Hello, and welcome to this presentation of the STM32 Touch Sensing Controller (TSC) which enables the designer to simply add touch sensing functionality to any application.
Over recent years, Touch Sensing has become quite common in many applications such as mobile phones, induction cooktops and ovens, coffee machines, etc. This type of interface is more flexible and reliable compared to standard push buttons because mechanical parts are no longer needed.

The Touch Sensing Controller (TSC) embedded in STM32WB devices offers a simple way to manage such interfaces. The TSC supports a robust charge transfer acquisition principle with up to 21 capacitive sensing channels. It is fully configurable and only a few external components are required to design a user-friendly interface.
The key features of the Touch Sensing Controller are:

- Proven charge transfer acquisition principle which is available on several STM32 MCU series (STM32F0, STM32F3, STM32L0 and STM32L4).
- Supports up to 21 capacitive sensing channels which are split over 7 analog I/O groups. The number of channels and I/O groups depends on the selected MCU.
- For optimum performance, up to 7 capacitive sensing channels can be acquired in parallel. This offers a very good response time.
- Only one sampling capacitor is needed to manage up to 3 capacitive sensing channels. This ensures a reduced BOM.
- The charge transfer acquisition is fully managed by hardware to reduce CPU overhead. A spread spectrum feature is available to improve system robustness in noisy environments.
- Finally, the Touch Sensing Controller is designed to
operate with the free STM32Cube touch sensing library available in the corresponding STM32Cube package. This library offers all the processing required to develop a robust capacitive sensing solution and supports proximity, touchkey, linear and rotary touch sensors.
To offer sufficient clock granularity, the TSC is directly clocked using the AHB clock. This clock is used by the spread spectrum block while the clock feeding the pulse generator is reduced using a prescaler. GPIOs supporting touch sensing must be configured in an alternate mode in order to connect them to the Touch Sensing Controller. The SYNC input pin is used to synchronize the capacitive sensing acquisition with an external stimulus without the need for CPU interaction. One counter per analog I/O group is used to store the result of the acquisition. An interrupt can be generated upon the end of acquisition of all the enabled analog I/O groups or when an error is detected. This interrupt helps limit CPU overhead.
The charge transfer acquisition technique works using the electrical properties of the capacitor. It consists in charging the sensor capacitor \( C_X \) to \( V_{DD} \). Once this capacitor is fully charged, a part of the capacitor's charge is transferred to a sampling capacitor \( C_S \). Charge is transferred using analog switches directly embedded into the GPIO.

The charge transfer cycle is repeated \( N \) times until the voltage on the sampling capacitor reaches the \( V_{IH} \) threshold of the GPIO it is connected to.

The number \( N \) of transfer cycles required to reach the threshold represents the size of \( C_X \). The number of transfer cycles decreases when the sensor is touched.

The charge transfer is performed through analog switches embedded in the GPIO.
The measuring circuit is simple. Let’s consider one analog I/O group composed of four I/Os. One of these I/Os is the sampling capacitor I/O. It is connected to an external capacitor \( (C_S) \) commonly called the sampling capacitor. There is a single sampling capacitor per analog I/O group. The sampling capacitor value depends on the channel sensitivity. The higher the \( C_S \), the higher the sensitivity and the longer the acquisition time.

The three other I/Os are dedicated to channels. Each of them is connected to a sensor electrode through a serial resistor \( (R_S) \). \( R_S \) is used to improve the ESD robustness of the application. Within one analog I/O group, only one channel is acquired at a time. This means that if three channels are implemented, three consecutive acquisitions will be required to get the image of the three sensors. For optimum performance, the sensor capacitance should be as low as possible. We often consider this capacitor value to be in the tens of picofarads. A touch leads to an increase in the
sensor capacitance by a few picofarads, for example 5 picofarads.
The charge transfer acquisition sequence is composed of 7 steps.
First, the sampling capacitor and the sensor capacitor are discharged to get a stable starting point by closing the analog switch S1 and enabling S3. Between each major step, an intermediate step is inserted to avoid an acquisition artifact. This step, called dead time, consists of opening all active analog switches and disabling all active transistors.
Next, the sensor capacitor (C_X) is charged to V_DD by closing S2.
After the dead time, a portion of the charge accumulated in C_X is transferred into the sampling capacitor C_S by closing the analog switch S1.
Once the charge is transferred, the voltage on C_S (V_CS) is read. If the voltage is lower than V_{IL}, a logical ‘0’ is returned. If it is greater than V_{IH}, a logical ‘1’ is read. If the returned logical value is ‘0’, Steps 3 to 7 are repeated. After each charge transfer loop, a counter is incremented, representing
the capacitance of the sensor.
In order for a Touch Sensing GPIO to be controlled by the TSC:

- A sampling capacitor I/O must be configured in alternate output open-drain mode. In addition, to avoid artifacts, the Schmidt trigger hysteresis must be disabled.
- A channel I/O must be configured in alternate output push-pull mode.
To reduce the CPU load, two acquisition modes are supported:

- Normal acquisition mode where the acquisition starts by setting the START bit of the TSC_CR register.
- Synchronized acquisition mode where the acquisition only starts upon the detection of a falling or rising edge and a high level on the SYNC input pin. This mode is useful to limit the effect of noise in some applications such as an induction cooktop.

In both modes, the end of acquisition and/or max count error can be managed either by polling or interrupt.
In addition to managing the charge transfer capacitive acquisition principle, the TSC peripheral allows the designer to individually control the analog switch and Schmidt trigger hysteresis of I/Os belonging to analog I/O groups. This capability could be useful to implement a different capacitive sensing acquisition principle or for other purposes such as an analog multiplexer.
The TSC peripheral offers two interrupt sources:

- **End of acquisition** which notifies the CPU when all the active channels are acquired.
- **Max count error** which is set when the acquisition fails on one or several channels. It is useful in preventing an infinite acquisition which can occur in the event of a hardware failure.

<table>
<thead>
<tr>
<th>Interrupt event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>End of acquisition</td>
<td>Set when an acquisition is completed.</td>
</tr>
<tr>
<td>Max count error</td>
<td>Set when a max count error occurs. The management of the max count error avoids getting into an infinite acquisition in case of channel issues (C₃ in shortcircuit,...).</td>
</tr>
</tbody>
</table>
The touch sensing controller is active in Run, Sleep, Low-power run and Low-power sleep modes. This means that charge transfer acquisition can only be performed in these modes. In all others modes (Stop 0, Stop 1, Stop 2, Standby and Shutdown), the touch sensing controller is not operational. In Stop modes, the peripheral is frozen but the registers content is kept. In Standby and Shutdown modes, the registers content is lost and the peripheral must be reinitialized.
This example details a solution with 6 touchkeys and one linear touch sensor. It is important to note that a dedicated voltage regulator is used and that the solution also uses the active shield for optimum conducted noise robustness. A COG type capacitor is used for the sampling capacitors as they offer a good stability over the temperature range and no memory effect.
A voltage regulator used to power the touch controller is strongly recommended. It will minimize the measurement noise induced by power supply variations.

For optimum sensitivity, the parasitic capacitance to ground must be minimized. This implies short and thin sensing tracks. Serial resistors (Rs) and sampling capacitors (Cs) must be placed as close as possible to the MCU. The sensing tracks driving sensors which are acquired at the same time should be grouped together (bank) and kept separated from others banks. Finally, a bypass capacitor should be used in case of high impedance drive (i.e. LED driven through an open drain circuitry) to ensure a low impedance path.

For optimum conducted noise performance, we recommend to use an active shield around tracks and sensor pads combined with spread spectrum.

Finally, conductive paint must be avoided and a stable mechanical assembly is required to avoid false or spurious touch detection.
touch detections.
As previously indicated, the TSC peripheral is designed to operate with the touch sensing library. This free C library supports proximity, touchkey, linear and rotary touch sensors. It allows combining capacitive sensing functions with traditional MCU features such as LCD drive, communication with a host device, … This library offers all the processing required to get optimum sensitivity and to design a robust application. Some of the features include power-on calibration, environment control system (ECS), debounce filtering and a detection exclusion system (DxS). This library offers a simple API to configure the channels, sensors and to get the state of the sensors. It is MISRA compliant and it supports all STM32 C compilers.
The STM32Cube touch sensing library is composed of several modules. The library relies on the corresponding STM32 series HAL and it is configured through a dedicated configuration file. Once included into your project, the STM32Cube touch sensing library is part of the overall application and each C function can be launched to get the appropriate behavior. For further details on the STM32Cube touch sensing library, please refer to the corresponding user manual.
This is a list of peripherals related to the STM32 Touch Sensing Controller. Please refer to these peripheral trainings for more information if needed.
For more details, please refer to application notes AN4299, AN4310, AN4312 and AN4316.

Thank you.