Hello and welcome to this presentation of the STM32 independent watchdog. It covers the main features of this peripheral which can be used either as a watchdog to reset the device when a problem occurs, or as a free-running timer for application timeout management.
The independent watchdog is used to detect and resolve malfunctions due to software failures.

It triggers a reset sequence when it is not refreshed within the expected time-window. Since its clock is an independent 32-kHz low-speed internal RC oscillator (LSI), it remains active even if the main clock fails. Once enabled, it forces the activation of the low-speed internal oscillator, and it can only be disabled by a reset.

One of the main benefits for applications is its ability to run independently from the main clock.
The independent watchdog offers a wide range of timeout values: from 125 microseconds to 32 seconds. It is clocked by a 32-kHz RC oscillator which cannot be disabled when the independent watchdog is enabled. It generates a reset when the programmed timeout value elapses, or when a watchdog refresh occurs outside a programmed time-window. This window feature is optional, and not present in all independent watchdogs.

It is possible to enable automatically the independent watchdog after a system reset. It is possible to define the behavior of the independent watchdog in Debug, Stop or Standby mode.
The independent watchdog registers are located in the CORE voltage domain while its functions are in the VDD voltage domain.
Two clocks are needed:
The APB clock is required in order to access registers
The LSI clock is required for the functional part of the watchdog

This architecture allows the independent watchdog to work even in Stop and Standby modes.
A programmable 8-bit prescaler is used to divide the LSI oscillator frequency.
The 12-bit downcounter defines the timeout value.
The IWDG is clocked by the 32 kHz RC oscillator named LSI. When the IWDG is enabled, the LSI oscillator is enabled automatically. The option bits allow the adjustment of the IWDG behavior according to the application needs:

- Possibility to automatically enable or not the IWDG after a reset via the IWDG_SW bit. This is often called the hardware or software activation mode.
- Possibility to freeze the IWDG when the product is in Stop mode via the IWDG_STOP bit.
- Possibility to freeze the IWDG when the product is in Standby mode (IWDG_STDBY).

It is possible to select if the watchdog will freeze or not when the product is in Debug mode (core halted).
This diagram illustrates how the independent watchdog operates. When the downcounter reaches zero, the watchdog reset is activated. This happens when the application software did not refresh the window watchdog on time.

If the software refreshes the watchdog while the downcounter is greater than the value stored in WIN[11:0], then a reset is generated as well. To prevent a watchdog reset, the refresh must occur when the downcounter value is higher than zero, and lower than the time-window value WIN[11:0].
Configuring IWDG hardware start

- In IWDG hardware start, the IWDG is automatically enabled after a system reset.
- After a system reset, the reload counter is set to 0xFFF and the prescaler to 0.
  ➤ Timeout value of about 0.5 seconds
- Key register (IWDG_KR) must be written by the software with 0x0000AAAA at regular intervals before the counter reaches 0 and within the window (if the window option is enabled)

The independent watchdog hardware is enabled by the device’s option bytes. If the hardware mode is enabled, after every system reset, the watchdog automatically loads the down-count with 0xFFF, and starts to count down. The prescaler is set to 0, providing a division by 4 on the input clock.

To prevent any reset, the Key register must be refreshed at regular intervals before the counter reaches 0 and within the time window, if this option has been selected.

Considering that the LSI is at exactly 32 kHz, the application has about 0.5 seconds to refresh the IWDG before the generation of a watchdog reset.
The independent watchdog software start is configured in only a few steps.

- The first step is to write the Key register with value 0x0000 CCCC which starts the watchdog.
- Then remove the independent watchdog register protection by writing 0x0000 5555 to unlock the key.
- Set the independent watchdog prescaler in the IWDG_PR register by selecting the prescaler divider feeding the counter clock.
- Write the reload register (IWDG_RLR) to define the value to be loaded in the watchdog counter.

After accessing the previous registers, it is necessary to wait for the IWDG_SR bits to be reset in order to confirm that the registers have been updated.

- Two options are now available: enable or disable the
independent watchdog window option.

- To enable the window option, write the window value in the IWDG_WINR register.
- Otherwise, refresh the counter by writing 0x0000 AAAA in the Key register to disable the window option.
The IWDG time base is prescaled from the LSI clock at 32 kHz. The IWDG_PR prescaler register can divide the LSI clock frequency by 4 up to 256. The watchdog counter reload value is a 12-bit value written in the IWDG_RLR register.

A formula can be used determine the independent watchdog timeout. The independent watchdog time is based on the LSI period and its prescaler, as well as the selected watchdog counter reload value.

Note that the Reset and Clock Controller (RCC) of the product provides registers giving the source of the reset. In that way the application can check if a reset is caused by an independent watchdog.
The IWDG can be active in all modes, except in Shutdown mode.
When the product exists from Shutdown, the IWDG register are set to their initial values.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run</td>
<td>Active*</td>
</tr>
<tr>
<td>Low-power run</td>
<td>Active*</td>
</tr>
<tr>
<td>Sleep</td>
<td>Active*</td>
</tr>
<tr>
<td>Low-power sleep</td>
<td>Active*</td>
</tr>
<tr>
<td>All Stop modes</td>
<td>Active*</td>
</tr>
<tr>
<td>Standby</td>
<td>Active*</td>
</tr>
<tr>
<td>Shutdown</td>
<td>Not working. The IWDG is reset when exiting from Shutdown</td>
</tr>
</tbody>
</table>

* If IWDG enabled