pages.

Figure 5. **TSC characteristics**



5.2 Signal threshold

To tune the detection thresholds, it must determine the sensitivity of each touchkey. For each touchkey, can be used few parameters to adjust these signal thresholds.

For debug purposes, it can get touchkey parameters using printf or the STMStudio tool:

```
for (Index = 0;Index < NUMBER_OF_TOUCHKEYS;Index++)
{
    printf("K%1d [%2d][%4d %3d %3d %4d] %d %d %d %d %d"
          ,Index
          ,MyTKeys[Index].p_Data->StateId
          ,MyTKeys[Index].p_ChD->Ref
          ,MyTKeys[Index].p_ChD->RefRest
          ,MyTKeys[Index].p_ChD->Delta
          ,MyTKeys[Index].p_ChD->Meas
          ,MyTKeys[Index].p_Param->ProxInTh
          ,MyTKeys[Index].p_Param->ProxOutTh
          ,MyTKeys[Index].p_Param->DetectInTh
          ,MyTKeys[Index].p_Param->DetectOutTh
          ,MyTKeys[Index].p_Param->CalibTh
          );
}
```

Note: ProxInTh and ProxOutTh are defined for the proximity detection feature only, when TSLPRM_USE_PROX = 1.

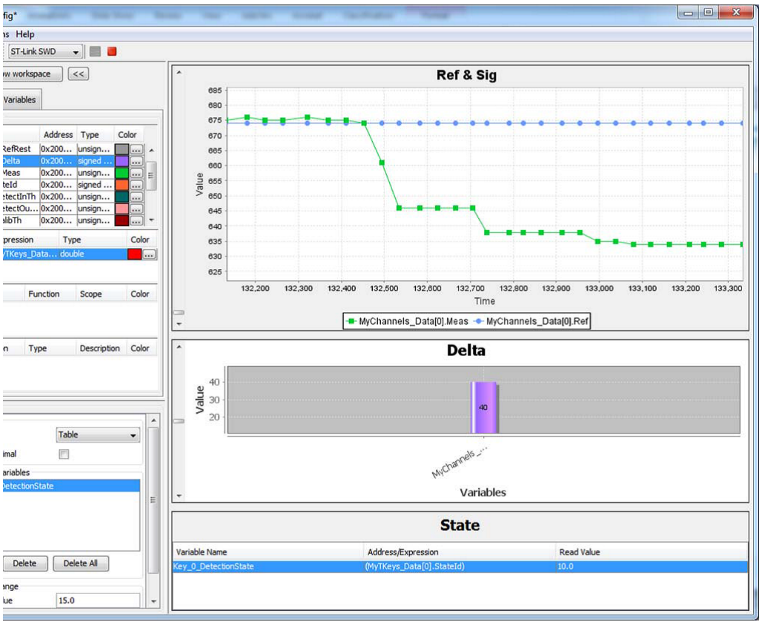


Figure 6. STMStudio outputs

- On software side:
  - Relevant information is available in tsl_conf.h and tscl_user.c files.
  - Threshold (xx_TH) can be adjusted in tsl_conf_tsc.h file.

See below an example:

```
#define TSLPRM_TKEY_DETECT_IN_TH (64)
#define TSLPRM_TKEY_DETECT_OUT_TH (60)
#define TSLPRM_TKEY_CALIB_TH (56)
#define TSLPRM_LINROT_DETECT_IN_TH (50)
#define TSLPRM_LINROT_DETECT_OUT_TH (40)
```

- The TSL api, tsl_user_SetThresholds, located in `tsl_user.c` allows the adjustment of each channel independently. See below an example:

```
void tsl_user_SetThresholds(void)
{
/* USER CODE BEGIN Tsl_user_SetThresholds */
  /* Example: Decrease the Detect thresholds for the TKEY 0*/
  MyTKeys_Param[0].DetectInTh -= 10;
  MyTKeys_Param[0].DetectOutTh -= 10;
 /* USER CODE END Tsl_user_SetThresholds */
  }
```

Table 4 gives a list of documents containing information about the signal threshold usage.

Table 4. Signal threshold use documentation

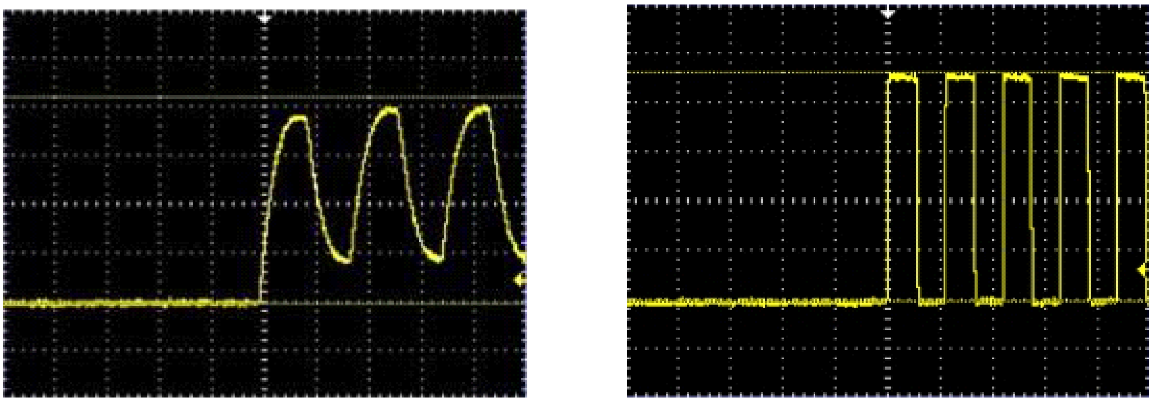
Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Debug with STM Studio
AN4316	Tuning a touch sensing application on MCUs	Monitoring STMTouch driver variables using STM-Studio Tuning the thresholds Touchkeys thresholds Linear and rotary touch sensors thresholds

5.3 Charge transfer

The acquisition is based on the measurement of the sensor channel capacitance.

To ensure that the C_x capacitance is correctly charged, it is necessary to monitor the pin connected to the sensor. On the sensors and shield sides, it must observe a complete Charge/Discharge cycle.

Figure 7. Incomplete and complete charge transfer cycle



In this example, to complete the charge transfer cycles, the following parameter must be modified as below:

- INCREASE:
  - htsc.Init.PulseGeneratorPrescaler
  - htsc.Init.CTPulseHighLength
  - htsc.Init.CTPulseLowLength
- DECREASE:
  - Sysclk

Table 5 gives a list of documents containing information about the charge transfer.

Table 5. Charge transfer documentation

Id	Title	Chapters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	Active shield
AN4316	Tuning a touch sensing application on MCUs	Charge transfer period tuning

5.4 Sensitivity

Sensitivity is a key point in touch sensing applications. The sensitivity can be improved by:

- reducing the air gap,
- reducing the panel thickness,
- choosing the dielectric with higher ε_R ,
- a GND plane that is not too close to the shield and sensors,
- avoiding metallic paint near shield and sensors.

Table 6 gives a list of documents containing information about the sensitivity.

Table 6. Sensitivity documentation

Id	Title	Chapter
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Available touch sensing channels
AN4312	Design with surface sensors for touch sensing applications on MCUs	Air gap: • Reduce air gap Panel material: • Reduce panel thickness • Choose dielectric with higher ε_R Metal chassis: • GND not too closed from Shield and Sensors • Avoid Metallic paint near Shield and Sensors Mechanical construction and PCB to panel bonding. Surface sensor design
AN4316	Tuning a touch sensing application on MCUs	All chapters

Dielectric example

Table 7. Dielectric constants of common materials used in a panel construction

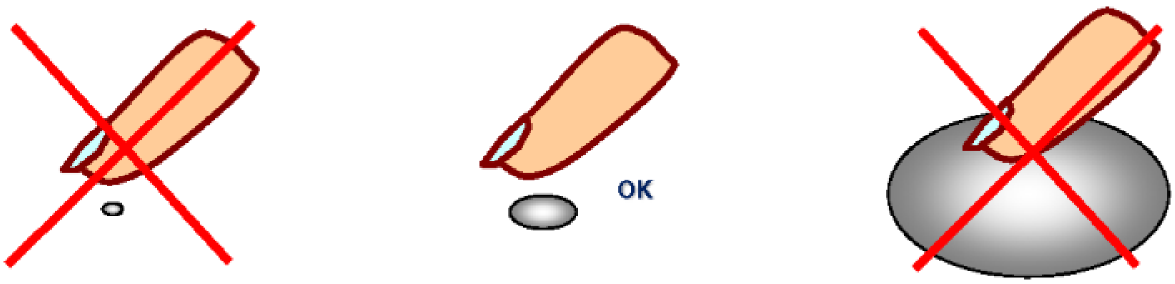
Material	ε_R
Air	1.00059
Glass	4 to 10
Sapphire glass	9 to 11
Mica	4 to 8
Nylon	3
Plexiglass	3.4
Polyethylene	2.2
Polystyrene	2.56
Polyethylene terephthalate (PET)	3.7
FR4 (fiberglass + epoxy	4.2
PMMA (Poly methyl methacrylate)	2.6 to 4
Typical PSA	2.0 - 3.0 (approximately)

5.5 Sensors

- It is recommended to use the same shape for all electrodes.
- The touchkeys can be customized by the drawing on the panel. TSL compensates capacitance differences.
- Acquisition time and processing parameters can be optimized when electrodes have similar capacitance.

Sensor size example

Figure 8. Sensor size



5.5.1 Key

- Key sensors are used in common application
- You can get deeper key information in the following documents:

Table 8 gives a list of documents containing information about the key.

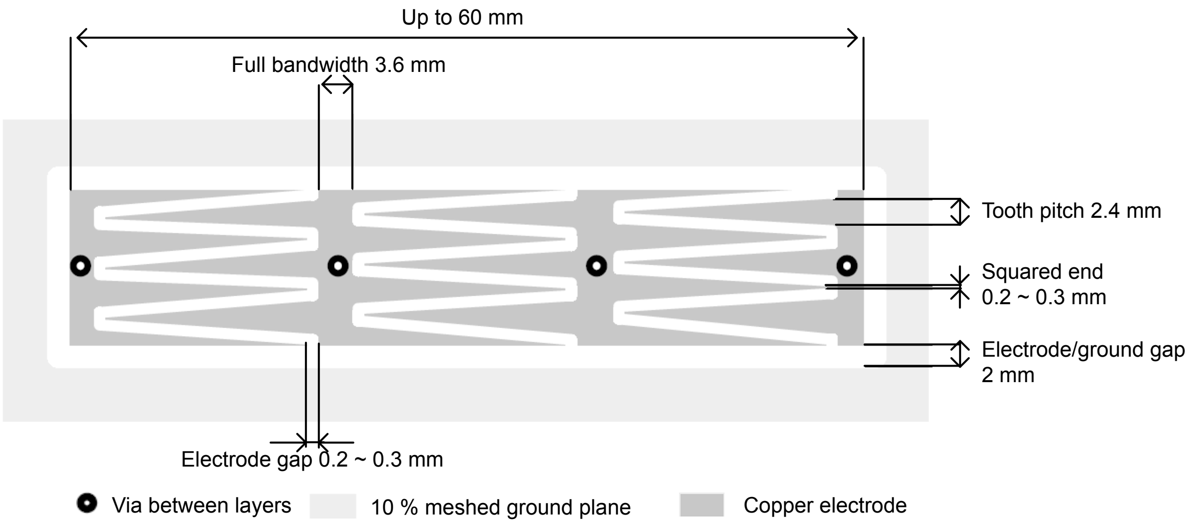
Table 8. Key documentation

Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Touchkey sensor
AN4312	Design with surface sensors for touch sensing applications on MCUs	Touchkey sensor

5.5.2 Linear or slider

A linear is a set of contiguous capacitive electrodes. Figure 9 shows a slider used on a discovery board.

Figure 9. Interlaced linear touch sensor with three channels / four electrodes (half-ended electrodes design)



Note: The teeth of the interlaced linear touch sensor must be perfectly regular.

Table 9 gives a list of documents containing information about the linear touch sensor.

Table 9. Linear touch sensor documentation

Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Linear and rotary touch sensors
AN4312	Design with surface sensors for touch sensing applications on MCUs	Linear sensor

5.5.3 Rotary sensor or wheel

A rotary is a set of contiguous capacitive electrodes.

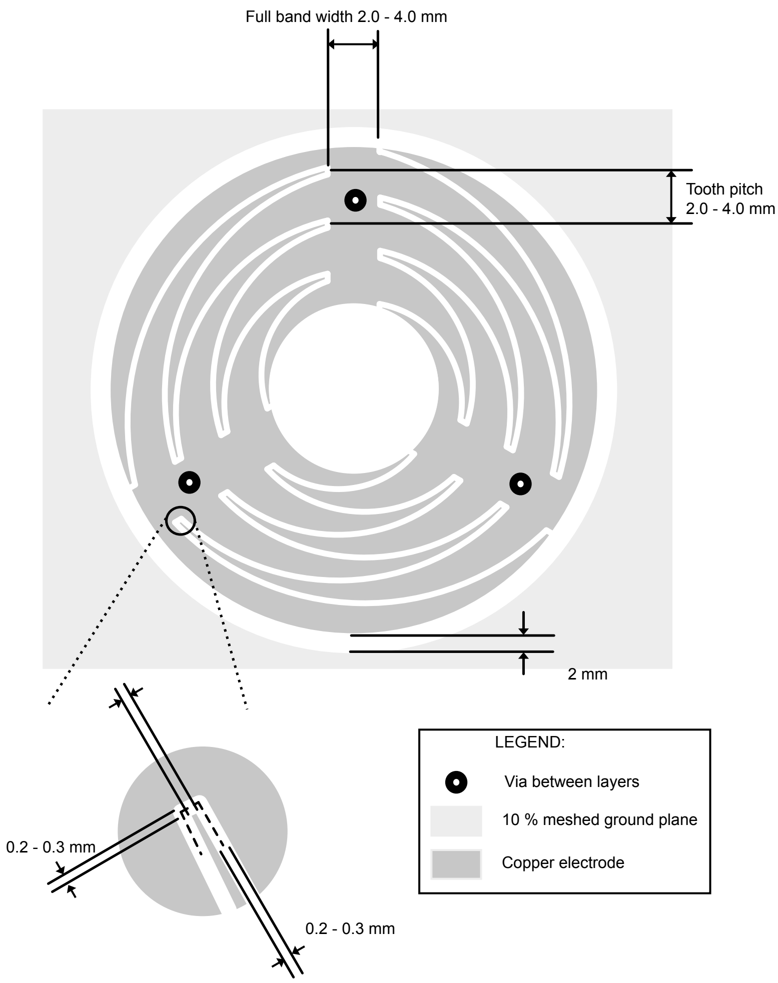


Figure 10. Interlaced patterned rotary sensor with three channels / three electrodes

Table 10 gives a list of documents containing information about the rotary sensor.

Table 10. Rotary sensor documentation

Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Linear and rotary touch sensors
AN4312	Design with surface sensors for touch sensing applications on MCUs	Rotary sensor

5.5.4 Active shield or driven shield

Active shield or driven shield. (this name is used in some application notes) drives the shield plane with the same signal as the electrode.

There are several advantages using Active Shield instead of a grounded shield:

- The parasitic capacitance between the electrode and the shield no longer needs to be charged.
- Protect the touch electrodes from a noise source.
- Increase system stability and performance when a moving metal part is close to the electrode.

Figure 11. Active shield principle

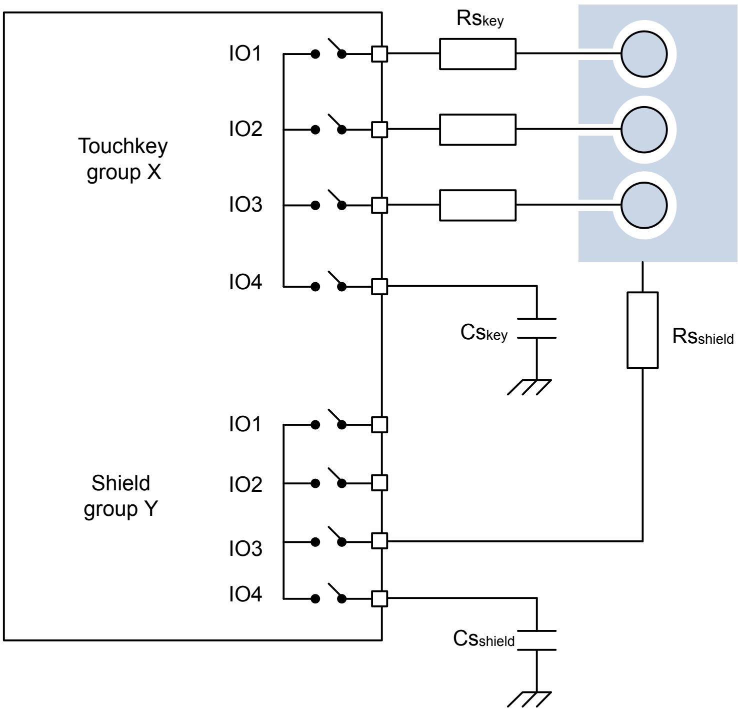


Table 11 gives a list of documents containing information about the active shield.

Table 11. Active shield documentation

Id	Title	Chapters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	Active shield
AN4312	Design with surface sensors for touch sensing applications on MCUs	Driven shield
AN4316	Tuning a touch sensing application on MCUs	Shield adjustment
OLT	STM32L4 online training	Touch sensing controller (TSC)

5.6 Layout and PCB

Rules to follow to improve TSC systems

5.6.1 Led rules

Figure 12. Led layout example

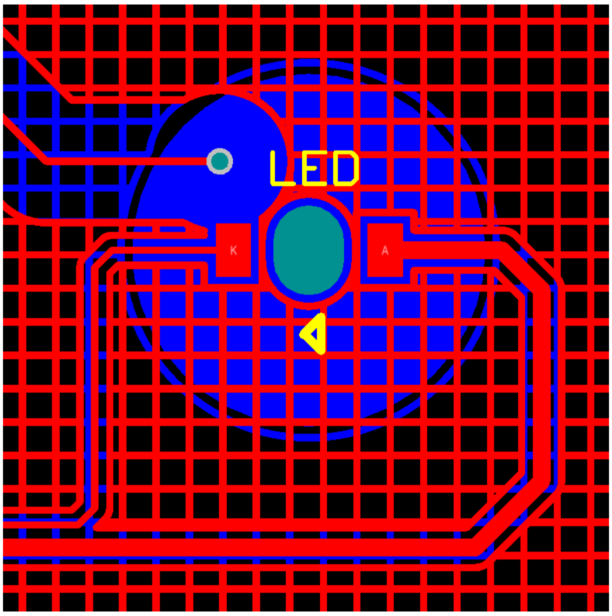


Figure 13. Example of cases requiring a LED bypass capacitor

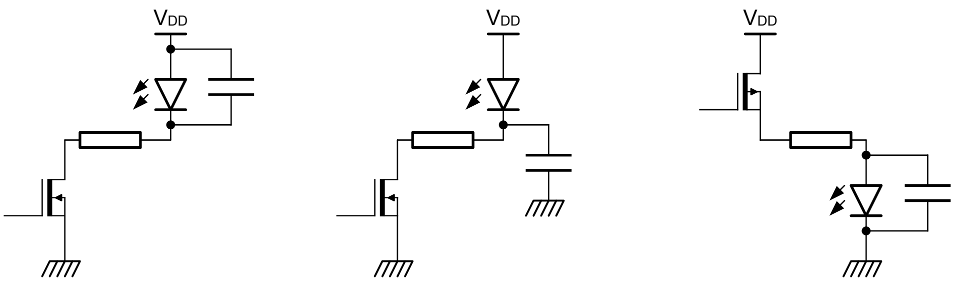


Table 12 gives a list of documents containing information about led rules.

Table 12. Led rules documentation

Id	Title	Chapters
AN4312	Design with surface sensors for touch sensing applications on MCUs	• LEDs and sensors • Placing of LEDs close to sensor

5.6.2 Electrode not located on PCB

It is possible but it is not recommended, because when the electrode it is not located on the PCB, the sensitivity decreases and additional extra parasitic capacitances are added.

Figure 14. Electrode not located on PCB example

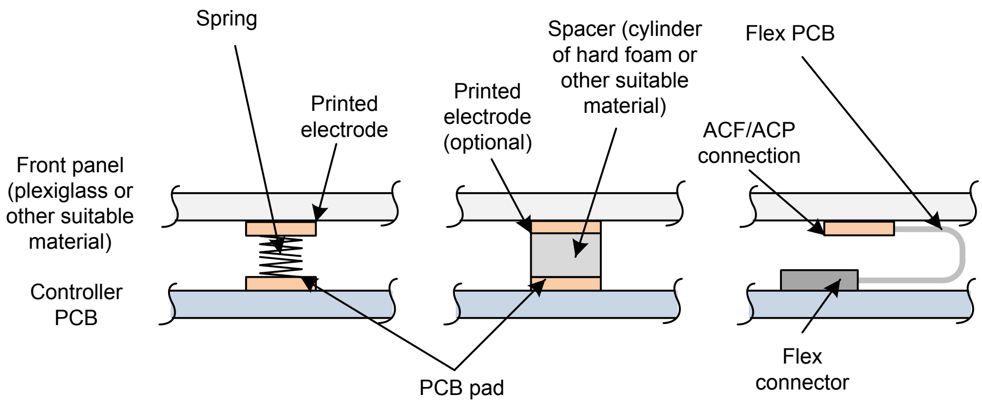


Table 13 gives a list of documents containing information about the electrode.

Table 13. Electrode documentation

Id	Title	Chapters
AN4312	Design with surface sensors for touch sensing applications on MCUs	Using electrodes separated from the PCB

5.6.3 Ground, shield and sensors

Table 14 gives a list of documents containing information about the layout.

Table 14. Layout documentation

Id	Title	Chapters
AN4312	Design with surface sensors for touch sensing applications on MCUs	• PCB and Layout • Ground considerations • Rotary and linear sensor recommendations

Figure 15 shows the ground plane and the signal tracks.

Figure 15. Hatched ground and signal tracks

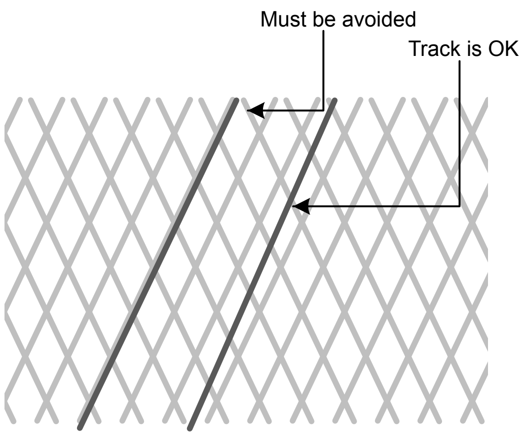


Figure 16. Ground plane example

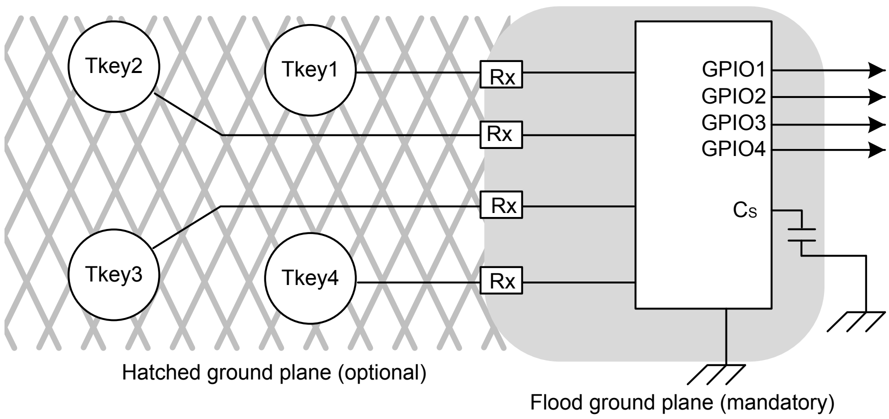


Figure 17. Track routing

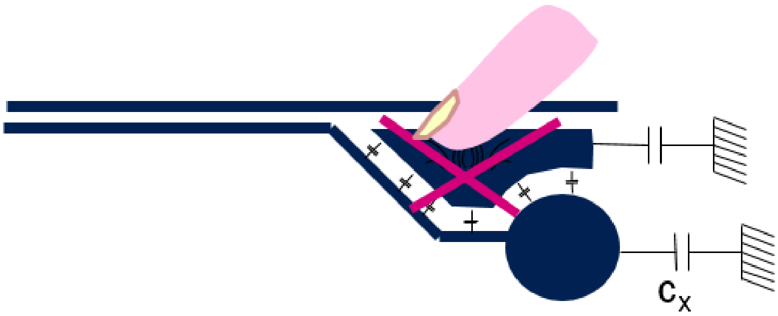


Figure 18. Track routing recommendation

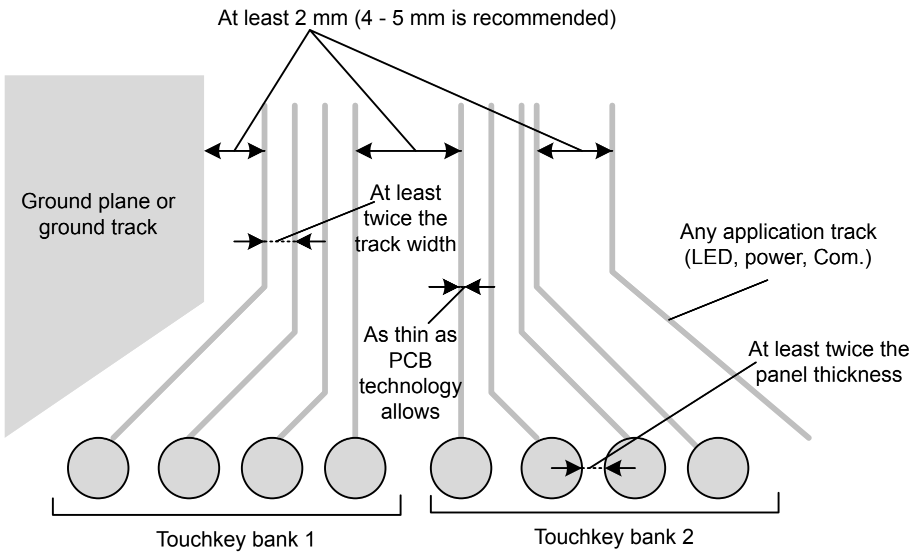
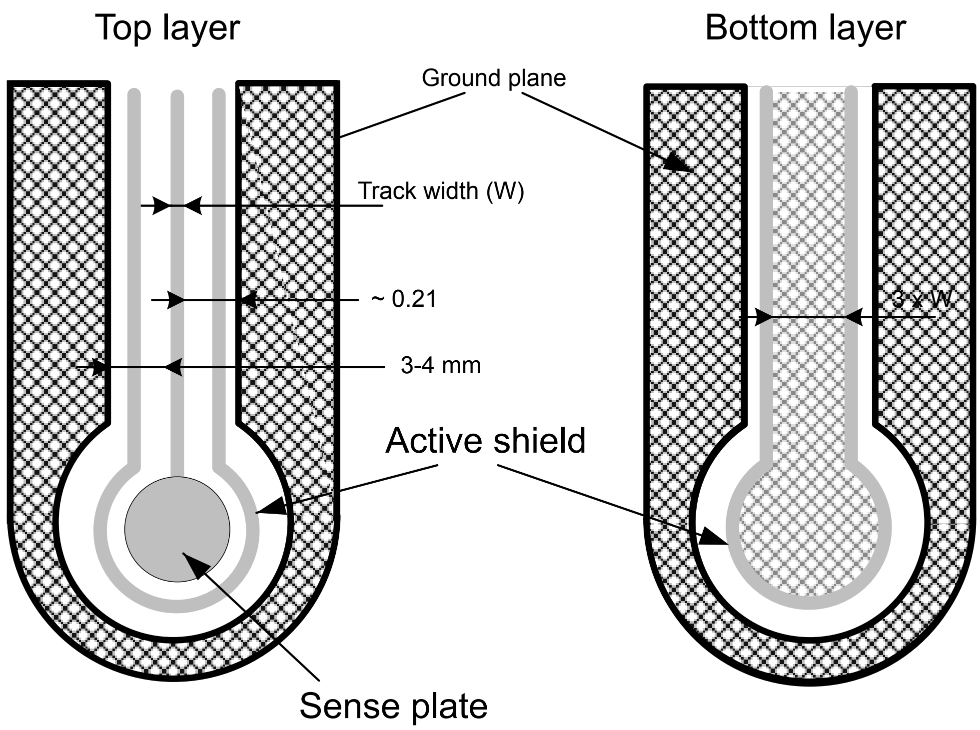


Figure 19. Shield



5.6.4 FAQ

System keys points:

- Direct connection between earth and board ground is required to avoid conducted noise issues.
- Conductive painting on the front panel must be avoided.
- Robust mechanical assembly is required.

Layout keys points:

- GND plane is mandatory under MCU, sampling capacitors and up to serial resistors
- Hatched GND plane recommended for sensor traces from both sides of the PCB:
  – minimize parasitic capacitance
  – mesh plane possible with 25% to 40% copper
- Route the sensors and ground on the same layer while the components and other tracks are routed on the other layers

Driven shield, or Active shield, is recommended.

- If there are LEDs close to the sensors, to indicate a touch event, they must be bypassed by a capacitor whose typical value is 10 nF.
- External LDO regulator should be used to power the MCU only to provide a stable power supply voltage without any ripple, especially all the switching components like transistors, LEDs, in the application must not be powered from the same voltage. This regulator should not be placed close to the sensors and their tracks, but close to the MCU.
- It is strongly recommended to dedicate pins to be used as touch sensors and do not share them with other features

R_S and C_S keys points:

- PPS or NPO sampling capacitors are recommended. Possible X5R or X7R.
- Never use tantalium sampling capacitors.
- Serial ESD 10 K (down to 1 K) resistors are recommended to be placed as close as possible to the MCU.
- No track crossing or via between these resistors and the MCU.
- The value of the sampling capacitor of the active shield should be different than the value of the sampling capacitors used for acquisition.
- The capacitance of active shield is higher (larger area) than C_X of a single touch sensing channel. In order to achieve the same waveform on active shield and active touch sensing channel, the ratios C_S/C_X of active shield and active touch sensing channel (touchkey). therefore, the CS of the active shield should also have higher value (k x C_S of touch sensing channel).

Sensor key points:

- Other traces must not cross the touch sensing traces or the whole touch sensing area
- The touch sensing traces should be as thin as technology allows and as short as possible.
  – No longer than 10 cm
- The space between traces and GND plane should be ideally 5 mm
- TC pins are more robust against external interference than FT:
- Consider modification of PCB layout to allow connection of external VDD clamping diode to touch sensing electrode traces.
  – Use low-capacitance diode like BAR18, BAS70 with Cmax = 2 pF.
  – In case it is later needed, add pads and connection to the PCB without assembling components.
- Floating panes must never be placed close to the sensors.

5.7 Noise

Noise is a key point for touch sensing applications. Noise can come from power supply.

5.7.1 Power supply

Main rules to follow:

- Place Buzzer and LED before LDO.
- Place LDO close to MCU.

Figure 20. Typical power supply schematic

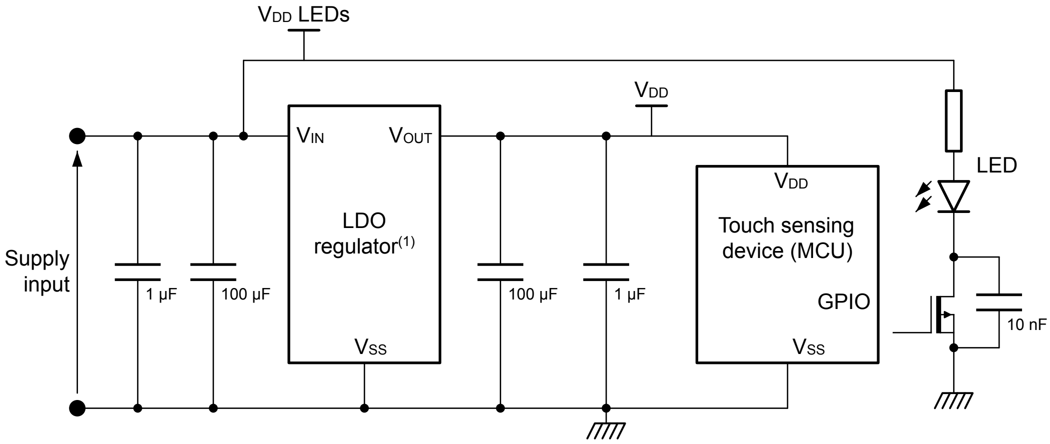


Table 15 gives a list of documents containing information about the power supply.

Table 15. Power supply documentation

Id	Title	Chapters
AN4312	Design with surface sensors for touch sensing applications on MCUs	Power supply

5.7.2 False detection

To avoid false detection TSL embed ECS, DXS and DTO algorithms.

Table 16 gives a list of documents containing information about the false detection.

Table 16. False detection documentation

Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	• Environment change system (ECS) – Power supply voltage, temperature and air humidity • Detection exclusion system (DXS) • Detection time out (DTO)

5.7.3 Noise immunity

Noise filtering can be done on hardware and software (TSL) sides.

Table 17 gives a list of documents containing information about the noise immunity.

Table 17. Noise immunity documentation

Id	Title	Chapters
UM1913	Developing applications on STM32Cube with STMTouch touch sensing library	Noise filters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	How to improve noise immunity
OLT	STM32L4 online training	Touch sensing controller (TSC)

5.7.4 Conducted noise

- Touch sensing systems require the conducted noise immunity.
- A key point is the signal to noise ratio (SNR).
- The test condition to be followed by the user is described in the standard IEC61000-4-6.

Table 18 gives a list of documents containing information about the conducted noise.

Table 18. Conducted noise documentation

Id	Title	Chapters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	All chapters

6 Tuning

For tuning purposes, dedicated application notes are available.

Sensors

Table 19 gives a list of documents containing information about the sensor.

Table 19. Sensors documentation

Id	Title	Chapters
AN4312	Design with surface sensors for touch sensing applications on MCUs	All chapters

ESD

Table 20 gives a list of documents containing information about the ESD.

Table 20. ESD documentation

Id	Title	Chapters
AN3960	ESD considerations for touch sensing applications on MCUs	All chapters

CN

Table 21 gives a list of documents containing information about the conducted noise.

Table 21. Conducted noise documentation

Id	Title	Chapters
AN4299	Improve conducted noise robustness for touch sensing applications on MCUs	All chapters

CS

Table 22 gives a list of documents containing information about the sampling capacitor.

Table 22. Sampling capacitor documentation

Id	Title	Chapters
AN4310	Sampling capacitor selection guide for touch sensing applications on MCUs	All chapters

7 Getting started TSC with STM32CubeMX

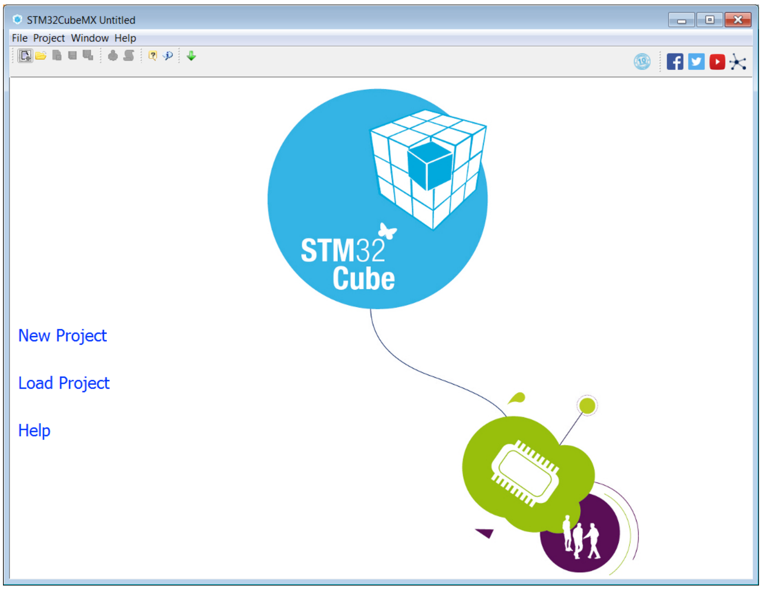
7.1 Uses cases

How to set up a TSC application based on TSL is explained in the following two examples. These examples describe the way to set up TLS on STM32F072B-DISCO and STM32L0538-DISCO discovery boards.

This description can be used as an example to set up other TSC series such us L4, F3, L0, L1 and L4+.

A STM32CubeMX new feature is available from version 4.24.0. This new feature can help to speed up TSL, TouchSensingLib, installation.

Figure 21. Main project panel



7.2 Discovery board: STM32F072B-DISCO

The STM32F072 Discovery kit helps the user to discover the STM32F072, which has the full set of features available in the STM32F0 series. The STM32F072 Discovery kit also helps the user to develop applications easily. It includes everything required for beginners and experienced users to get started quickly.

Based on the STM32F072RBT6, it includes an ST-LINK/V2 embedded debug tool interface, an ST MEMS gyroscope, LEDs, push-buttons, linear touch sensor, RF EEPROM connector and a USB mini-B connector.

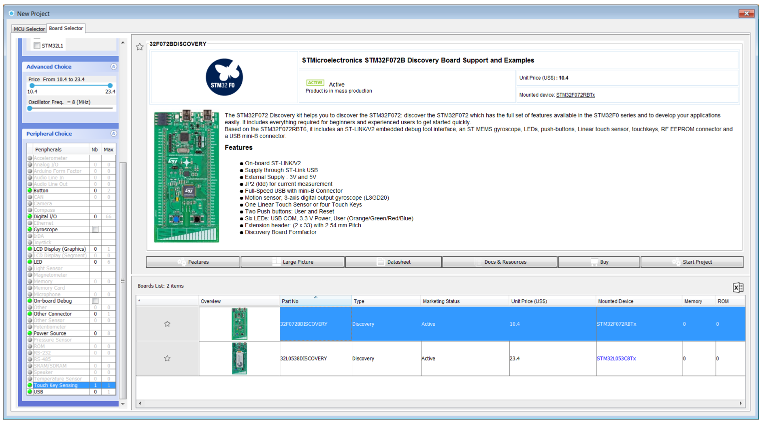
This discovery board provides a three channels linear (or slider) sensor. The main characteristics of these sensors are:

- On-board ST-LINK/V2
- Supply through ST-LINK USB
- External Supply: 3V and 5V
- JP2 (Idd) for current measurement
- Full-Speed USB with mini-B Connector
- Motion sensor, 3-axis digital output gyroscope (L3GD20)
- One Linear Touch Sensor or four Touch Keys
- Two push-buttons: User and Reset
- Six LEDs: USB COM, 3.3 V Power, User (Orange/Green/Red/Blue)
- Extension header: (2 x 33) with 2.54 mm pitch
- Discovery Board Formfactor

7.2.1 STM32F072B-DISCO board selection

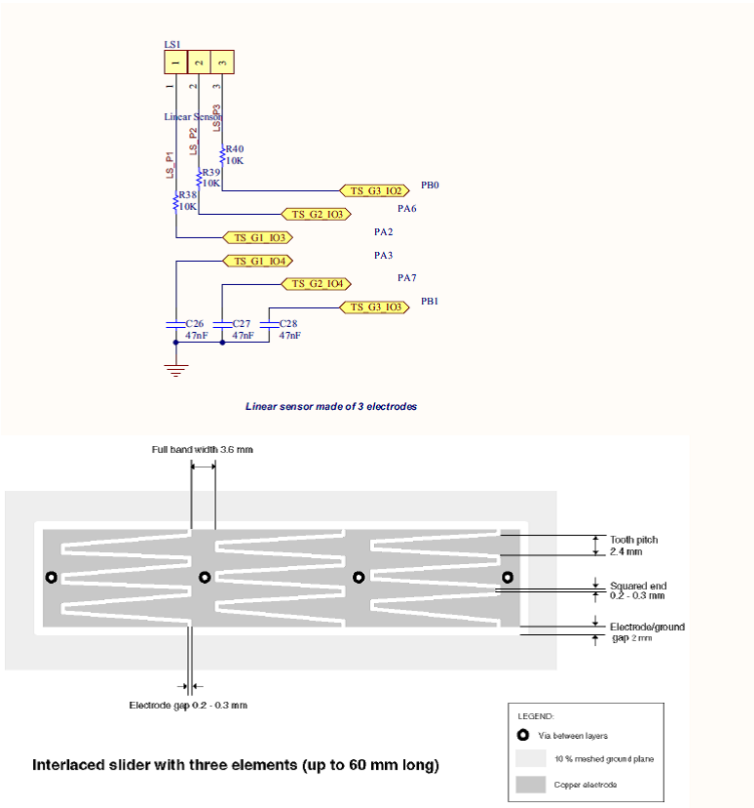
Start to select STM32F072B-DISCO board.

Figure 22. STM32F072B-DISCO board selection



To start Linear Touch Sensor channel acquisition at the same time, three groups are used. (See Figure 23).

Figure 23. STM32F072B-DISCO board schematics



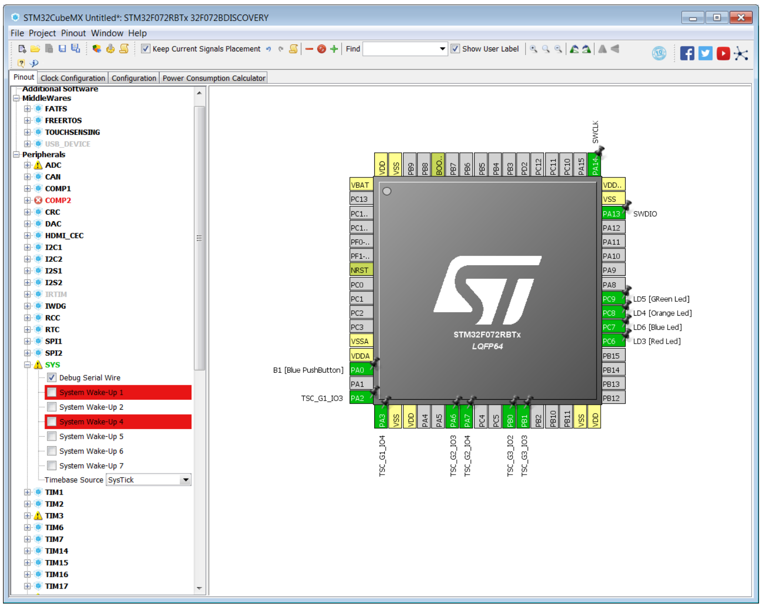
7.2.2 STM32F072B-DISCO TSC group and sensor activation

To activate the TSC group, sampling capacitors and sensor channels follows the below steps:

- activate TSC according schematics information.
- desactivate unrelevant peripheral like USB, SPI, NCF(L0), EPaper(L0), MFX(L0)

SWD peripheral must be set according to Figure 24.

Figure 24. STM32F072B-DISCO pinout SWD



TSC peripheral must be set according to Figure 25.

Figure 25. STM32F072B-DISCO pinout TSC

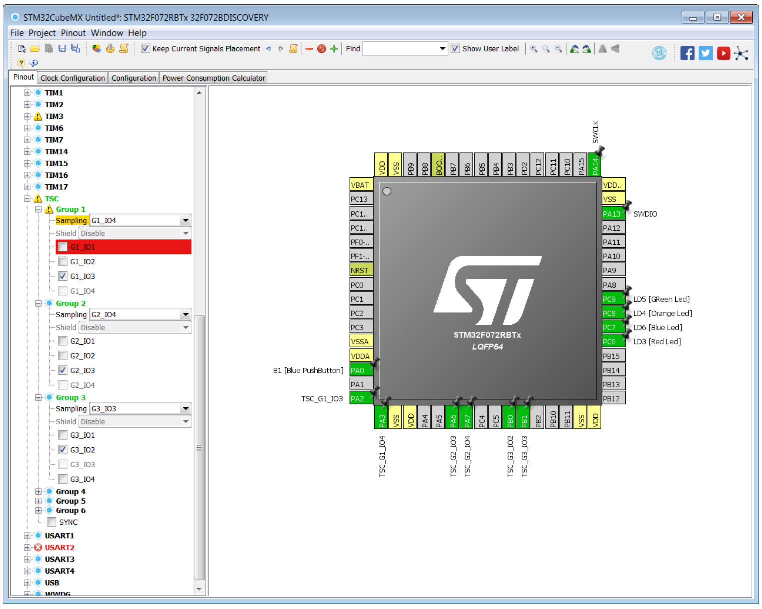
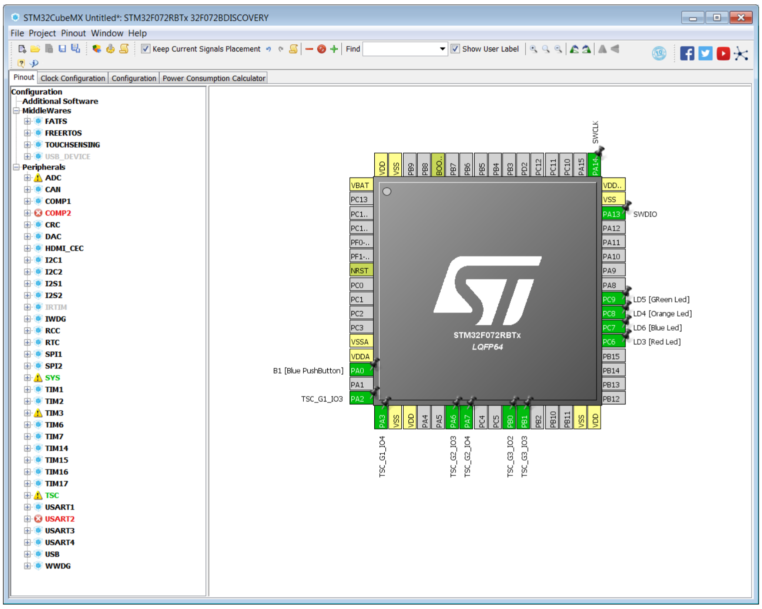


Figure 26 shows the results obtained.

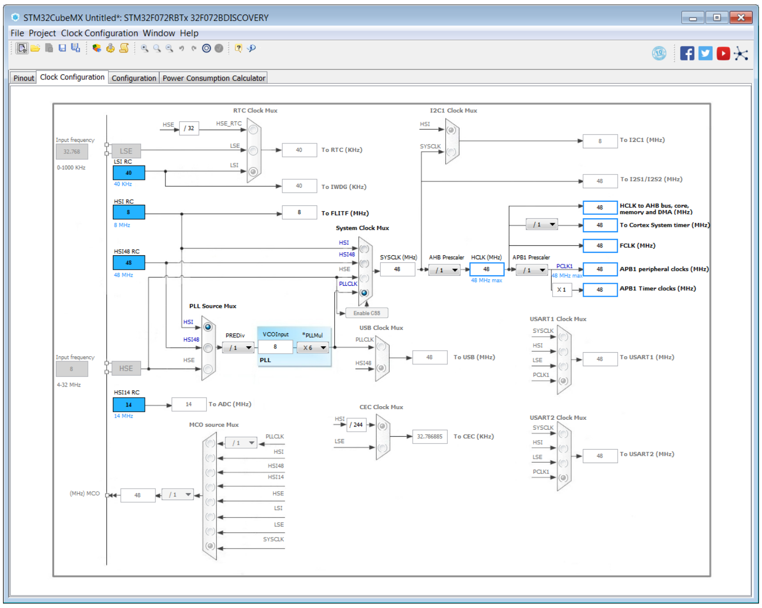
Figure 26. STM32F072B-DISCO pinout overview



7.2.3 STM32F072B-DISCO clock tree

It uses the default clock tree setting.

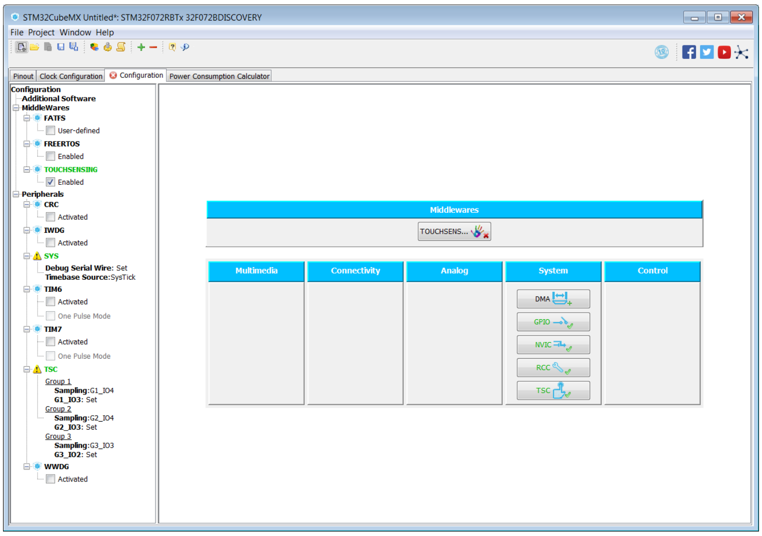
Figure 27. STM32F072B-DISCO clock configuration



7.2.4 STM32F072B-DISCO touchsensing library

To activate the TLS usage, switch on TOUCHSENSING box configuration.

Figure 28. TOUCHSENSING box configuration



Select three channels Linear slider and assign dedicated Gx_IOy.

- For training purpose, the user can use three channels Linear slider as three keys sensors
- Select three keys and assign dedicated Gx_IOy

Figure 29 to Figure 33 show these steps.

Figure 29. STM32F072B-DISCO sensor selection

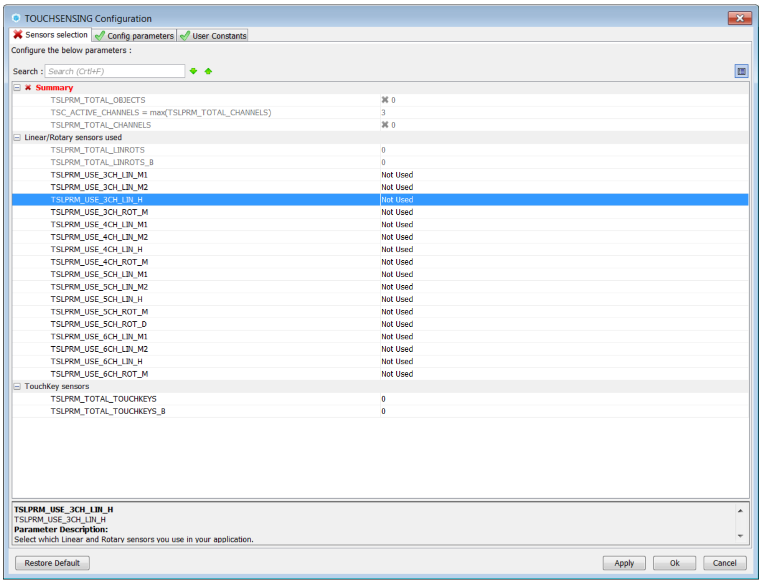


Figure 30. STM32F072B-DISCO sensor selection step2

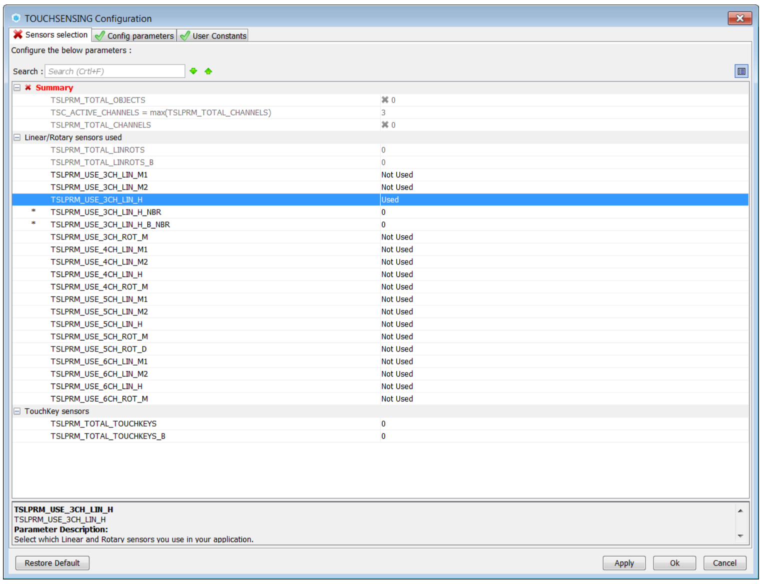


Figure 31. STM32F072B-DISCO sensor selection step3

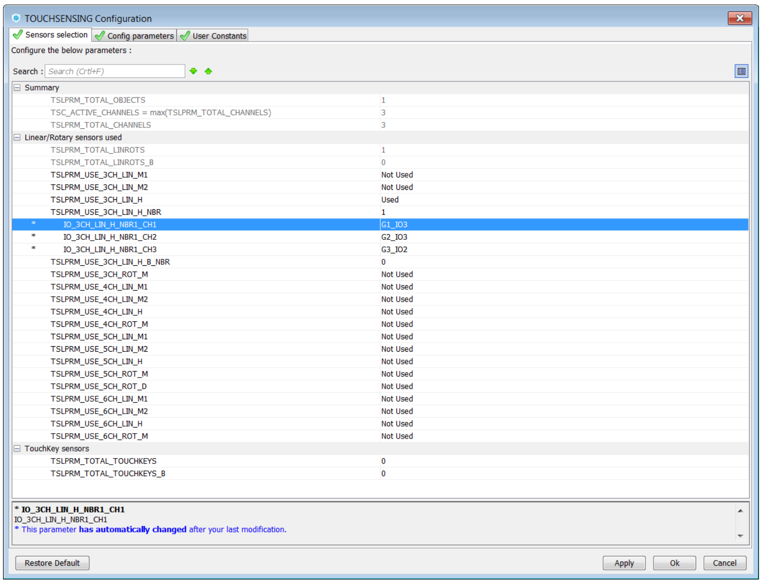


Figure 32. STM32F072B-DISCO sensor selection step4

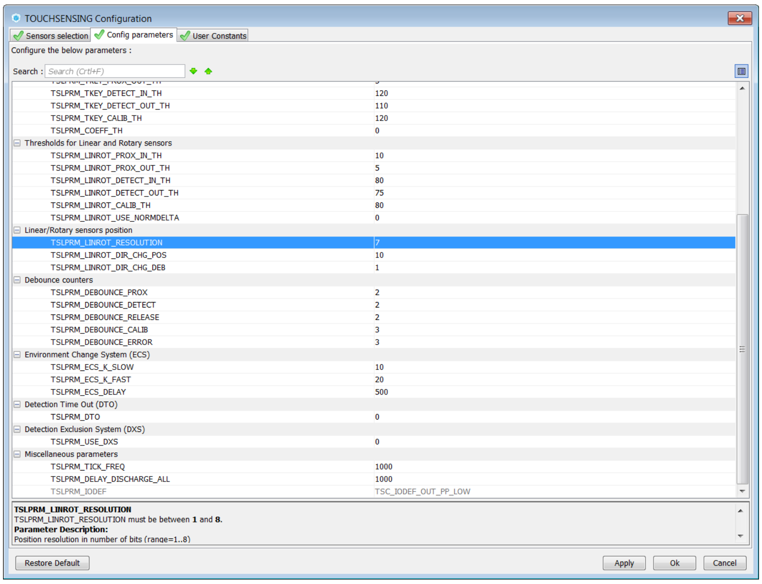
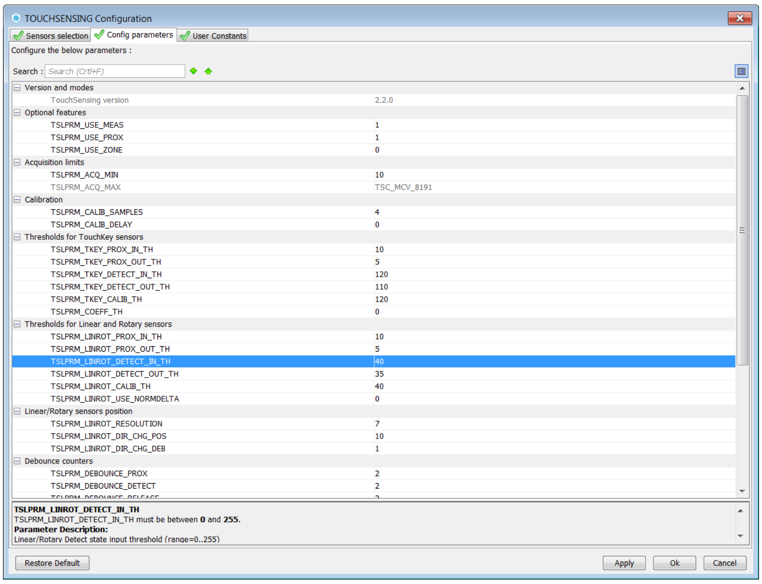


Figure 33. STM32F072B-DISCO sensor selection step5



7.2.5 STM32F072B-DISCO software project generation

It is possible to generate the complete software project based on TSC HAL and TSL. Figure 34 to Figure 37 show all these steps.

Figure 34. STM32F072B-DISCO software generation step1

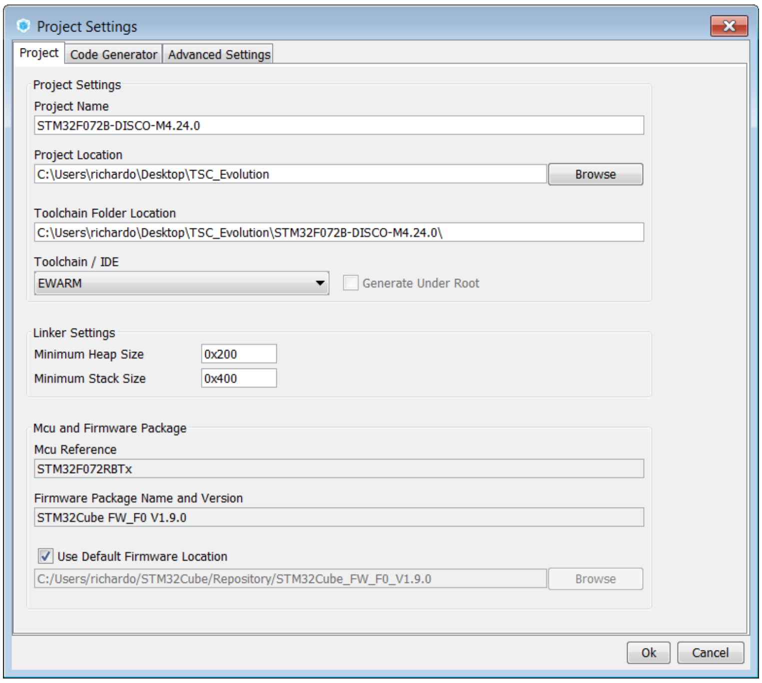


Figure 35. STM32F072B-DISCO software generation step2

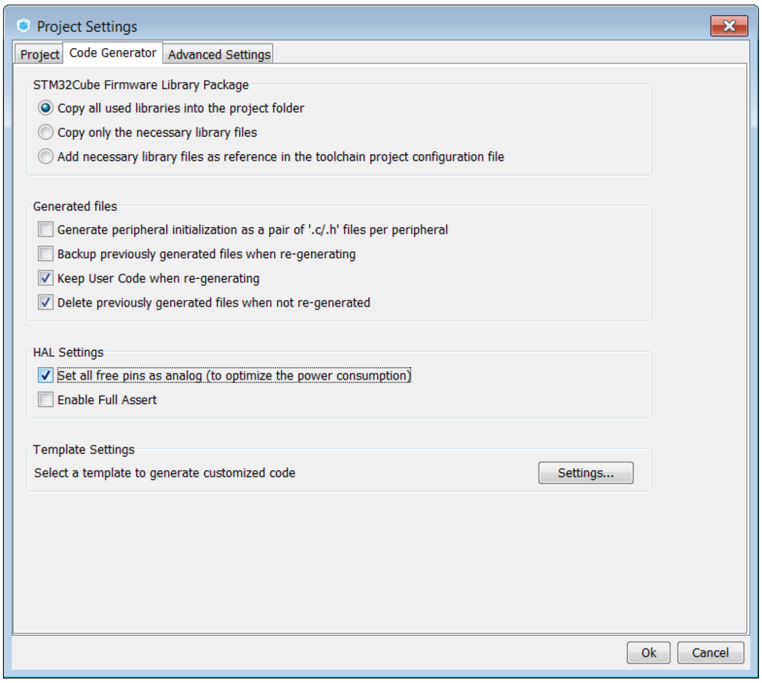


Figure 36. STM32F072B-DISCO software generation step3

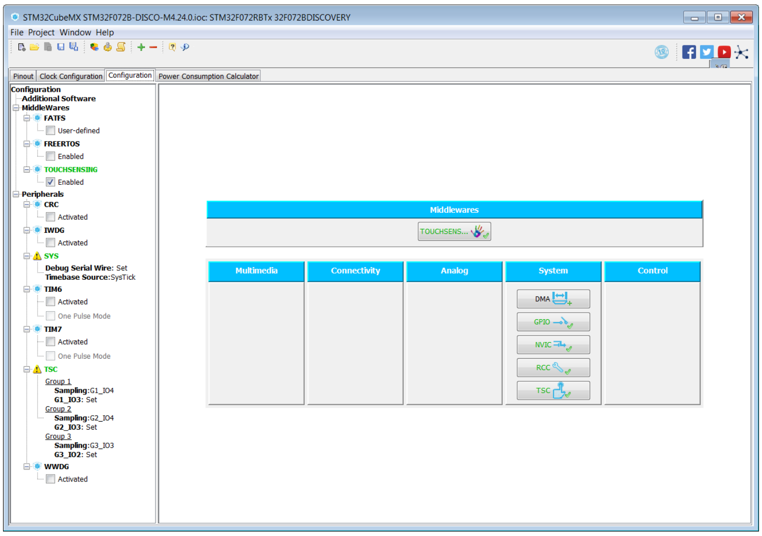
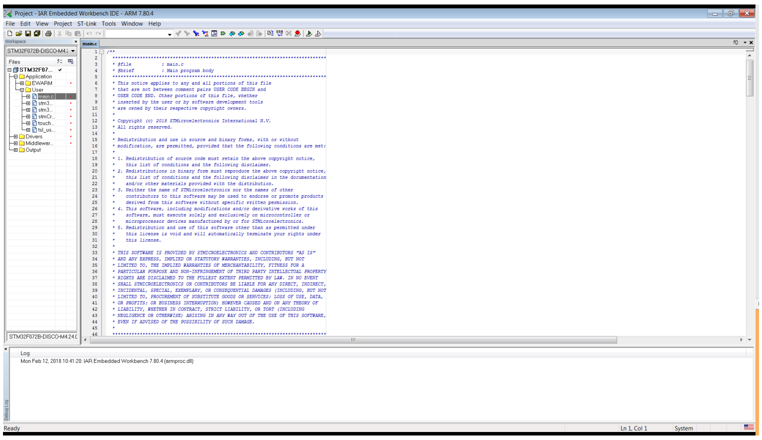


Figure 37. STM32F072B-DISCO IDE workspace



7.2.6 STM32F072B-DISCO software basic algorythm

The user needs to write the main application loop.

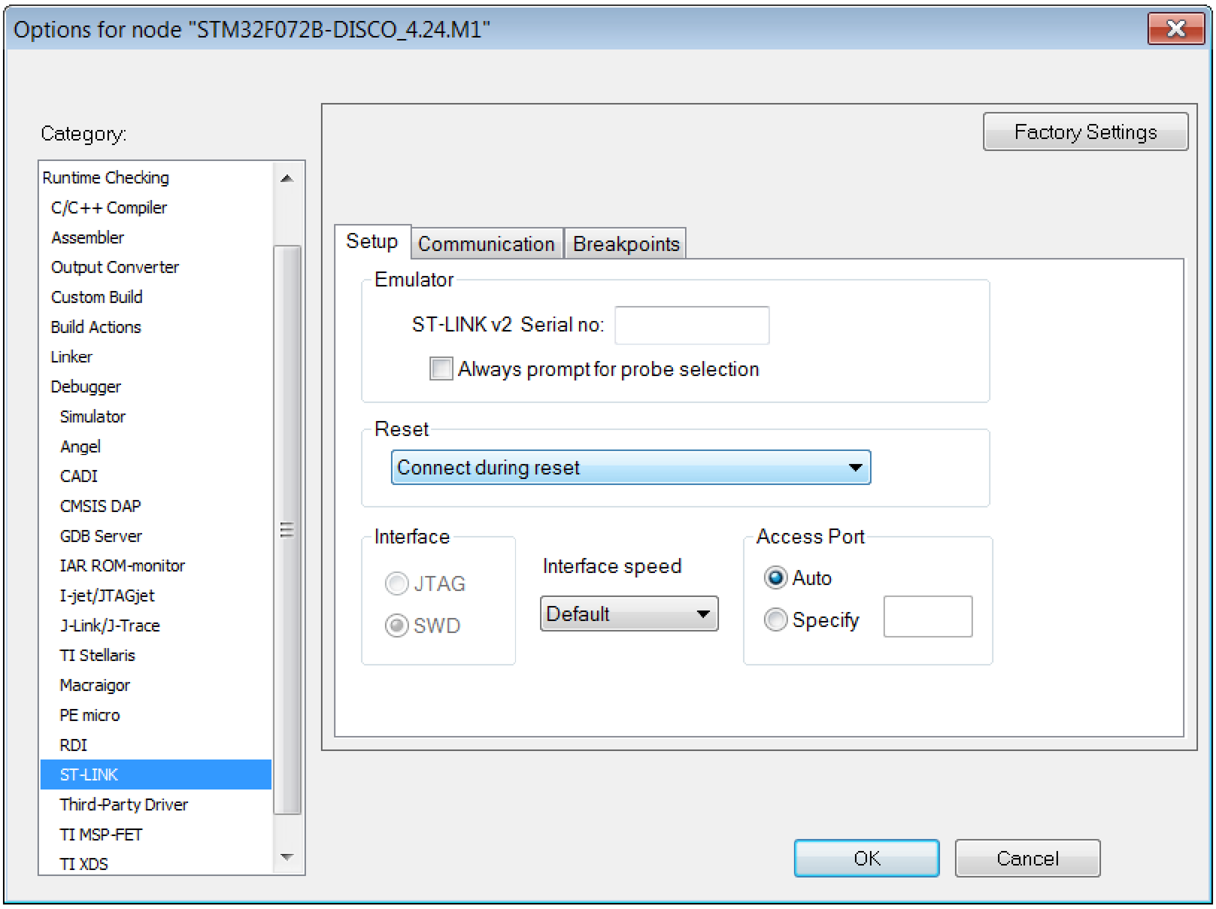
Example to show keys usage instead of slider usage.

- Open the IDE and in `main.c` file add the following lines:

```
/* USER CODE BEGIN 3 */
    extern TSL_LinRot_T MyLinRots[];
    static uint32_t    cnt=0;
    tsl_user_status_t status = TSL_USER_STATUS_BUSY;
    status = tsl_user_Exec();
    if(TSL_USER_STATUS_BUSY == status)
    {
      // Nothing to do
      if(cnt++%50==0){
        HAL_GPIO_TogglePin(LD3_GPIO_Port, LD3_Pin);
      }
      HAL_Delay(1);
    }
    else
    {
      if(MyLinRots[0].p_Data->StateId == TSL_STATEID_DETECT)
      {
        //TSLPRM_LINROT_RESOLUTION
        if(MyLinRots[0].p_Data->Position >= 5 && MyLinRots[0].p_Data->Position < 50)
        {
          HAL_GPIO_WritePin(LD4_GPIO_Port, LD4_Pin, GPIO_PIN_SET);
          HAL_GPIO_WritePin(LD6_GPIO_Port, LD6_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(LD5_GPIO_Port, LD5_Pin, GPIO_PIN_RESET);
        }
        if(MyLinRots[0].p_Data->Position >= 50 && MyLinRots[0].p_Data->Position < 80)
        {
          HAL_GPIO_WritePin(LD6_GPIO_Port, LD6_Pin, GPIO_PIN_SET);
          HAL_GPIO_WritePin(LD4_GPIO_Port, LD4_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(LD5_GPIO_Port, LD5_Pin, GPIO_PIN_RESET);
        }
        if(MyLinRots[0].p_Data->Position >= 80 && MyLinRots[0].p_Data->Position < 120)
        {
          HAL_GPIO_WritePin(LD5_GPIO_Port, LD5_Pin, GPIO_PIN_SET);
          HAL_GPIO_WritePin(LD4_GPIO_Port, LD4_Pin, GPIO_PIN_RESET);
          HAL_GPIO_WritePin(LD6_GPIO_Port, LD6_Pin, GPIO_PIN_RESET);
        }
      }
      else //if(MyLinRots[0].p_Data->StateId == TSL_STATEID_RELEASE)
      {
        HAL_GPIO_WritePin(LD4_GPIO_Port, LD4_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(LD5_GPIO_Port, LD5_Pin, GPIO_PIN_RESET);
        HAL_GPIO_WritePin(LD6_GPIO_Port, LD6_Pin, GPIO_PIN_RESET);
      }
    }
  }
  /* USER CODE END 3 */
```

To take care of ST-Link setup, see Figure 38.

Figure 38. STM32F072B-DISCO setup



The system is functional and ready to be used. The led blinks according to finger position on the slider.

7.3 Discovery board: STM32L0538-DISCO

The STM32L053 discovery kit helps you to discover the ultra-low-power microcontrollers of the STM32L0 series. It offers everything required for beginners and experienced users to get started quickly and develop applications easily.

Based on an STM32L053C8T6, it includes an ST-LINK/V2-1 embedded debug tool interface, linear touch sensor, IDD current measurement, 2.04” E-paper display, NFC connector for PLUG-CR95HF-B board, LEDs, push-buttons, and a USB Mini-B connector.

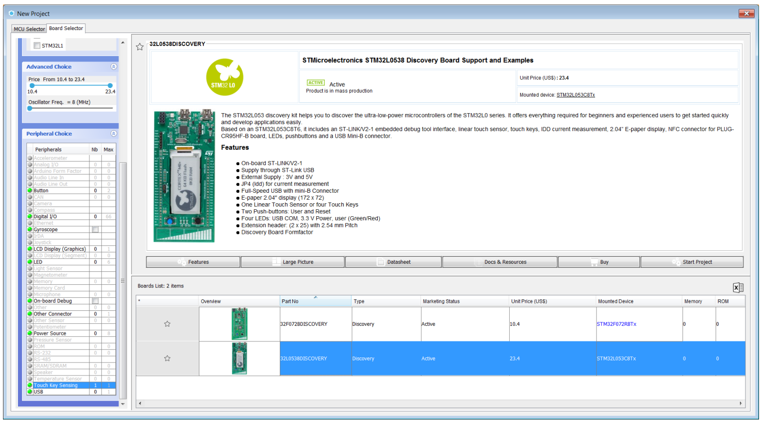
This discovery board provides a three channels linear (or slider) sensor. Their main characteristics are:

- On-board ST-LINK/V2-1
- Supply through ST-LINK USB
- External Supply : 3V and 5V
- JP4 (Idd) for current measurement
- Full-Speed USB with mini-B Connector
- E-paper 2.04" display (172 x 72)
- One Linear Touch Sensor or four Touch Keys
- Two push-buttons: User and Reset
- Four LEDs: USB COM, 3.3 V Power, user (Green/Red)
- Extension header: (2 x 25) with 2.54 mm pitch
- Discovery Board Formfactor

7.3.1 STM32L0538-DISCO board selection

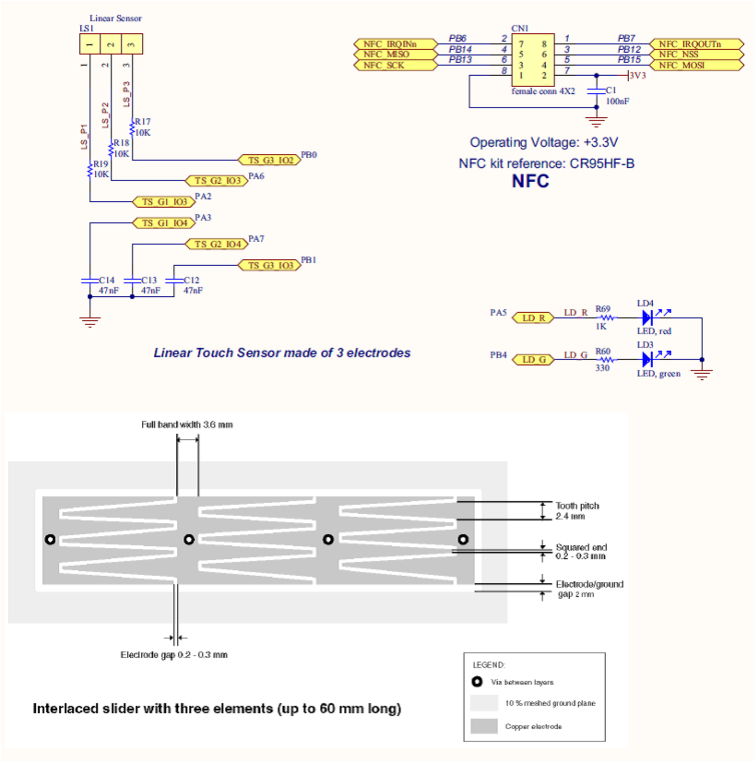
Start to select STM32L0538-DISCO board.

Figure 39. STM32L0538-DISCO board selection



To start linear touch sensor channel acquisition at the same time, three groups are used.

Figure 40. **STM32L0538-DISCO board schematics**



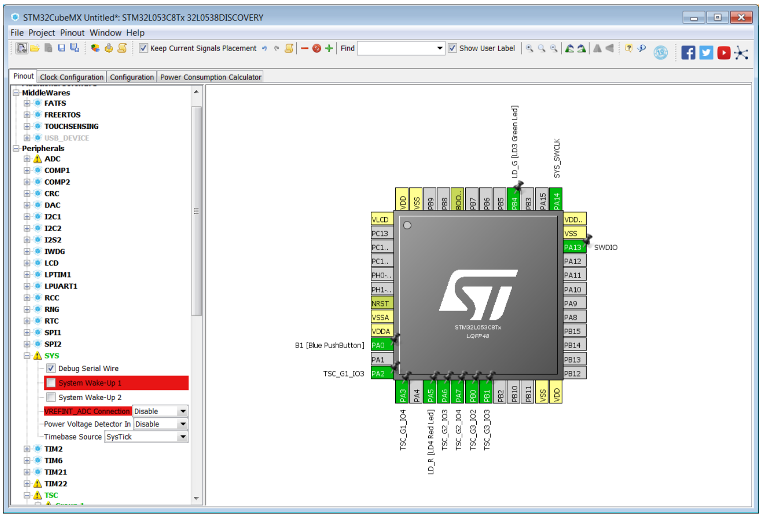
7.3.2 STM32L0538-DISCO TSC group and sensor activation

To activate the TSC group, sampling capacitors and sensor channels follows the below steps:

- Activate TSC according schematics information.
- You can deactivate unrelevant peripheral like USB, SPI, NCF(L0), EPaper(L0), MFX(L0)

SWD peripheral must be set according to Figure 41.

Figure 41. Pinout SWD



TSC peripheral must be set according to Figure 42.

Figure 42. Pinout TSC

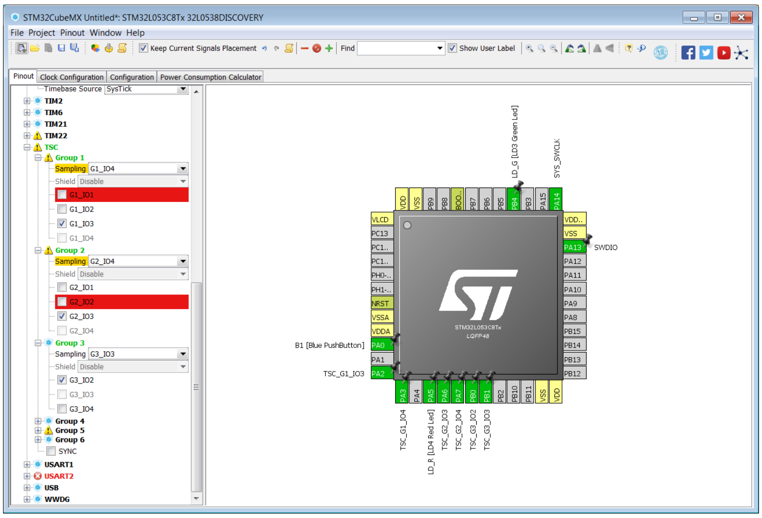
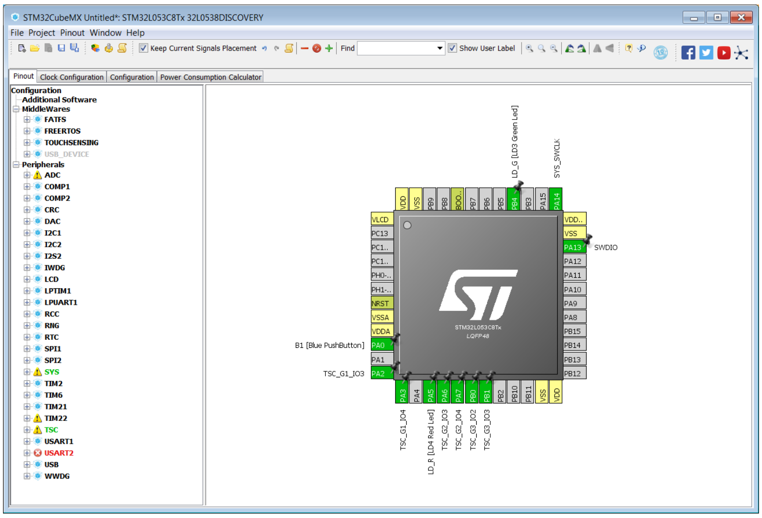


Figure 43 shows the results obtained.

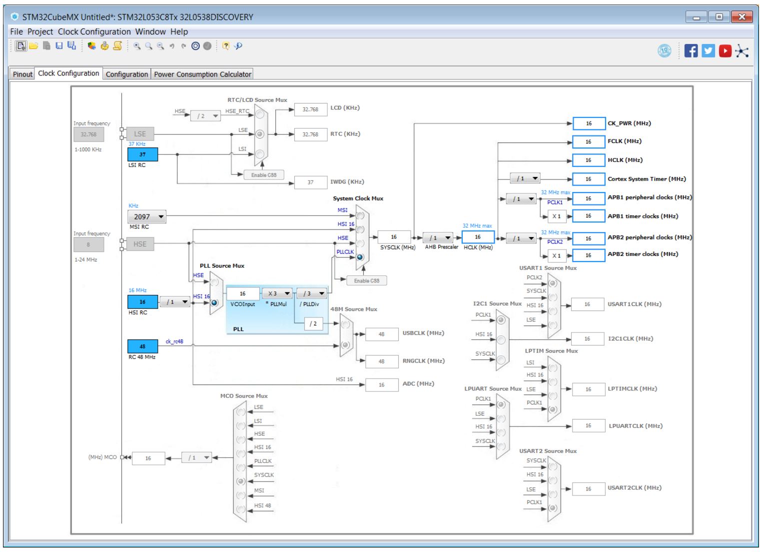
Figure 43. Pinout overview



7.3.3 STM32L0538-DISCO clock tree

It uses the default clock tree setting.

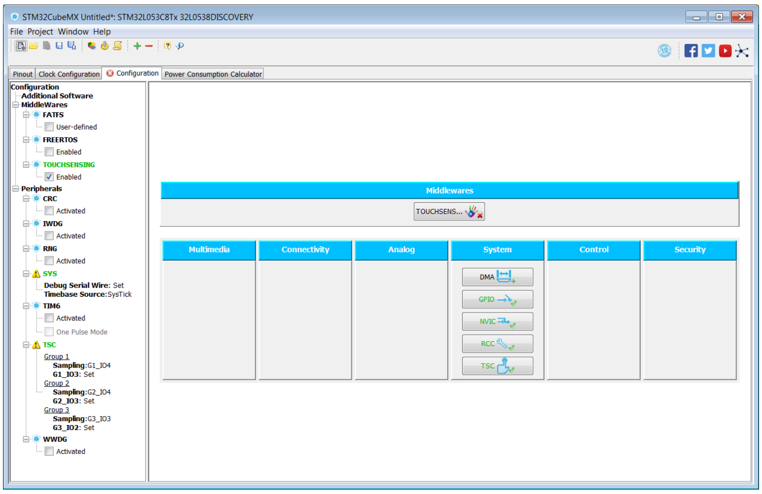
Figure 44. Clock configuration



7.3.4 STM32L0538-DISCO touchsensing library

To activate the TLS usage, switch on TOUCHSENSING box configuration.

Figure 45. TOUCHSENSING box configuration

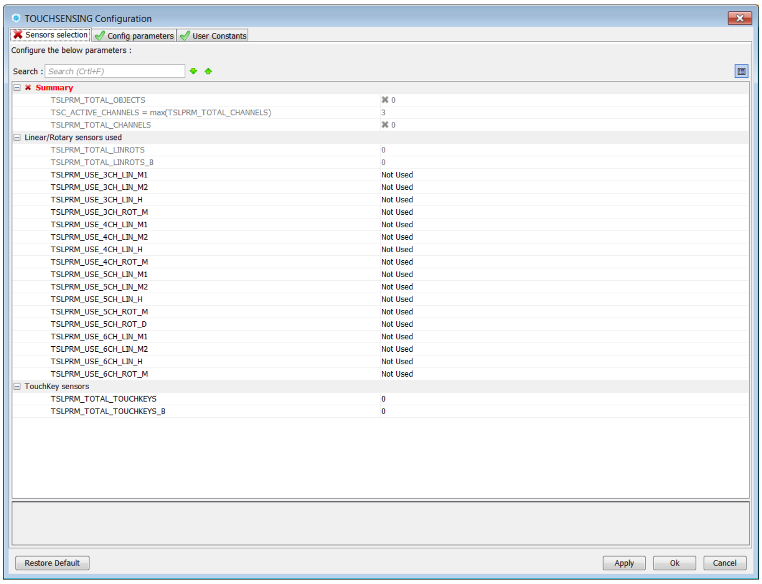


Select three channels Linear slider and assign dedicated Gx_IOy.

- For training purpose, the user can use three channels linear slider as three keys sensors
- Select three keys and assign dedicated Gx_IOy

Follow Figure 46 to Figure 50 to set sensors.

Figure 46. STM32L0538-DISCO sensor selection step1



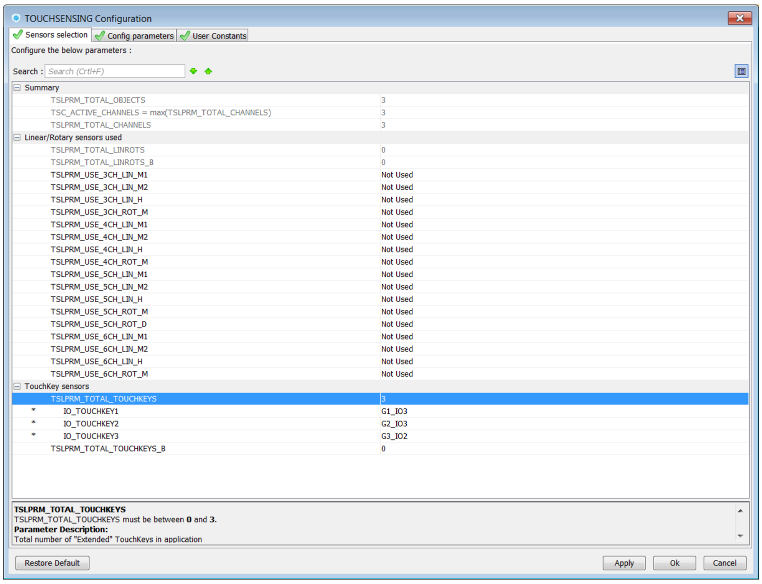


Figure 47. STM32L0538-DISCO sensor selection step2

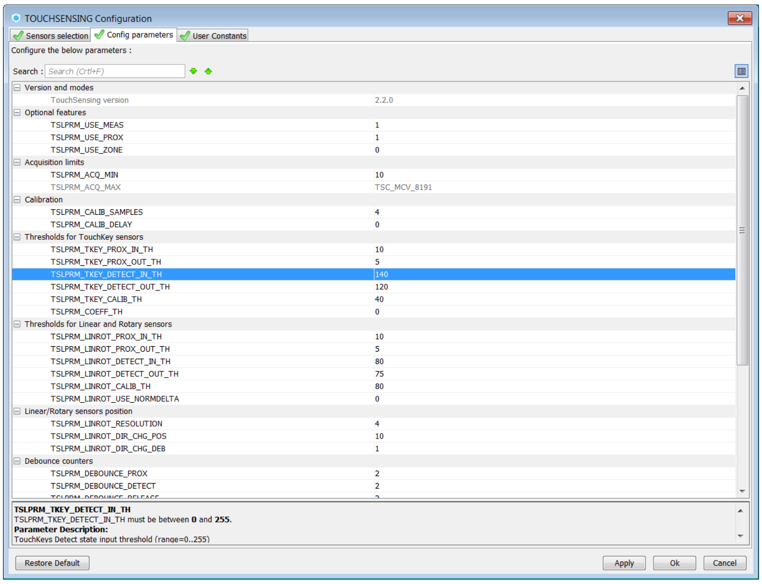


Figure 48. STM32L0538-DISCO sensor selection step3

Figure 49. STM32L0538-DISCO sensor selection step4

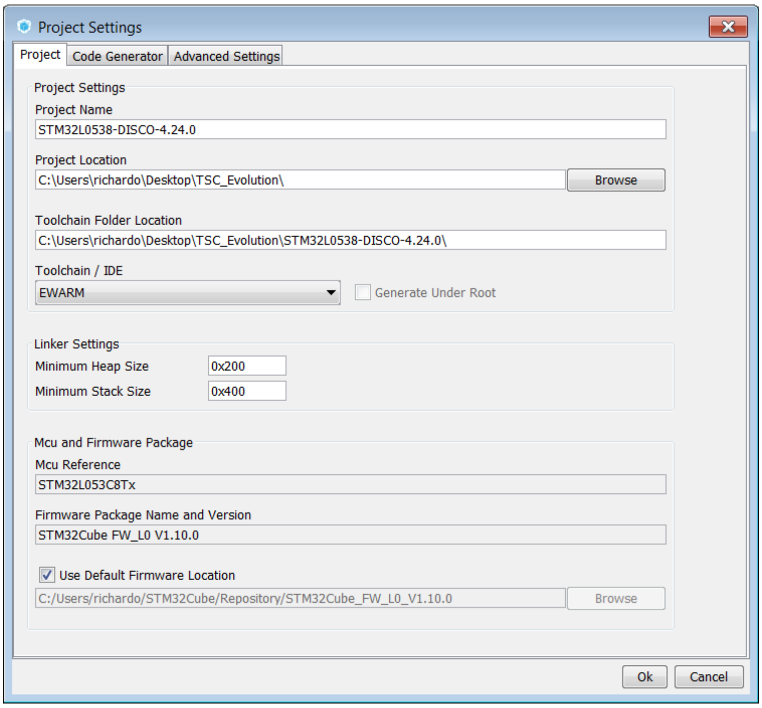
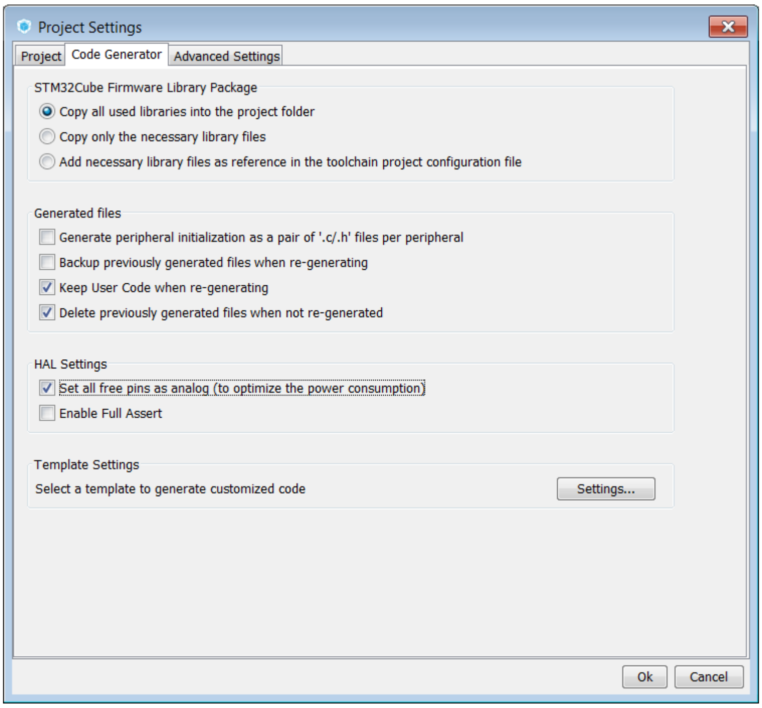


Figure 50. STM32L0538-DISCO sensor selection step5



7.3.5 STM32L0538-DISCO software project generation

It is possible to generate the complete software project based on TSC HAL and TSL.

See details in Figure 51 to Figure 55.

Figure 51. STM32L0538-DISCO software generation step1

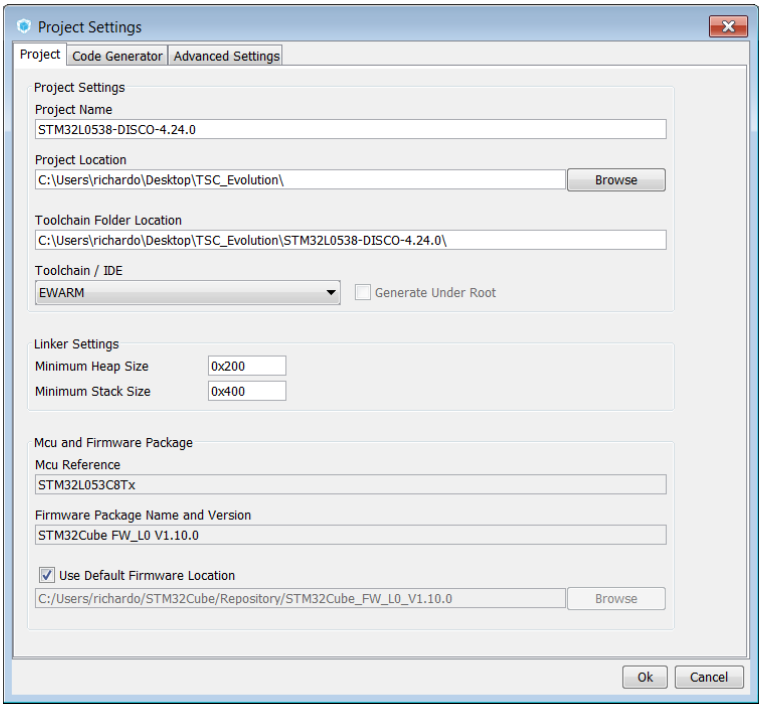


Figure 52. STM32L0538-DISCO software generation step2

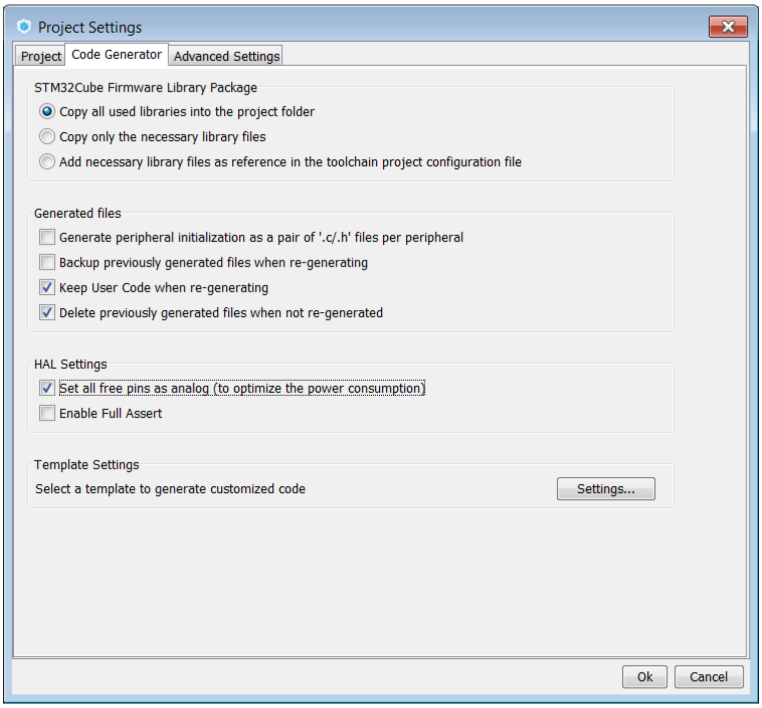


Figure 53. STM32L0538-DISCO complete project overview

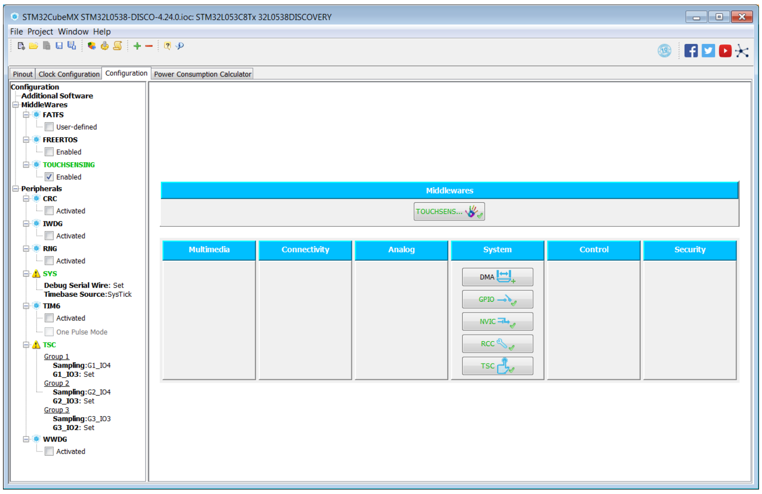


Figure 54. STM32L0538-DISCO IDE workspace

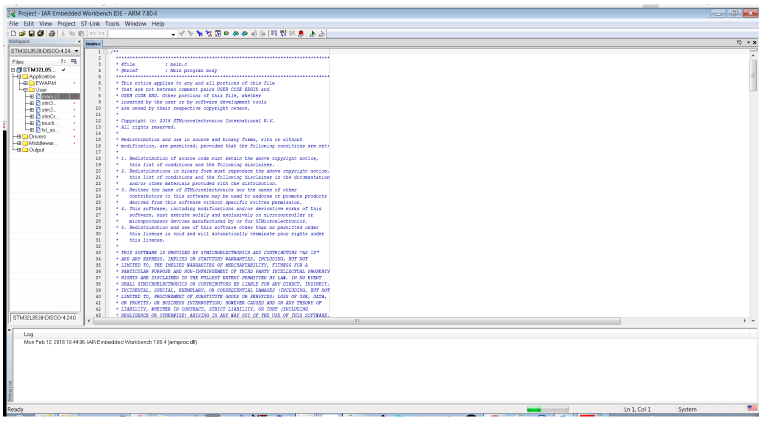
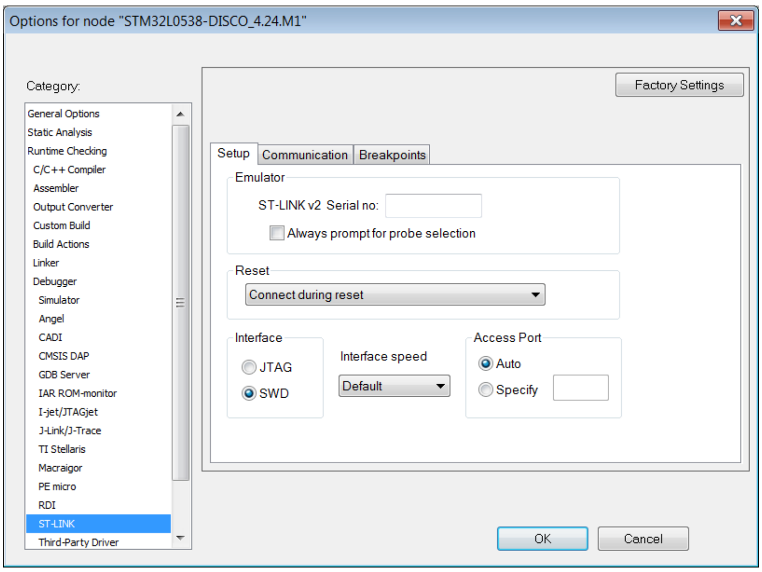


Figure 55. SWD settings



7.3.6 STM32L0538-DISCO software basic algorithm

Below is showed an example to show keys usage instead of slider usage.

- Open the IDE and in `main.c` file add the following lines:

```
/* USER CODE BEGIN 3 */
  extern TSL_TouchKey_T MyTKeys[];
  static uint32_t    cnt=0;
  tsl_user_status_t status = TSL_USER_STATUS_BUSY;

  status = tsl_user_Exec();
  if(TSL_USER_STATUS_BUSY == status)
  {
    // Nothing to do
    if(cnt++%50==0){
    }
    HAL_Delay(1);
  }
  else
  {
    HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, GPIO_PIN_RESET);    //00
    HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, GPIO_PIN_RESET);
    if(MyTKeys[0].p_Data->StateId == TSL_STATEID_DETECT)
    {
        HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, GPIO_PIN_SET); //11
        HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, GPIO_PIN_SET);
    }
    if(MyTKeys[1].p_Data->StateId == TSL_STATEID_DETECT)
    {
        HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, GPIO_PIN_SET); //01
        HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, GPIO_PIN_RESET);
    }
    if(MyTKeys[2].p_Data->StateId == TSL_STATEID_DETECT)
    {
        HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, GPIO_PIN_RESET);//01
        HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, GPIO_PIN_SET);
    }
  }
}
/* USER CODE BEGIN 3 */
```

The system is functional and ready to be used.

The Led is blink according to the position of the on slider.

Revision history

Table 23. **Document revision history**

Date	Version	Changes
19-Sep-2018	1	Initial release.
23-Jul-2021	2	Added: • STM32L4+, STM32L5, STM32U5, and STM32WB series • Table 1. Applicable products Updated: • Section Introduction • Table 2. Change transfer principle documentation • Table 3. Reference documentation • Table 4. Signal threshold use documentation • Table 5. Charge transfer documentation • Table 6. Sensitivity documentation • Table 8. Key documentation • Table 9. Linear touch sensor documentation • Table 10. Rotary sensor documentation • Table 11. Active shield documentation • Table 12. Led rules documentation • Table 13. Electrode documentation • Table 14. Layout documentation • Table 15. Power supply documentation • Table 17. Noise immunity documentation • Table 18. Conducted noise documentation • Section 6 Tuning
16-Jan-2023	3	Updated: • Section Introduction to incorporate the STM32WBA series. • the whole document with minor changes.

Contents

List of tables

List of figures