How to use Low-power timer (LPTIM) on STM32 MCUs and MPUs

Introduction

This application note describes the various modes and specific features of the low-power timer (LPTIM) embedded in the STM32 microcontrollers (MCUs) and microprocessors (MPUs) listed in the table below.

This document contains some applicative examples provided with:

- the X-CUBE-LPTIMER Expansion Package which includes:
  - asynchronous pulse counter in Stop mode
  - PWM (pulse-width modulation) generation in Stop mode
  - timeout wakeup mode
- the STM32CubeU5 MCU Package firmware for STM32U5 Series which includes:
  - PWM generation in Stop mode with LPBAM (low-power background autonomous mode)
  - input capture in Stop mode with Autonomous mode

These use cases demonstrate the importance of the low-power timer and clarify the features that differ from the general-purpose timer.

Table 1. Applicable products

Type	Series and product lines
Microcontrollers	STM32F410 and STM32F413/423 product lines
	STM32F7 Series, STM32G0 Series, STM32G4 Series, STM32H5 Series, STM32H7 Series, STM32L0 Series, STM32L4 Series, STM32L4+ Series, STM32L5 Series, STM32U5 Series, STM32WB Series, STM32WL Series
Microprocessors	STM32MP1 Series

1 General information

This document applies to STM32 MCUs and MPUs based on the Arm® Cortex®-M processor.

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

2 Overview

Low-power techniques aim to reduce the device current consumption and extend the battery lifetime. This is done by optimizing the consumption during both run-time and idle-time. The low-power timer (LPTIM) helps to reduce the power consumption specifically while the system is in low-power mode.

The STM32 LPTIM allows the system to perform simple tasks while the power consumption is kept at an absolute minimum. The LPTIM main features are listed below:

- a 16-bit auto-reload register to set the period, a 16-bit compare register to set the duty-cycle for a PWM waveform signal output on the timer
- a repetition counter that allows the adjustment of the counter roll-over
- channels that can be functional both as an input or output
- Input-capture mode
- capability to generate DMA requests to retrieve the input-capture counter values and to reprogram part of the LPTIMER without the intervention of the CPU
- ability to remain fully functional in Stop mode thanks to the Autonomous mode

The LPTIM can be used for timing and for output generation while the STM32 device is in low-power mode. The LPTIM also features a wide diversity of clock sources that are used to stay active in most power modes, except in Standby and Shutdown modes.

Thanks to a flexible clock scheme, the LPTIM can be used as a pulse counter by running with an external clock source. The LPTIM can also wake up the system from low-power modes, and realize a “timeout function” with extremely-low power consumption.

The LPTIM provides the basic functions of the STM32 general-purpose timers with the advantage of a very-low power consumption. Additionally, when configured in Asynchronous counting mode, the LPTIM keeps running even when no internal clock source is active.

The LPTIM remains active both in Sleep and Stop modes, and can wake up the STM32 device from these modes. Conversely, in Standby or Shutdown mode, the LPTIM is powered down and must be completely reinitialized when the STM32 device exits any of these modes.

The tables below details the main features of the various LPTIM types and the peripheral implementation on STM32 devices.

Table 2. Features on various LPTIM types

Features	LPTIM		
	Type 1	Type 2	Type 3
Auto-reload register	X	X	X
Compare register	X	X	X
Repetition counter	-	X	X
Input/output channels	-	-	X
Input-capture mode	-	-	X
DMA requests	-	-	X

Table 3. LPTIM types on STM32 devices

LPTIM type	STM32 devices[1]
Type 1	STM32F410 and STM32F413/423 product lines STM32F7 Series, STM32G0 Series, STM32G4 Series, STM32H7 Series, STM32L0 Series, STM32L4 Series[2] STM32L4R/4Sxx devices STM32MP1 Series, STM32WB Series
Type 2	STM32L41/42xx devices, STM32L4P5/4Q5xx devices STM32L5 Series, STM32WL Series
Type 3	STM32H5 Series, STM32U5 Series

1. *All features are not activated on all instances of a given product.*
2. *Except STM32L41/42xx devices.*

The LPTIM presents up to 16 external trigger sources with a configurable polarity. The external trigger inputs embed digital filters to cancel-out possible faulty triggers in noisy operating environments. The LPTIM can be configured to run:

- in Continuous mode that is used to generate PWM waveforms
- in One-shot mode that is used to generate pulse waveforms

The LPTIM features an Encoder mode. This function enables the LPTIM to interface with incremental quadrature encode sensors using the Input1 and Input2 of the peripheral. Both inputs feature a glitch-filtering circuitry.

The LPTIM can output different waveform types even when the STM32 device is in specific low-power modes whereas almost all internal clock sources are turned off. LPTIM_CMP (or LPTIM_CCRx for LPTIM type 3) and LPTIM_ARR registers, in conjunction with the WAVE bitfield in LPTIM_CFGR and SNGSTRT in LPTIM_CR, are used to control the output waveform.

The output waveform can be a typical PWM signal with its period and duty-cycle controlled by LPTIM_ARR and LPTIM_CMP (or LPTIM_CCRx) registers respectively, or the output waveform can be a single pulse with the last output state defined by the configured waveform.

If the output waveforms are not equal, the SetOnce mode is configured. The polarity of the LPTIM output is controlled through the WAVPOL bitfield in LPTIM_CFGR for LPTIM type ½, and through CCxP field in LPTIM_CCMRx for LPTIM type 3.

3 LPTIM versus general-purpose timer

The main LPTIM advantages compared to any other timer available in the STM32 MCUs are:

- The LPTIM can be fully functional in Stop mode thanks to the Autonomous mode and LPDMA.
- The LPTIM generates events to wake up the device from Stop mode.

A general-purpose timer (TIM) is active in Run and Sleep mode, while it is frozen in Stop mode: both the TIM state and the register contents are preserved. The TIM directly resumes operations when the device is woken up.

Depending on the selected clock source, the power consumption can be substantially lowered with LPTIM compared to a general-purpose timer. The table below details the difference between TIM and LPTIM peripherals during different power modes.

Table 4. TIM versus LPTIM in working modes

Peripheral	Run	Low-power run	Low-power Sleep	Stop
TIM	X	X	-	-
LPTIM	X	X	X	X

4 LPTIM clock sources

The LPTIM is a peripheral with two clock domains:

- APB (advanced peripheral bus) clock domain: contains the APB interface and the core functions of the peripheral such as the registers and signals connected to the CPU (typically the interrupt request)
- kernel clock domain: can be clocked by the APB clock source or by other internal clock sources including the LSE, LSI, MSIK and HSI sources. It can also be clocked from an external clock source through the input (Input1) of the timer.

The LPTIM main feature is its ability to keep running in low-power modes when almost all clock sources are turned off. The LPTIM also includes a very flexible clocking scheme enabling the features below:

- The LPTIM can be clocked from on-chip clock sources such as LSE, LSI, HSI, MSIK or APB clocks (refer to the product reference manual for more details).
- The LPTIM can be clocked from an external clock source through the Input1.

The flexible clocking scheme is used to build "Pulse-counter" applications. It is a key function for metering applications such as gas-meters. The table below summarizes the LPTIM clock sources in different power modes.

Table 5. LPTIM clock source on different power mode

Mode	HSI	LSI	LSE	MSIK(1)	PCLK	External clock
Run	X	X	X	X	X	X
Sleep	X	X	X	X	X	X
Stop	-	X	X	-	X(2)	X

1. *Only available on STM32U5 devices.*
2. *Only for STM32U5 devices: PCLK can be activated in Autonomous mode.*

5 Synchronization block

The STM32 LPTIM can be configured in two modes:

- Asynchronous mode: the LPTIM is not clocked by an internal clock signal.
  The LPTIM can operate with any external clock (below 15 MHz), while the STM32 device system clock is stopped.
- Synchronous mode: the LPTIM is clocked by an internal clock signal.
  The external signal is synchronized with a clock signal different from the one used by the LPTIM peripheral. The synchronization circuit used to synchronize the external signals is connected to the LPTIM inputs. This circuit is mainly composed of cascaded D flip-flops that are clocked by the LPTIM core clock signal. This synchronization circuit introduces a delay of at least two LPTIM core clock cycles. For this reason, the internal clock signal frequency must be at least two times higher than the external clock signal frequency. In the case that both edges are configured to be the active ones, the internal clock signal frequency must be at least four times higher than the external clock signal frequency. This delay of two counter clocks is also needed after enabling the LPTIM and before the LPTIM starts counting.

6 Use cases

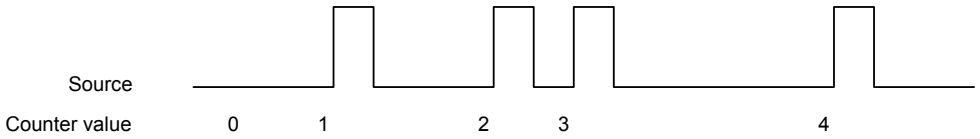
6.1 Asynchronous pulse counter in Stop 2 mode

This application example is provided in the X-CUBE-LPTIMER, with a STM32L476RG device.

The pulse counting method for flow measurement consists of converting the kinetic energy from rotation into electrical digital signals in the form of pulses. The pulse density varies with the measured flow: the faster the rotation, the higher the number of pulses, varying in accordance with the measured flow.

In typical applications, the STM32 device counts the number of pulses, but it does not want to be waked up from Stop mode at every pulse. In such case, the LPTIM configured as pulse counter is very useful: it uses the Asynchronous mode and is clocked by the pulses, which do not have a regular pace. The following figure shows the input signal and the corresponding counter value.

Figure 1. Input signal and corresponding counter value



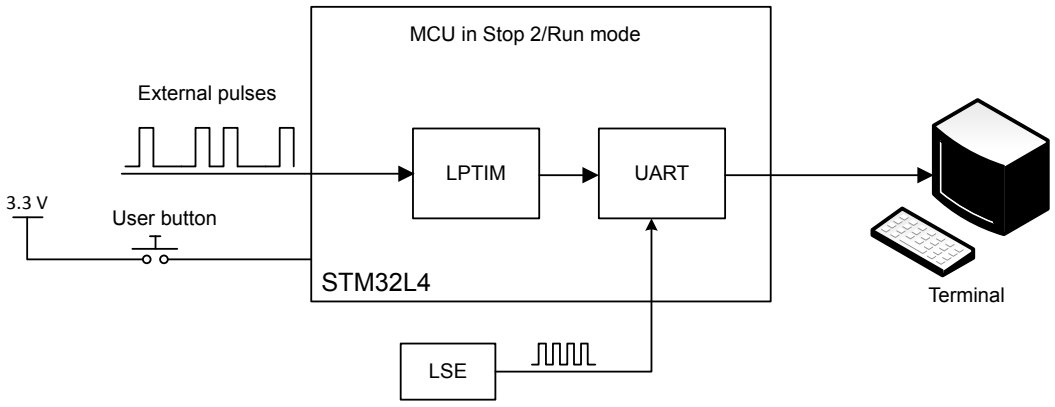
A flow meter can be divided in three parts:

- a transducer that converts the physical quantity to measure into electrical signal
- a data treatment logic that counts and stores
- a display that shows the accumulated count

This application counts the pulses coming from a pulse source and displays them on a terminal by using a serial communication UART. The used parameters are the pulse counter that is based on the LPTIM in Asynchronous mode, and the MCU that is in Stop 2 mode. These parameters have been chosen to show the LPTIM features in low-power mode. The UART is clocked by the LSE to guarantee a low power consumption.

The transmission of the pulse counted by the UART is performed in the interrupt service routine (ISR) of the push-button interrupt. The figure below describes the synoptic of the pulse counter.

Figure 2. Pulse-counter architecture example



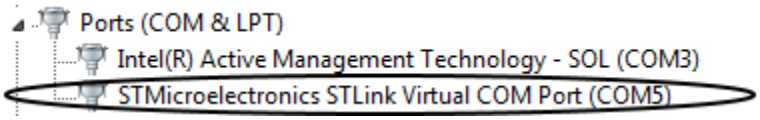
The hardware environment used for this example development is the NUCLEO-L476RG.

To establish the serial communication, the STM32 Nucleo-64 board already embeds the ST-LINK debug and programming probe that features an auxiliary UART interface. This UART is used in this example to interface with the PC and to offer a Virtual COM port interface. This application does not require any additional hardware. It only requires a Mini-B USB cable for data transmission.

A COM port monitor software application on the PC can be used:

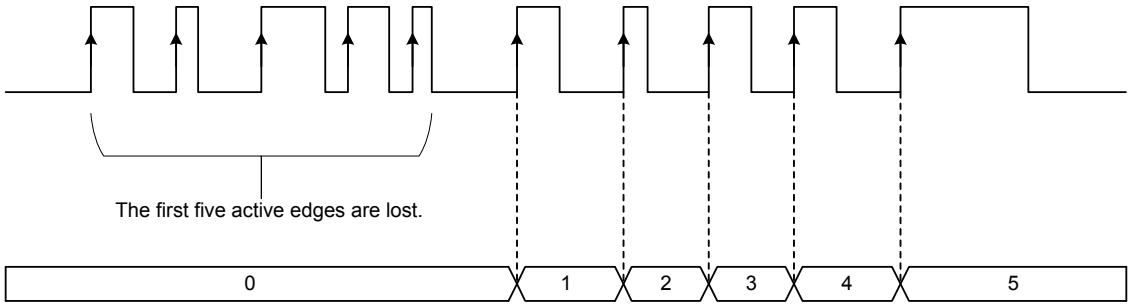
- to display the messages coming from the device via the serial link
- to be able to send data via the serial link to the device

Figure 3. STMicroelectronics Virtual COM port device manager



In Asynchronous mode, the external pulses injected on the LPTIM external Input1 are also used to clock the LPTIM that looses the first five active edges. This effect is illustrated in the figure below. The LPTIM counter can be updated by following either a rising or a falling edge, and not both at the same time (because this mode is not supported in Asynchronous mode).

Figure 4. Asynchronous pulse-counter increment

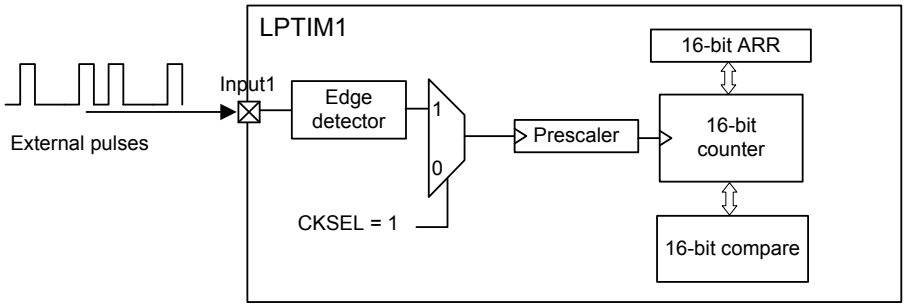


To apply this configuration:

- Set the control bit CKSEL in the LPTIM_CFGR register.
- Use CKPOL bits to configure the active edges.

The following figure shows the block diagram for the asynchronous pulse counter.

Figure 5. Pulse-counter block diagram



Firmware description

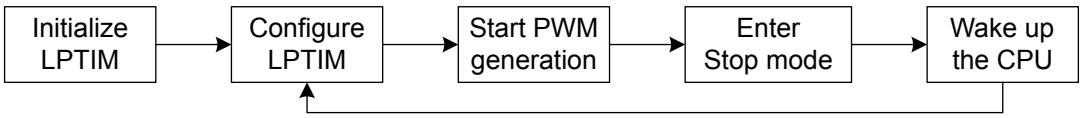
- System clock:
  - System clock: HSI
  - LPTIM1 source clock: external signal
  - UART2 source clock: LSE
- LPTIM1:
  - Source clock: external source (LPTIM1_input1)
  - Prescaler = 1
  - Clock polarity: rising edge
  - Trigger source: software
- UART2:
  - Source clock: LSE
  - Baudrate = 1200
  - Stop bits = 1
  - Parity: none
  - Mode: Tx
  - Hardware flow control: none

6.2 PWM generation in Stop 2 mode

This PMW generation is done in a different way depending on the software used:

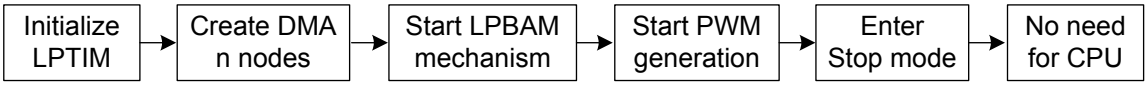
- Legacy mechanism (example available in X-CUBE-LPTIMER)

Figure 6. PWM generation in Stop mode with legacy mechanism



- LPBAM mechanism (example available in STM32CubeU5 firmware)

Figure 7. PWM generation in Stop mode with LPBAM mechanism



One of the main advantages of the LPTIM is its ability to work in Stop mode, with the possibility to generate several different waveforms.

Thanks to the PWM in Stop 2 mode, a signal can be generated:

- with a frequency determined by the value of LPTIM_ARR
- with a duty cycle determined by the value of LPTIM_CMP (or LPTIM_CCRx)

There are two ways to update the values of LPTIM_ARR and LPTIM_CMP (or LPTIM_CCRx) that are controlled by the PRELOAD bit in LPTIM_CFGR:

- PRELOAD = 0: LPTIM_ARR and LPTIM_CMP (or LPTIM_CCRx) are updated immediately after the write access.
  There is a short delay between the write access to the concerned register and the actual update of its value: the waveform at the LPTIM output is directly impacted by the selected PRELOAD mode.
- PRELOAD = 1: LPTIM_ARR and LPTIM_CMP (or LPTIM_CCRx) are updated at the end of the running period of the timer.
  The write access to the concerned register must be done before the last clock cycle of the period, otherwise the update is postponed to the next period.

The LPTIM output depends on the continuous comparison between the LPTIM counter and the LPTIM_CMP (or LPTIM_CCRx) values. Writing directly to the channel register in the middle of a PWM period may generate spurious waveforms. The synchronization of the LPTIM clock source with the APB clock domain creates some latency after the write access in LPTIM_CMP (or LPTIM_CCRx).

Any write access during this latency period must be avoided because it leads to unpredictable results. This latency is the delay between a write in one of the registers (LPTIM_ARR or LPTIM_CMP/CCRx), and the setting of the corresponding flag (ARROK or CMPOK).

The resynchronization time of the LPTIM core clock with the ABP clock is part of the delay. The delay includes 2 × APB_CLK + 3 × LPTIM_CLK, with LPTIM_CLK being the kernel clock of the LPTIM.

The user must then add 2 × APB_CLK to set the ARROK flag to 1, knowing that this write access must be made after the LPTIM enabling.

The change in LPTIM_ARR or LPTIM_CMP (or LPTIM_CCRx) is applied only when their flag is set to inform the application that the APB bus operation has been successfully completed. When ARROK or CMPOK is set, an interrupt is generated to the STM32 device without any need for polling to check the completion of the write operation.

Example:

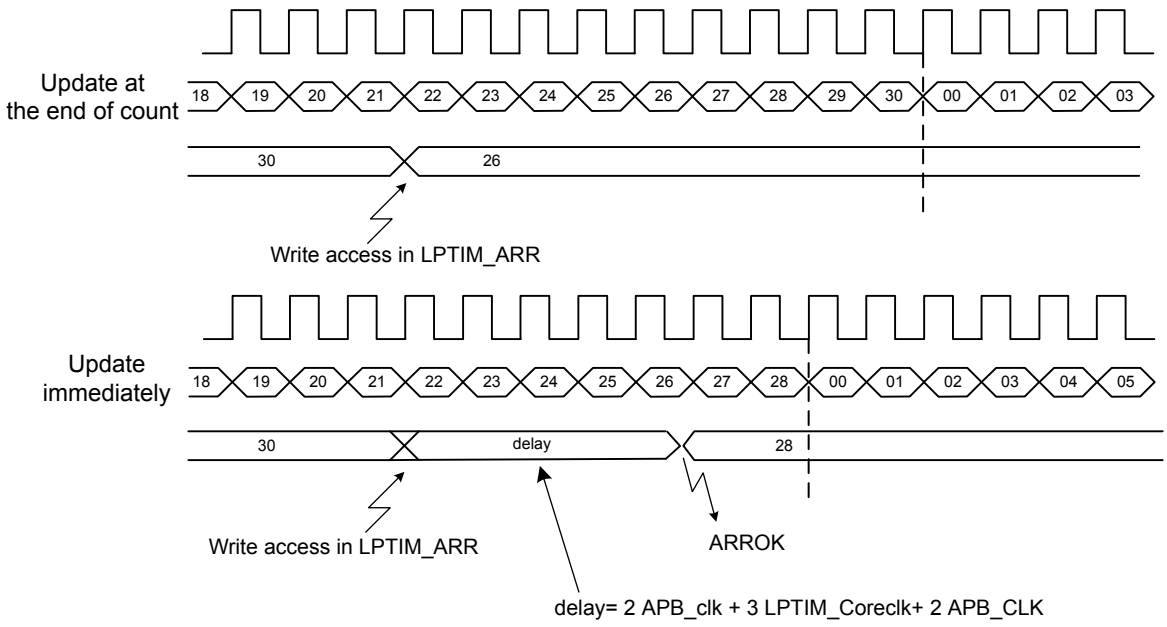
- system clock = 180 MHz and LPTIM_CLK = LSI = 32 kHz
- system cycle = 5.55 ns and LPTIM_CLK cycle = 31.25 µs

The delay is then given by 2 × APB_CLK + 3 × LPTIM_CLK + 2 × APB_CLK =
2 × 5.55 ns + 3 × 31.25 µs + 2 × 5.55 ns = 93.77 µs = 16 896 system cycles

If these ARROK and CMPOK flags do not exist, the application wastes 16 896 cycles to test the update of the register status.

The figure below summarizes the preload mechanism for the LPTIM_ARR register.

Figure 8. Counter timing diagram with and without update immediately



6.2.1 Tasks with legacy mechanism

In this example, the LPTIM1 is selected because it is available in Stop 2 mode. To run the LPTIM1, the chosen source clock is the LSI. The PWM output waveform is configured through:

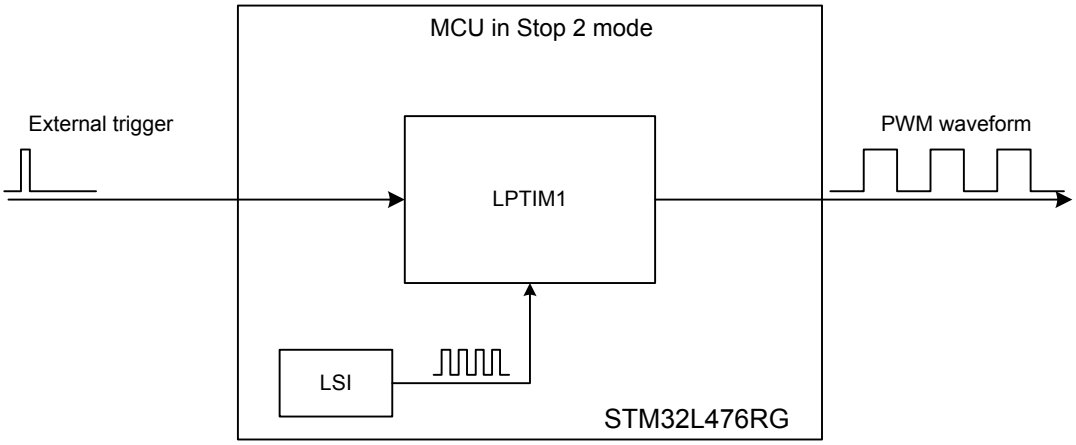
- the WAVE bit reset to 0
- the CNTSTRT bit set to 1
- the WAVEPOL bit that controls the output polarity

The LPTIM1 is started after the detection of an active edge on the LPTIM_ETR pin.

Before the device enters the low-power mode, the LPTIM1 must be configured by loading the period and pulse values to the LPTIM_ARR and LPTIM_CMP (or LPTIM_CCR1) registers, and then by enabling the LPTIM1. To start the LPTIM1, a pulse must be applied on the LPTIM_ETR even if the device is in Stop mode.

The architecture of this example is described in the following figure.

Figure 9. Architecture example of PWM generator in Stop 2 mode with legacy mechanism



Firmware description

- System clock:
  - system clock: MSI
  - LPTIM1 source clock: LSI
- LPTIM1:
  - source clock: internal source (LSI)
  - prescaler = 1
  - trigger source: LPTIM_ETR pin
  - trigger polarity: rising edge
  - output polarity: high

6.2.2 Tasks with LPBAM mechanism

The LPTIM1 is started after a software trigger. This example is based on a STM32U585 device.

The following actions are needed before the device enters the low-power mode:

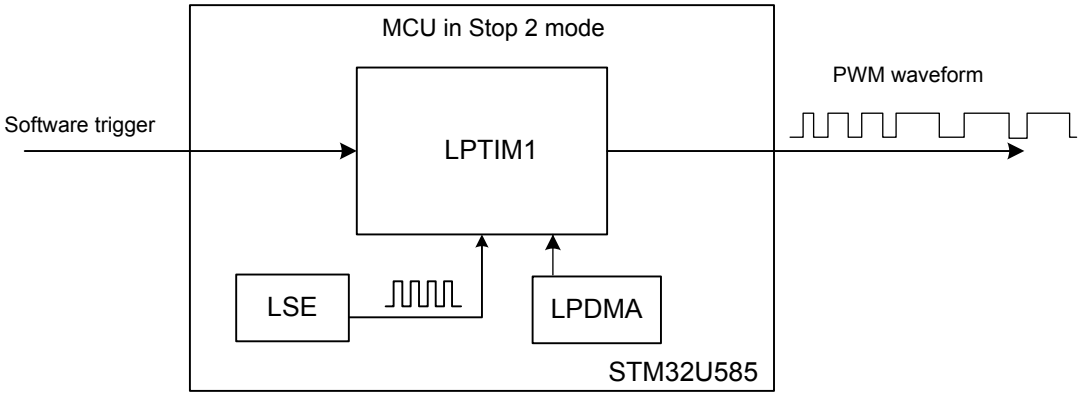
- Configure the LPTIM1 by loading the period and pulse values to the LPTIM_ARR and LPTIM_CRR1 registers.
- Set the DMA information (such as instance and queue type).
- Configure the LPTIM1 to start in Continuous mode.
- Build the LPTIM1 start full DMA linked-list queue:
  - Configure the LPTIM1 enable node:
    - Set the LPTIM1 instance and channel.
    - Set the node descriptor.
    - Set the LPTIM1 configuration.
    - Fill the node configuration.
    - Build the stop node.
    - Insert the node to the LPTIM1 queue.
    - Update register and node indexes.
  - Configure the LPTIM1 wakeup interrupt node.
  - Configure the LPTIM1 configuration node.
- Set PWM parameter to be changed.
- Set PWM configuration X.

The application calls `ADV_LPBAM_LPTIM_PWM_SetFullQ()` to update period, pulse and repetition values. A linked-list queue is created and placed in the SRAM that is executed by a DMA channel in Stop 2 mode. This queue contains configuration nodes to set up the LPTIM1 each time an update event is generated to ensure data transfer in DMA mode:

- Build the LPTIM1 PWM full DMA linked-list queue:
  - Set the LPTIM1 instance and channel.
  - Set the node descriptor.
  - Set the LPTIM1 configuration.
  - Configure the LPTIM1 period node.
  - Configure the LPTIM1 pulse node.
  - Configure the LPTIM1 repetition node.
  - Configure the LPTIM1 clear flag node.
- Set the Circular mode.
- Configure the DMA linked list.
- Start the LPTIM1 PWM generation.
- Enter the whole system to Stop 2 mode.

The architecture of this example is described in the figure below.

Figure 10. Architecture example of PWM generation in Stop 2 mode with LPBAM mechanism



Firmware description

- System clock:
  - system clock: MSI
  - LPTIM1 source clock: LSE
- LPTIM1:
  - instance: LPTIM1_Channel1
  - source clock: LSE
  - prescaler = 1
  - trigger source: software trigger
- LPDMA
  - instance: LPDMA1_Channel0
  - source clock: MSI
  - linked-list mode: Circular mode
  - transfer-event mode: TC event generated at the end of the last linked-list item
  - with the LPBAM task: sequence of DMA transactions conditioned by the LPTIM1 request/trigger

6.3 Timeout wakeup mode

The LPTIM can be configured to periodically wake up the device from Sleep or Stop mode, for example to refresh a display or read a sensor. The main purpose of this mode is to wake up the device only in case of abnormal operation or malfunction (like a watchdog). As long as the periodic trigger keeps coming, the device stays in ultra-low-power mode. If no pulse is generated when it is expected, the device wakes up to identify the reason.

The clock source can be either the LSE, the LSI oscillator or an external clock source. This external clock source can be generated externally and injected to the LPTIM Input1 whenever the LPTIM is already configured to use it (CKSEL appropriately configured).

This LPTIM feature is controlled through the TIMOUT bit in the LPTIM_CFGR register that allows the trigger signal to reset the LPTIM counter by the detection of an active edge when the timer is already started.

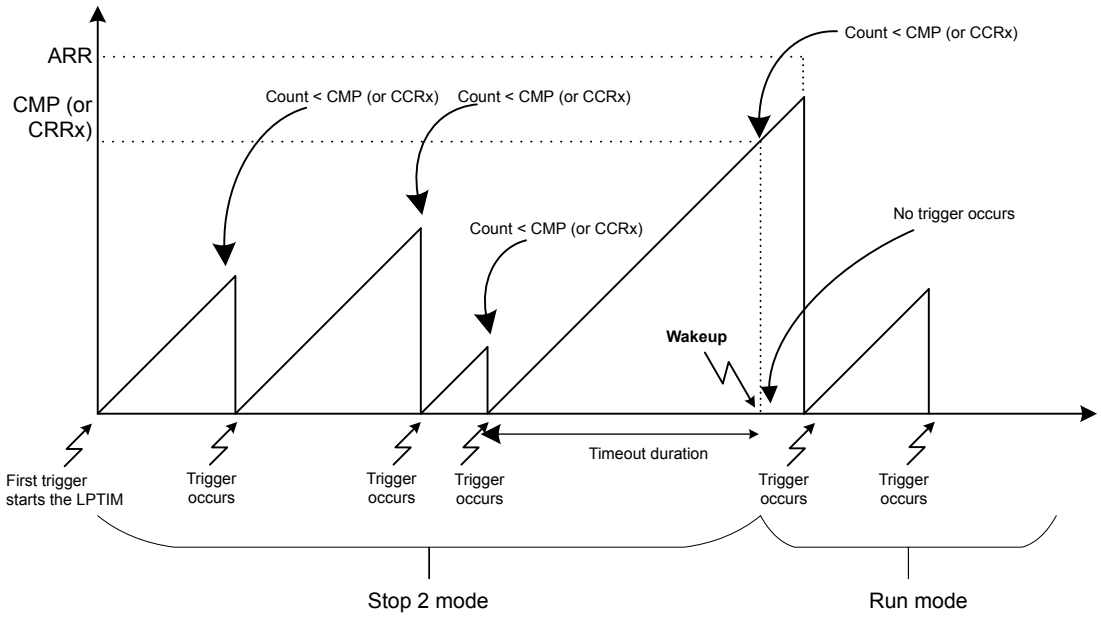
The timeout duration depends on the LPTIM frequency count and LPTIM_CMP (or LPTIM_CCRx) value, as follows:

Timeout = (CMP + 1) / LPTIM clock

Note: CMP is replaced by CCRx for LPTIM type 3.

If the duration between two active edge triggers is smaller than the timeout, the LPTIM resets and restarts the counter while the device remains in Sleep or Stop mode, else the device is awoken by the compare match event. The timeout timing is presented in the diagram below.

Figure 11. Timing of timeout wakeup from Stop 2 mode



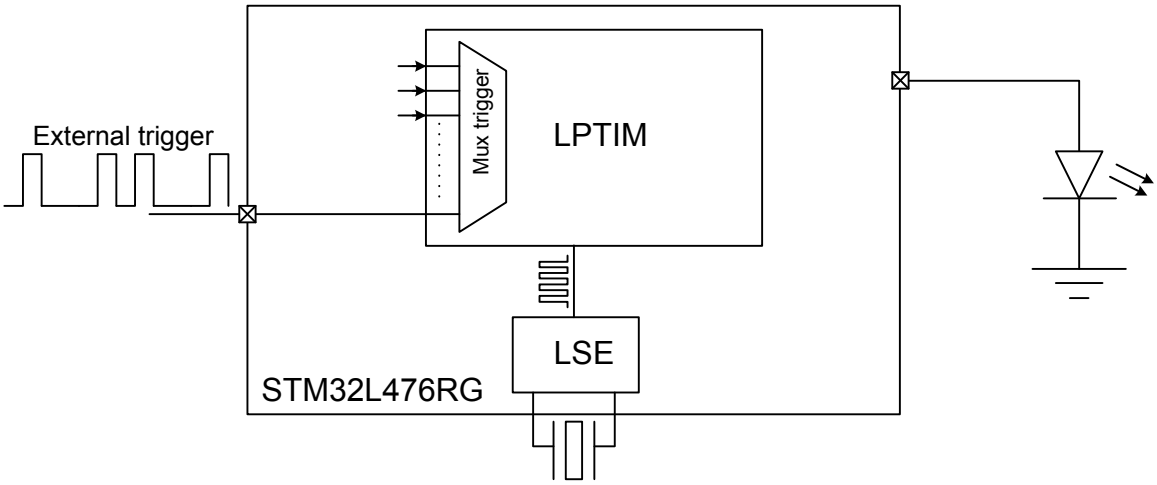
The trigger signal can be selected from several sources by the TRIGSEL bits, such as GPIO, RTC alarm, RTC_TAMP and COMP_OUT. The TRIGSEL bits are used only when TRIGEN[1:0] is different from 00.

In this example, the LPTIM1 is selected because it is available in the Stop 2 mode. The LSE has been chosen as source clock to run this LPTIM. The Timeout mode is configured through the TIMOUT bit in the LPTIM1_CFGR register. The LPTIM1 is started after detection of the first active edge on the LPTIM_ETR.

Before entering the low-power mode, the period and the pulse must be loaded to the LPTIM1_ARR and LPTIM_CMP (or LPTIM_CCRx) registers to set the timeout duration, then the LPTIM1 must be enabled.

The architecture of this example is described in the figure below.

Figure 12. Architecture example of timeout wakeup mode



Firmware description

- System clock:
  - System clock: HSI
  - LPTIM1 source clock: LSE
- LPTIM1:
  - Source clock: internal source (LSE)
  - Prescaler = 1
  - Trigger source: LPTIM_ETR pin
  - Trigger polarity: rising edge
  - Counter source: internal source (LSE)

6.4 Autonomous input capture

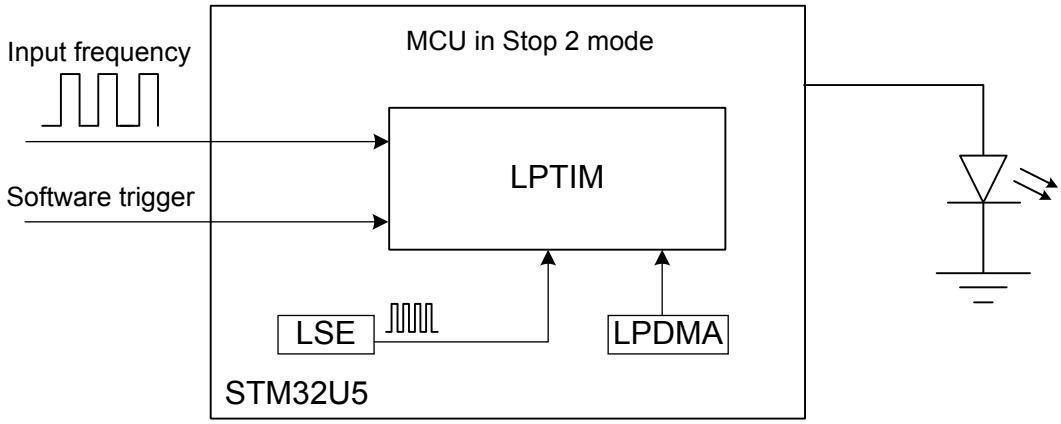
In this example provided in the STM32CubeU5, the LPTIM instance used is LPTIM1 and the low-power mode is Stop mode. The LPTIM1 is configured in Input-capture mode: the external signal is connected to LPTIM1 Channel1 used as input pin.

LPTIM1 configuration

- Enable Autonomous mode: Enable or disable the peripheral bus clock when the SRD domain is in DRUN. After reset,the peripheral clock is disabled when CPUs are in CSTOP.
- Initialize the LPTIM according to the passed parameters.
- Suspend SysTick.
  To minimize the power consumption, after starting Input-capture mode, the device enters Stop 2 mode and DMA transfers data from the LPTIM_CCR1 register to `SRD_DmaCapturedValue` buffer . When the transfer is completed, the DMA generates an interruption to wake up the device.
- Enter in Stop mode.
- Check that the system is resumed from Stop 2 mode, clear the STOP flag, and check this last operation has been correctly done.
- Compute the expected SysTick value.
- Suspend SysTick.
- Get the input-capture value.

The architecture of this example is described in the figure below.

Figure 13. Architecture example of input capture in Stop 2 mode (with Autonomous mode)



Firmware description

- System clock:
  - system clock: MSI
  - LPTIM1 source clock: LSE
- LPTIM1:
  - instance: LPTIM1_Channel1
  - source clock: LSE
  - prescaler = 1
  - trigger source: software trigger
- LPDMA
  - instance: LPDMA1_Channel0
  - source clock: MSI
  - transfer direction: peripheral to memory
  - transfer-event mode: TC event generated at the end of each block

Revision history

Table 6. **Document revision history**

Date	Version	Changes
18-Aug-2016	1	Initial release.
07-Nov-2016	2	Updated cover page.
13-Dec-2016	3	Extended scope to the document to add STM32F413/423 line devices.
19-Feb-2019	4	Updated: • Extended scope to the document to add STM32L4+ Series, STM32H7 Series, STM32G0 Series and STM32WB Series devices. • All tables in the document. • Cover page. Added: • Section 1 General information
17-Dec-2019	5	Extended the document scope to STM32L5 Series devices. Added mention of the repetition counter in Overview.
11-Aug-2020	6	Updated LPTIM availability for STM32H7 Series microcontrollers in Table 3.
27-Sep-2021	7	Added: Section 6.4: Autonomous input capture. Updated: • Introduction with new STM32G4 Series, STM32MP1 Series, STM32U5 Series and STM32WL Series • Section 2: Overview • Section 5: Synchronization block • Section 4: LPTIM clock sources • Section 6.2: PWM generation in Stop 2 mode
08-Dec-2025	8	Updated document title. Added STM32H5 series to Table 1. Applicable products and Table 3. LPTIM types on STM32 devices.

Contents

1 General information 2
2 Overview 3
3 LPTIM versus general-purpose timer 5
4 LPTIM clock sources 6
5 Synchronization block 7
6 Use cases 8
6.1 Asynchronous pulse counter in Stop 2 mode 8
6.2 PWM generation in Stop 2 mode 10
6.2.1 Tasks with legacy mechanism 12
6.2.2 Tasks with LPBAM mechanism 13
6.3 Timeout wakeup mode 15
6.4 Autonomous input capture 16
Revision history 18
List of tables 20
List of figures 21

List of tables

Table 1. Applicable products . . . 1
Table 2. Features on various LPTIM types . . . 3
Table 3. LPTIM types on STM32 devices . . . 4
Table 4. TIM versus LPTIM in working modes . . . 5
Table 5. LPTIM clock source on different power mode . . . 6
Table 6. Document revision history . . . 18

List of figures

Figure 1. Input signal and corresponding counter value . . . 8
Figure 2. Pulse-counter architecture example. . . . 8
Figure 3. STMicroelectronics Virtual COM port device manager . . . 9
Figure 4. Asynchronous pulse-counter increment . . . 9
Figure 5. Pulse-counter block diagram . . . 9
Figure 6. PWM generation in Stop mode with legacy mechanism . . . 10
Figure 7. PWM generation in Stop mode with LPBAM mechanism . . . 10
Figure 8. Counter timing diagram with and without update immediately . . . 11
Figure 9. Architecture example of PWM generator in Stop 2 mode with legacy mechanism . . . 12
Figure 10. Architecture example of PWM generation in Stop 2 mode with LPBAM mechanism . . . 14
Figure 11. Timing of timeout wakeup from Stop 2 mode . . . 15
Figure 12. Architecture example of timeout wakeup mode . . . 16
Figure 13. Architecture example of input capture in Stop 2 mode (with Autonomous mode) . . . 17