How to optimize power consumption on STM32U5 MCUs

Introduction

The STM32U5 series microcontrollers (MCUs) are based on the high-performance Arm® 32-bit Cortex®-M33 CPU with Arm® TrustZone® and FPU. These MCUs use an innovative architecture to reach best-in-class, ultralow-power figures owing to their high flexibility and advanced set of peripherals. The STM32U5 series devices provide a very-high energy efficiency for applications.

The STM32U5 series devices with a "Q" suffix (such as STM32U5xxxxQ) support the use of an internal SMPS in Run and low-power modes, enabling very efficient and low-power application design.

With the integration of the ART accelerator 8-Kbyte instruction cache, the STM32U5 series MCUs can operate at frequencies up to 160 MHz and achieve 240 DMIPS performance, while maintaining extremely low dynamic power consumption.

The STM32U5 series devices embed a high number of smart, high-performance peripherals, and a large set of advanced and ultralow-power analog features. Many peripherals (including communication, analog, timers, and audio) can be functional and autonomous down to Stop 2 mode with direct memory access, using LPBAM (low-power background autonomous mode).

The combination of ultralow power design and processing performance allows these devices to achieve an industry leading EEMBC® ULPBench™ score, up to 464 ULPMark™.

The STM32U5 series MCUs embed several innovations, which minimize the consumption in the different modes, while maintaining most of the existing peripherals and an excellent pin-to-pin compatibility to allow an easy migration from existing families.

The built-in internal voltage regulator and voltage scaling keep the consumption in active modesat a minimum, whatever the external supply voltage. These devices are thus particularly suited for portable battery-supplied products, down to 1.71 V.

Additionally, their multivoltage domains allow a low-voltage supply, while the analog-to-digital and digital-to-analog converters can operate with a higher supply and reference voltage, up to 3.6 V.

The STM32U5 series devices support a battery backup domain to keep the RTC (real-time clock) running, and a set of 32 registers, each 32 bits wide, that can be retained in case of power loss. This optional backup battery can be charged when the main supply is present.

These devices support several main low-power modes, each of them with several submode options. This allows the designer to achieve the best compromise between low-power consumption, shorter startup time, available set of peripherals, and maximum number of wake-up sources.

1 General information

This application note applies to the STM32U5 series microcontrollers that are Arm® Cortex® core-based devices.

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

Reference documents

[1] Reference manual *STM32U5 series Arm®-based 32-bit MCUs* (RM0456)

[2] Datasheets for STM32U575xx (DS13737), STM32U585xx (DS13086), STM32U535xx (DS14217), STM32U545xx (DS14216), STM32U59xx (DS13633), and STM32U5Axx (DS13543).

[3] Application note *STM32 microcontroller GPIO configuration for hardware settings and low-power consumption* (AN4899)

[4] Application note *STM32U5 series power optimization using LPBAM* (AN5645)

[5] EEMBC organization on http://www.eembc.org

2 Energy efficiency processing

The high processing performance in Run mode (expressed in DMIPS/MHz) is achieved by using a Cortex-M33 core associated with the interfaces of its memories. To ensure full performance operation at maximum operating frequency, the STM32U5 series devices embed the ART accelerator instruction cache (ICACHE), which masks the flash memory access wait state. The processing performance can then achieve 1.5 DMIPS/MHz, whatever the system clock frequency.

The STM32U5 series devices support dynamic voltage scaling to optimize its power consumption in Run mode. The voltage from the main regulator that supplies the logic (V_{CORE}) can be adjusted according to the maximum operating frequency of the system. Refer to the document [1] for more details.

The main regulator operates in the following ranges:

- Range 1 (V_{CORE} = 1.2 V) with CPU and peripherals running at up to 160 MHz
- Range 2 (V_{CORE} = 1.1 V) with CPU and peripherals running at up to 110 MHz
- Range 3 (V_{CORE} = 1.0 V) with CPU and peripherals running at up to 55 MHz
- Range 4 (V_{CORE} = 0.9 V) with CPU and peripherals running at up to 25 MHz

All consumption data used in this application note are based on typical specifications extracted from the document [2] at T_A = 25°C and V_{DD} = 3.0 V, unless otherwise specified.

Note: *When the low-power regulator is used, the system clock can be either MSIS up to 24 MHz (4 MHz as default), or HSI16, depending on the software configuration.*

2.1 Internal regulator efficiency

The STM32U5 series devices embed two internal regulators, that can be selected when the application runs, depending on the application requirements:

- a SMPS step-down converter
- a linear voltage regulator (LDO)

The LDO and SMPS regulators have two modes:

- main regulator (used when performance is needed)
- low-power regulator

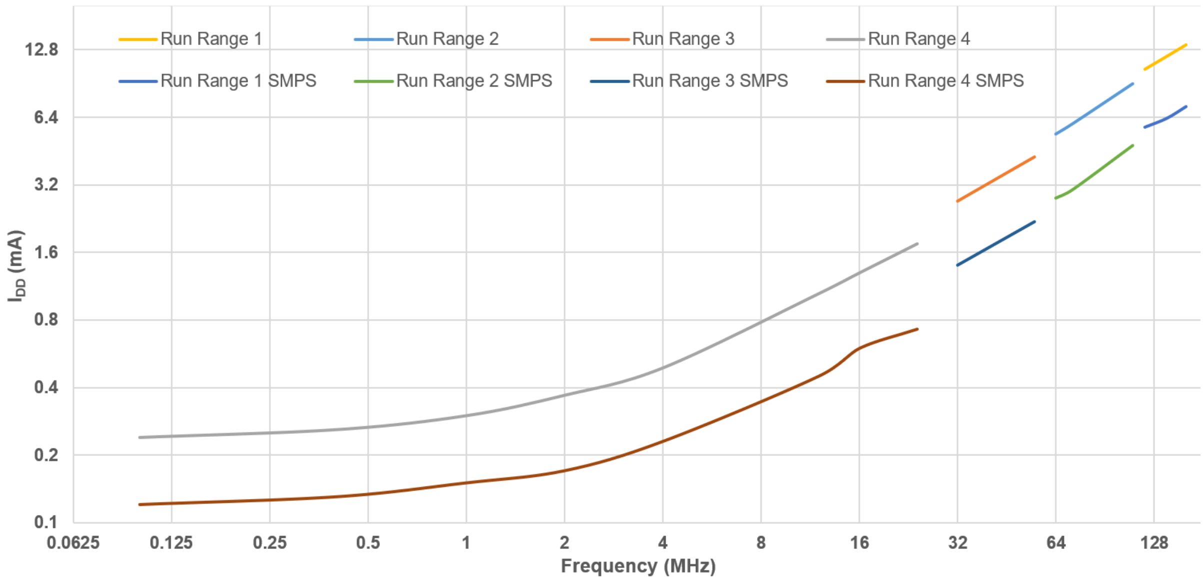
Both regulators can provide four different voltages (voltage scaling) and can operate in Stop mode.

Table 1. Power distribution of the internal regulators

Voltage regulator mode	Device mode
Main regulator (Range 1, 2, 3, 4)	Run, Sleep, Stop 0
Low-power regulator	Stop 1, Stop 2, Stop 3, (Standby with 8-Kbyte or full SRAM2)

The figure below shows the typical current consumption of an STM32U5 series microcontroller in Run mode, as a function of system frequency, for both SMPS and LDO configurations.

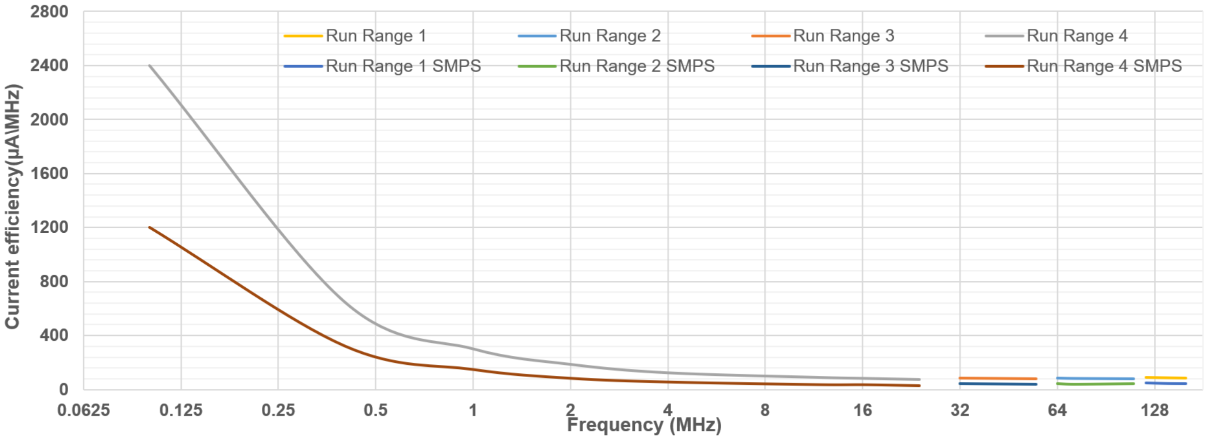
Figure 1. Current consumption for STM32U575/U585 in Run mode with ICACHE ON, 1 -way, prefetch ON, SMPS versus LDO



Note: *The lowest power consumption is achieved when running from the internal flash memory with cache 1-way. The instruction cache tends to reduce the number of accesses to the memory thus reducing the overall current consumption. The current consumption from the internal SRAM is similar to the flash memory when the ICACHE is used.*

The figure below shows the same curve translated into power efficiency (by dividing the current consumption, multiplied by supply voltage to get power figure), by the CPU frequency.

Figure 2. Run mode power efficiency for STM32U575/U585 with ICACHE ON, 1-way, prefetch ON, SMPS versus LDO



2.2 ICACHE efficiency

The purpose of the instruction cache (ICACHE) is to cache instruction fetches or instruction memory loads coming from the processor. ICACHE only manages read transactions and does not manage write transactions. It reduces power consumption by fetching instructions from the internal ICACHE most of the time, rather than from the bigger and more power-consuming main memories.

The default ICACHE configuration (at reset) is a 2-way set associative cache. For applications needing a very-low-power consumption profile, ICACHE can be configured as 1-way, meaning direct-mapped cache.

Table 2. Current consumption for STM32U575/U585 in Run mode with ICACHE ON (1-way) versus prefetch

Fibonacci benchmark with SMPS and V_{DD} = 1.8 V

Frequency (MHz)	Current consumption (mA)	
	Prefetch ON	Prefetch OFF
160	8.25	8.35
110	5	5.06
55	2.4	2.41
24	0.97	0.97

Note: Prefetch tends to increase the code execution performance at the cost extra flash memory accesses.

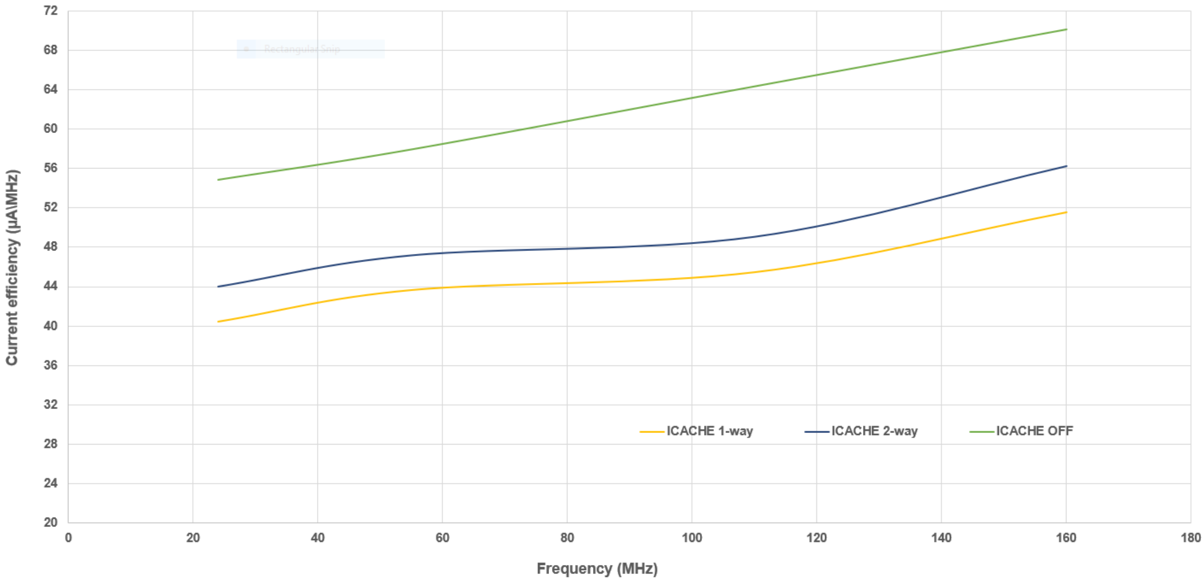
Table 3. Current consumption for STM32U575/U585 in Run mode with prefetch ON versus ICACHE configurations

Fibonacci benchmark with SMPS and V_{DD} = 1.8 V

Frequency (MHz)	Current consumption (mA)		
	ICACHE ON (1 way)	ICACHE ON (2 ways)	ICACHE OFF
160	8.25	8.99	11.21
110	5	5.39	7.075
55	2.4	2.59	3.186
24	0.97	1.05	1.316

The figure below shows the power efficiency of an STM32U575/U585 microcontroller in Run mode on a Fibonacci benchmark for different ICACHE configurations, as a function of system frequency, with SMPS and V_{DD} = 1.8 V.

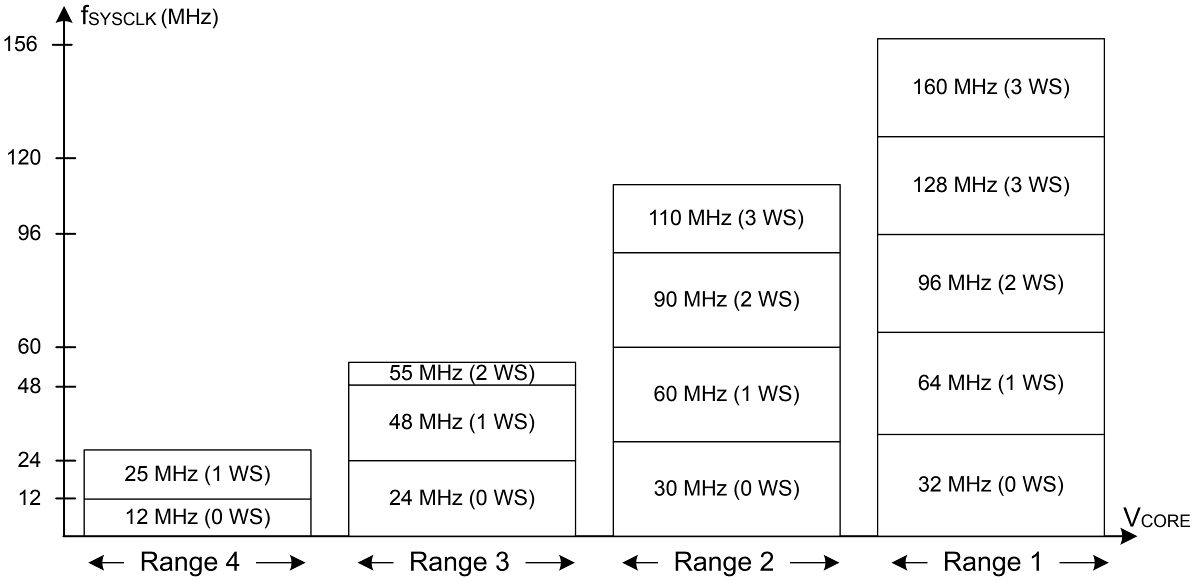
Figure 3. Power efficiency for various ICACHE configurations (SMPS, prefetch ON, V_{DD} = 1.8 V)



2.3 Flash memory efficiency

The figure below shows the flash memory latency (number of wait states to be programmed in the flash memory access control register), depending on the STM32U5 series regulator voltage scaling range and system clock frequency.

Figure 4. Flash memory latency vs V_{CORE} range (V_{DD} = 1.71 to 3.6 V, LPM = 0)



The flash memory supports a low-power read mode (LPM). The number of wait states (WS) depends on LPM:

- LPM = 0: up to 4 WS depending on supply voltage and frequency
- LPM = 1 (reduced consumption and increased latency): up to 15 WS depending on supply voltage and frequency

Table 4. Current consumption for STM32U575/U585 in Run mode versus LPM

Symbol	Conditions	Current consumption (mA)	
		LPM = 1	LPM = 0
$I_{DD(RUN)}$	SMPS, prefetch ON, Fibonacci code, V_{DD} = 3.0 V, Range 4	0.59	0.65

It is particularly interesting in Range 4 to reduce the dynamic consumption by enabling LPM in FLASH_ACR.

3 STM32U5 series ultralow-power feature

3.1 Low-power modes

The STM32U5 series microcontrollers implement many different power modes. By default, the MCU is in Run mode after a system or power reset. Several low-power modes are available to save power when the CPU does not need to be kept running, for example when waiting for an external event. It is up to the user to select the mode that gives the best compromise between low-power consumption, short startup time, and available wake-up sources. Refer to document [1] for more details.

3.1.1 Sleep mode

CPU clock off, all peripherals, including Cortex-M33 core peripherals like NVIC and SysTick, can run and wake up the CPU when an interrupt or event occurs.

3.1.2 Stop mode

Stop mode is based on the Cortex-M33 deep-sleep mode combined with the peripheral clock gating. All clocks in the core domain are stopped. The PLL, MSIS, MSIK, HSI16, and HSE oscillators are disabled with the possibility to enable MSIS, MSIK, HSI16 in Stop 0, Stop 1, or Stop 2 mode if a peripheral requests it.

Stop 0 and Stop 1 modes offer the largest number of active peripherals and wake-up sources, a smaller wake-up time but a higher consumption than Stop 2. In Stop 2 and Stop 3 modes, most of the core domain is put in a lower leakage mode.

When exiting Stop mode, the system clock can be either MSIS up to 24 MHz or HSI16, depending on software configuration.

Low-power background autonomous mode (LPBAM)

Some peripherals are autonomous and can operate in Stop 0, Stop 1, or Stop 2 mode by requesting their kernel clock (MSIK up to 24 MHz, or HSI16) and their bus clock (APB or AHB) when needed. This allows them to operate, and to transfer data with DMA (GPDMA1 or LPDMA1, depending on peripherals and power mode).

The LPBAM ecosystem (STM32CubeMX LPBAM based on the LPBAM utility) uses LPDMA1 linked-list transfers to develop complex use cases without any CPU wake-up. This functionality is independent from the device power modes, down to Stop 2 mode. It drastically reduces the application power consumption when peripherals must be kept active.

3.1.3 Standby mode

The Standby mode is used to achieve the lowest power consumption with a brownout reset (BOR): the internal regulator is switched off, so that the core domain is powered off. The PLL, MSI (MSIS and MSIK), RC, HSI16 RC, and HSE crystal oscillators are also switched off. The RTC can remain active. 8, 56, or 64 Kbytes of SRAM2 can be optionally retained. Internal pull-up or pull-down can be applied to keep I/O level.

The system clock after wake-up is MSIS up to 4 MHz.

3.1.4 Shutdown mode

The lowest power consumption is reached in Shutdown mode. It is based on the deep-sleep mode with the voltage regulator disabled. The core domain is consequently powered off. This mode provides the lowest consumption by switching off the internal voltage regulators, and by disabling the voltage power monitoring. A wake-up from this mode is done using one of the five wake-up pins or the reset pin. The RTC clocked by the low-speed external oscillator (LSE) is also functional in this mode, with wake-up capability.

The 'low-power mode summary' table in the 'Power modes' section of the document [1] summarizes the features available for each mode and provides an indication of their power consumption.

3.2 Power consumption optimization

3.2.1 ICACHE in Run mode

ICACHE reduces power consumption by fetching instructions from the internal ICACHE. Most of the time, applications with a lower-performance profile and stringent low-power consumption constraints may benefit from the lower power consumption of an ICACHE configured as direct mapped (ICACHE_1-way).

This single-way cache configuration is obtained by programming WAYSEL = 0 in ICACHE_CR.

3.2.2 FLASH in Run and Sleep modes

The STM32U5 series flash memory includes a main memory block organized as two banks up to 2 Mbytes each, containing 256 pages of 8 Kbytes. This structure is optimized in terms of power consumption using dedicated modes when the MCU is in Run or Sleep mode.

To reduce power consumption in Run and Sleep modes, each bank can be put in power-down mode independently by setting PDREQ1 or PDREQ2 in FLASH_ACR, depending on which bank is used.

The flash memory supports a low-power read mode to reduce power consumption by programming LPM = 1 in FLASH_ACR.

Note: Prefetch tends to increase the code execution performance at the cost of extra flash memory accesses. However, for most applications, power efficiency is better with prefetch ON.

3.2.3 Power control optimization

In Standby mode, BOR can be configured in discontinuous mode (ultralow-power mode) to further reduce current consumption by setting to 1 the ULPMEN bit in PWR_CR.

A switch from LDO to SMPS is recommended to reduce power consumption in Run, Sleep, or any other low-power mode.

Note:
- *The SMPS power supply pins are available only on a specific package with the SMPS step-down converter option.*
- *The SMPS to LDO and LDO to SMPS switches can be done on the fly.*

3.2.4 RCC in low-power modes

By default, the MSI bias is in continuous mode to maintain the output clocks accuracy. Setting MSIBIAS in RCC_ICSCR1 before entering low-power mode reduces MSI consumption when the regulator is in Range 4, and when the device is in Stop 1 or Stop 2 mode.

Before entering a low-power mode, selecting either HSI16 or MSIS (16 or 24 MHz) as a wake-up source is recommended to speed up the wake-up time, and to reduce the energy lost during wake-up. If 24 MHz is used, one wait-state latency must be configured for SRAM read access, which impacts code execution performance. This performance must then be evaluated between using 24 MHz with 1 SRAM wait state, and using 16 MHz for the system clock.

Note: The MSIS frequency during Stop mode or when waking up from Stop mode is the same as before entering Stop mode. If the MSIS is used to feed the PLL in Run mode, the MSIS is between 4 and 16 MHz due to the PLL input clock frequency constraint. In this case, it is recommended to switch off the PLL and to configure MSIS with high frequency before entering Stop mode (if MSIS is used as wake-up clock).

All peripherals that cannot be enabled in Stop 2 mode must either be disabled by clearing the enable bit in the peripheral itself, or put in reset state by configuring the RCC registers.

Each clock source can be switched on or off independently when it is not used, to optimize power consumption.

3.2.5 I/O states in low-power modes

Refer to document [3] for more details.

Configure unused GPIO

A GPIO always has an input channel, which can be either digital or analog. If it is not necessary to read the GPIO data, prefer the analog input configuration. This saves the input Schmitt trigger consumption.

Disable GPIO register clock

If a GPIO bank does not need to be used for a long period, disable its clock by using the `HAL_RCC_GPIOx_CLK_DISABLE()` function.

Configure GPIO when entering low-power modes

In Sleep, Stop 0, Stop 1, or Stop 2 mode, all I/O pins keep the same state as in Run mode. For outputs, set the level to that required by the external component.

In Stop 3, Standby, or Shutdown mode, the I/Os are in floating state by default. Apply a pull-up or pull-down, depending on the level required by the external component. Be aware that this pull-up/pull-down is not applied when exiting Shutdown mode, until the firmware configures the GPIO.

Note: In some applications, an external pull-up is recommended instead of an internal pull-up to further reduce power consumption in Stop modes, and to obtain a higher accuracy in the resistance value.

3.2.6 Internal SRAMs

The STM32U5 series devices embed five SRAMs, each with specific features: SRAM1, SRAM2, SRAM3[1], SRAM5[1], and SRAM6[1] are the main SRAMs. SRAM4 is in the SRAM used for peripherals low-power background autonomous mode (LPBAM) in Stop 2 mode.

All SRAMs and register contents are preserved, but the SRAMs can be switched off entirely or partially to further reduce consumption. These SRAMs are made of several blocks that can be powered down:

- Run mode
  To reduce power consumption in Run mode, power off SRAM1, SRAM2, SRAM3[1], SRAM4, SRAM5[1], and SRAM6[1] by programming SRAMxPD in PWR_CR1.
- Stop mode
  This mode achieves the lowest power consumption while retaining the content of SRAM and registers supplied by the low-power regulator. Stop 3 is the lowest power mode with full retention. To further optimize power consumption, the following is recommended:
  - Power down unused SRAMs by setting the corresponding bit to 1 in PWR_CR2:
    - SRAM1, SRAM3, SRAM4
    - SRAMs of DCACHE1, ICACHE, DCACHE2[1], DMA2D[1], PRAM (FMAC[1], FDCAN[1], USB/OTG_FS/OTG_HS[1]), PKA[1], GPRAM (LTDC[1][2], GFXMMU[1]), DSI[1], and JPEG[1]
  - Depending on the application data size, retain SRAM2 page1 and/or page2 by clearing SRAMPDS1 and/or SRAMPDS2 bits in PWR_CR2.
- Standby mode
  After entering Standby mode, SRAM and register contents are lost, except for registers and backup SRAM (2 Kbytes) in the backup domain and Standby circuitry. SRAM2 page1 (8 Kbytes), page2 (56 Kbytes), or both can be retained in Standby mode by setting RRSB1 and/or RRSB2 in PWR_CR1, supplied by the low-power regulator (Standby with RAM2 retention mode).

[1] This feature is only available on some devices in the STM32U5 series. Refer to the device datasheet for availability of its associated peripheral.

[2] LTDC SRAM content is always lost in Stop 2 and Stop 3 modes. It can be retained only in Stop 0 and Stop 1 modes.

3.2.7 Peripherals clock gating in Run and Stop modes

The STM32U5 series devices support the capability to gate off the AHB/APB clock to further reduce power consumption.

When none of the AHB/APB peripherals are used, and when their clocks are disabled, the following happens:

- All AHB1 peripheral clocks are off (except for BKPSRAM, DCACHE1, FLASH, ICACHE, and SRAM1) when setting AHB1DIS = 1 in RCC_CFGR2.

Note: The SRAM1 can be disabled independently by programming SRAM1EN=0 in RCC_AHB1ENR.

- All AHB2 peripheral clocks from RCC_AHB2ENR1 are off (except for SRAM2 and SRAM3(1)), when setting AHB2DIS = 1 in RCC_CFGR2.

Note: SRAM2 and SRAM3(1) can be disabled independently by programming SRAM2EN = SRAM3EN(2) = 0: in RCC_ AHB2ENR1.
SRAM5(1) and SRAM6(1) can be disabled by programming SRAM5EN(2) = SRAM6EN(2) = 0: in RCC_ AHB2ENR2.

- All AHB3 peripherals clocks are off (except for SRAM4), when setting AHB3DIS = 1 in RCC_CFGR3.

Note: SRAM4 can be disabled independently by programming SRAM4EN = 0 in RCC_AHB3ENR.

- All APB1/2/3 peripheral clocks are off when setting APB1DIS = APB2DIS = 1 in RCC_CFGR2 and APB3DIS = 1 in RCC_CFGR3.

1. This feature is only available on some devices in the STM32U5 series. Refer to the device datasheet for availability of its associated peripheral.

2. This bit is only available on some devices in the STM32U5 series. Refer to the device datasheet for the availability of its associated peripheral. If not present, consider this bit as reserved and keep it at reset value.

When a peripheral is enabled, its clock can be automatically gated off when the device is in Sleep mode. This is done by clearing the peripheral SMEN bit in RCC_AHBxSMENR and RCC_APBxSMENR. Both EN and SMEN bits of the peripheral must be set to 1 to keep the clock on in Sleep mode.

The SMEN bit of the peripheral is also used to allow peripheral clocking in Stop 0 and Stop 1 modes, upon request by the peripheral. When the clock is requested by a peripheral, it is distributed to all enabled peripherals. Therefore, if the peripheral is not used in Stop mode, the SMEN bit must be cleared before entering this mode.

Caution: The SMEN bit of the peripheral must be set to allow the generation of an interrupt capable to wake up the device from Stop mode.

3.3 Power optimization of peripherals using LPBAM

The STM32U5 series devices support a low-power background autonomous mode (LPBAM). This mode allows peripherals to be functional and autonomous in Stop 0, Stop 1, and Stop 2 modes (without any software running). LPBAM only supports the peripherals in the Smartrun domain (SRD, with AHB3 and APB3 peripherals).

The peripherals DMA transfers are also supported by the autonomous peripherals from the CPU domain (CD, with AHB1, AHB2, APB1, and APB2 peripherals), down to Stop 1 mode. The LPBAM ecosystem does not support those peripherals, because there is no significant power saving in Stop 0 or Stop 1 mode if the CPU is woken up to reconfigure the peripherals.

The autonomous peripherals functionality in Stop mode is made possible by the peripheral own clock request capability. The clock automatically switches on when a peripheral requests its use, and off when no peripheral does. Refer to document [1] for more details.

3.3.1 Peripherals supporting autonomous mode

The table below lists all STM32U5 series peripherals that support the autonomous mode.

Table 5. **Peripherals supporting autonomous mode**

Low-power mode	Peripherals
Stop 0 and Stop 1	ADC4, ADF1, DAC1, GPDMA1, LPDMA1, LPTIMx (x = 1 to 3), LPUART1, MDF1, I2Cx (x = 1 to 4), SPIx (x = 1 to 3), USARTx (x = 1 to 5)
Stop 2	ADC4, ADF1, DAC1, LPDMA1, LPTIM1, LPTIM3, LPUART1, I2C3, SPI3

3.3.2 Main use cases

Examples of use cases that can be done while remaining in Stop mode (see document [4] for more details):

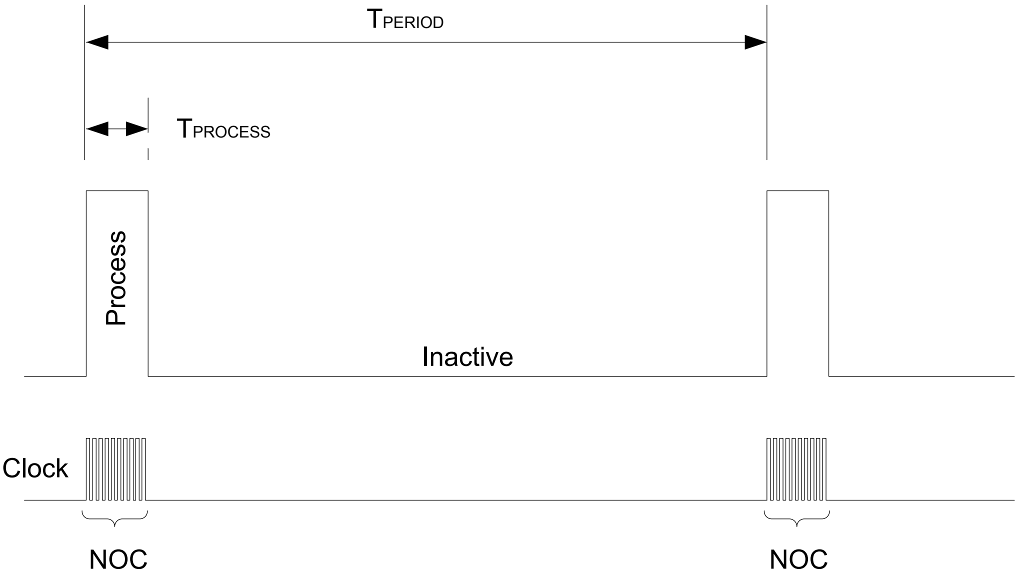
- ADC or DAC conversion triggered by a low-power timer (or any other trigger)
- Audio digital filter data transfer into SRAM: wake-up from Stop on sound activity detection
- I^2C/SPI slave transfer, UART/LPUART reception: wake-up at the end of peripheral transfer or on DMA buffer event
- I^2C/SPI master transfer, UART/LPUART transmission, triggered by LPTIM trigger (sensor periodic read for example): wake-up at the end of peripheral transfer or on DMA buffer event
- Bridges between peripherals. Example: ADC converted data transferred by communication peripherals
- Data transfer from a SRAM to another one

4 Low-power mode selection methodology

A simplified application model has been considered, where the application wakes up every T_{PERIOD} and performs some processing that is always the same and seen as a constant number of instructions (no waiting loops, no data dependencies). The following approximations can then be made:

- The process phase duration ($T_{PROCESS}$) can be defined in terms of the number of cycles (NOC) to be executed during each period. Defining F_{CLK} as the CPU system clock frequency, the duration of the process phase is equal to $T_{PROCESS}$ = NOC / F_{CLK}. The average current consumption during this phase is equal to $I_{PROCESS}$.
- The inactive phase duration is T_{PERIOD} - $T_{PROCESS}$, and its average current consumption is $I_{INACTIVE}$.

Figure 5. Application sequence and parameters



In order to get the average current consumption, both consumptions during the process phase and during the inactive phase have to be summed up as follows:

$$I_{AVERAGE} = I_{PROCESS} \times \left(\frac{T_{PROCESS}}{T_{PERIOD}}\right) + I_{INACTIVE} \times \left(\frac{T_{PERIOD} - T_{PROCESS}}{T_{PERIOD}}\right)$$

The process phase duration is proportional to the number of cycles to execute, which is a constant in this use case:

$$I_{AVERAGE} = I_{INACTIVE} \times + \left(\frac{(I_{PROCESS} - I_{INACTIVE}) \times NOC}{F_{CLK} \times T_{PERIOD}}\right)$$

This model does not consider the consumption during the transition from inactive phase to process phase, which is addressed in Section 4.5.

Two parameters need to be selected to optimize the average power consumption:

- Run mode and clock frequency to be used during the process phase
- low-power mode to be used during the inactive phase

4.1 Process phase

Depending on the processing requirement (DMIPS), one of these Run modes can be used:

- Run mode with Range 1 voltage, with a maximum of 240 DMIPS when run at 160 MHz
- Run mode with Range 2 voltage, with a maximum of 165 DMIPS when run at 110 MHz
- Run mode with Range 3 voltage, with a maximum of 82.5 DMIPS when run at 55 MHz
- Run mode with Range 4 voltage, with a maximum of 37.5 DMIPS when run at 25 MHz

Another parameter that needs to be considered when selecting the frequency is the ability to support the constraints related to the peripherals (if any).

4.2 Inactive phase

The STM32U5 series devices provide different low-power modes that can be used for the inactive phase:

- Sleep
- Stop 0 (with main regulator on)
- Stop 1 (with low-power regulator)
- Stop 2 (with low-power regulator)
- Stop 3 (with low-power regulator)
- Standby (with RTC and SRAM2 retention as options)
- Shutdown (with RTC as option)

Depending on the wake-up source and the duration of the Sleep period, one of the following modes can be selected:

- Sleep: if reactivity is the key parameter (in this case, the wake-up time is only eight system clock cycles)
- Stop 0: when the wake-up timing is critical (< 2.5 µs if the program is in the flash memory)
- Stop 1: if a lot of peripherals have to stay awake and the system has multiple wake-up sources (in this case, the application must tolerate a wake-up time from flash memory of approximately 13 µs)
- Stop 2: if few peripherals have to stay awake and can generate a wake-up event as DMA transfer complete, LPUART/SPI/I2C transfer event, voice detection through ADF, ADC analog watchdog (in this case, the application must tolerate a wake-up time from flash memory of about 20 µs)
- Stop 3: if no other peripherals than RTC need to stay awake with all SRAM retention, the wake-up time from flash memory is 66.5 µs with FSTEN = 0
- Standby: if no other peripherals than RTC need to stay awake, and no more than 64-Kbyte SRAM2 retention is required (in this case, the application must tolerate a wake-up time from flash memory of about 64.5 µs with MSIS = 4 MHz and FSTEN = 0)
- Shutdown: if only the RTC and backup registers need to stay awake (in this case, the application must tolerate a wake-up time of about 610 µs with MSIS = 4 MHz)

Note: The wake-up timing depends on the code location (flash memory), the system clock sources, and the frequencies. Refer to document [2] for detailed conditions. The wake-up from Stop figures above correspond to MSIS at 24 MHz or 4 MHz with the code in flash memory. To reduce wake-up time, it is recommended to set FLASHFWU = 1 and SRAM4FWU = 1 in PWR_CR2.

4.3 Sleep mode selection

The following numerical results are based on typical specifications extracted from the document [2] at 25°C (typical value).

The figures below give the theoretical power consumption for different selections of low-power mode (Sleep, Stop 1, Stop 2, Stop 3, Standby with RTC and SRAM2 retention, Standby and Shutdown), with RTC in the inactive phase, as a function of main supply (V_{DD}).

Figure 6. STM32U575/U585 low-power mode influence on average energy consumption (LDO)

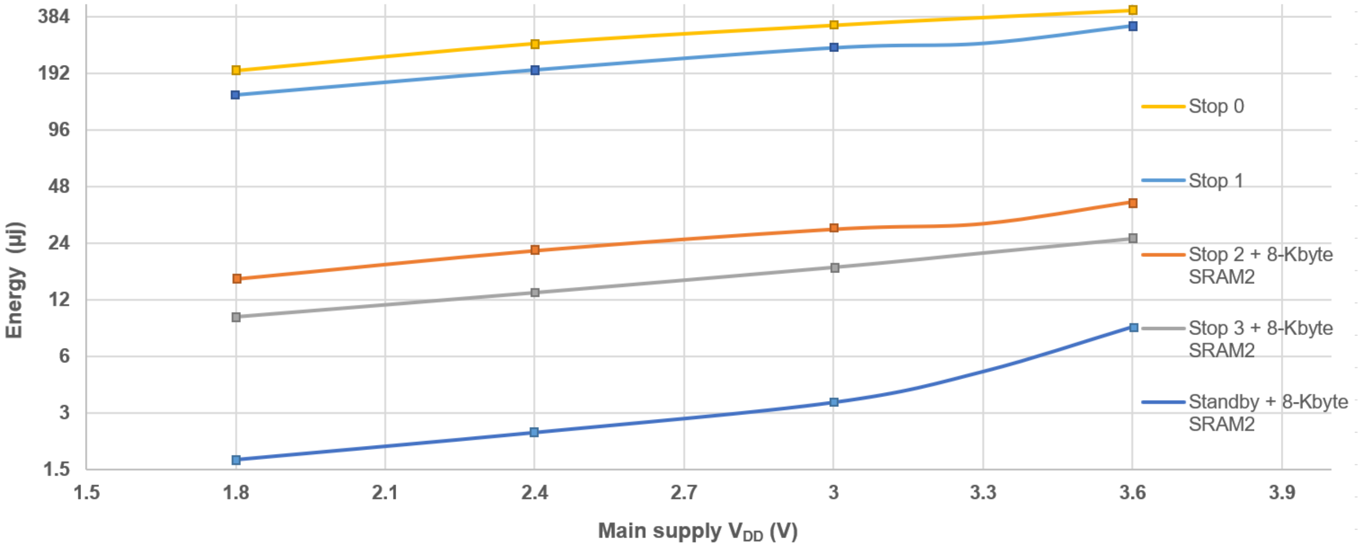
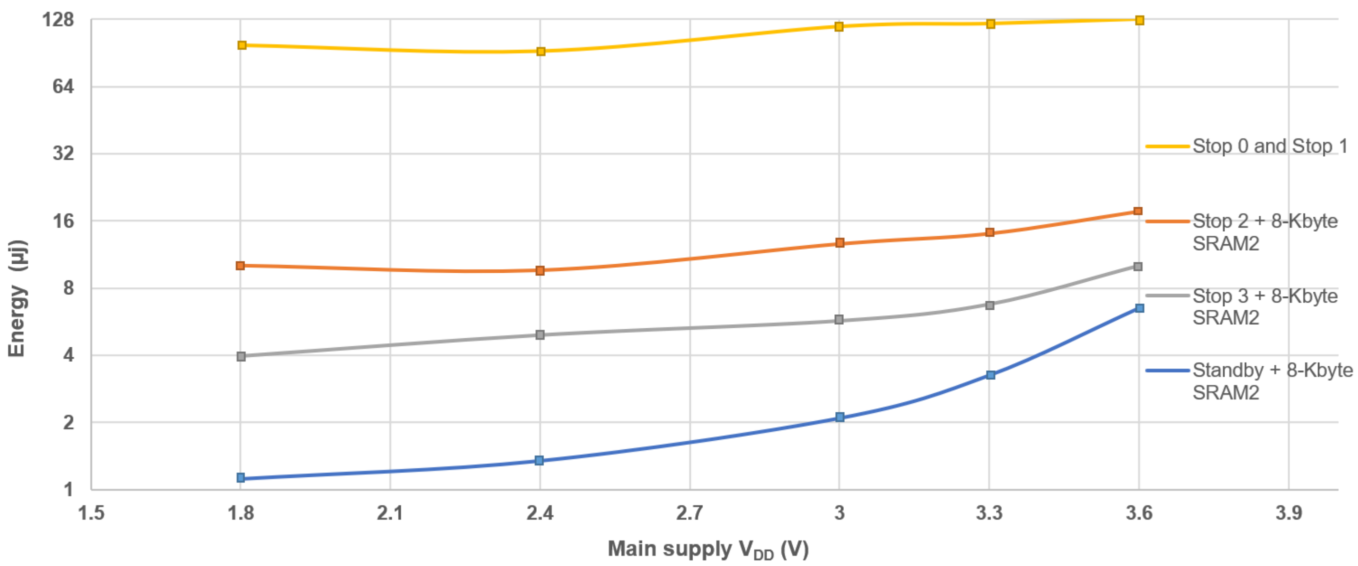


Figure 7. STM32U575/U585 low-power mode influence on average energy consumption (SMPS)



The previous figures show that for applications requiring some data retention, Standby mode gives the best energy consumption, and Stop 3 mode is the lowest power mode with all SRAM retention. Using SMPS halves energy consumption as shown in Figure 7.

Shutdown mode has been discarded (dotted line), because it does not present enough data retention capability for this type of application.

The low-power mode selection is not only dictated by the overall power consumption figure, but also by other wake-up considerations linked to the application. Figure 6 and Figure 7 show that, for a small additional power consumption versus Standby, Stop 2 and Stop 3 modes offer a much more powerful, simpler, and faster setup for waking up (without peripheral reinitialization).

Note:

1. *Average gain of about 16.5% in energy consumption for STM32U535/U545 compared to STM32U575/U585.*
2. *Energy consumption is increased by a factor of 2 for ST32U59x/U5Ax and STM32U5Fx/U5Gx compared to STM32U575/U585.*

4.4 Low-power mode selection

For a real application, the following rules apply, depending on the wake-up period:

- If the wake-up period is longer than some tens of ms, an implementation using Standby mode provides better consumption.
- If the wake-up period is shorter, an implementation with Stop 2 or Stop 3 mode provides better results.
- If the temperature is high, Standby mode is preferred.

The low-power mode selection depends not only on the power consumption, but also on the wake-up time requirement (system reactivity) and the requirement for data retention.

If the processing can accommodate a frequency below 24 MHz, the optimum points are the following:

- Standby or Stop 3 mode
  In Standby, the optimum points are a wake-up transition time longer than 66 µs, and retention area smaller than 64-Kbyte SRAM + 2-Kbyte BKPSRAM. In Stop 3 mode, all SRAMs can be retained.
- Stop 2 mode
  The optimum points are a wake-up transition time longer than 20 µs and retention area bigger than 66-Kbyte (64-Kbyte SRAM2 + 2-Kbyte BKPSRAM) retention.

If the wake-up period is longer than a few seconds and the retention memory is not needed, Shutdown mode provides the best power performance. In this case, the wake-up time is typically 610 µs.

4.5 Impact of transition

To complete this study, the energy spent during the transition phases (wake-up and deactivation) must be considered.

According to the document [2], the energy spent at V_{DD} = 3.0 V, with SMPS, when leaving a low-power mode for STM32U575/U585 microcontrollers is roughly:

- 1.1 nAs for a transition from Stop 1 to Run mode (MSI = 24 MHz)
- 0.57 nAs for a transition from Stop 2 to Run mode (MSI = 24 MHz)
- 4.54 nAs for a transition from Stop 3 to Run mode (MSI = 24 MHz)
- 21.22 nAs for a transition from Standby to Run mode (MSI = 4 MHz)
- 457.29 nAs for a transition from Shutdown to Run mode (MSI = 4 MHz)

5 ULPMark-CP use case optimizations

5.1 ULPMark-CP description

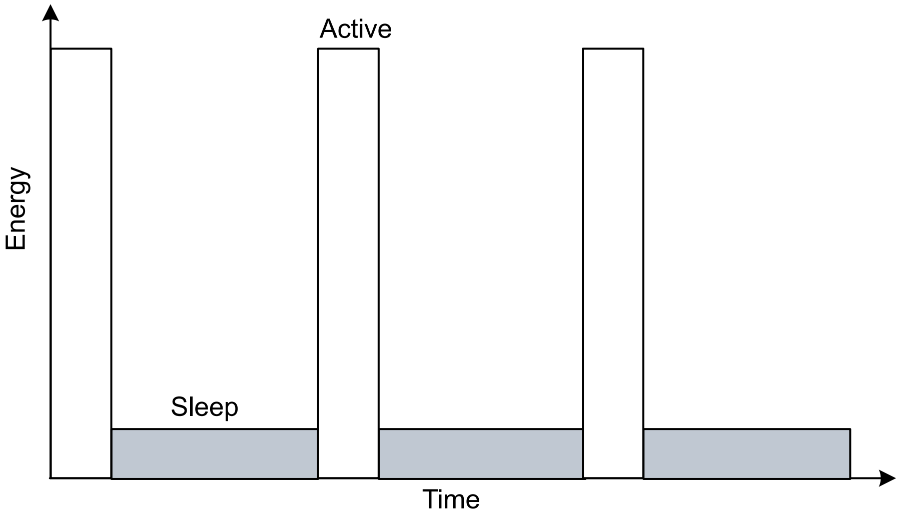
The ULPMark-Core Profile benchmark focuses on the MCU core, specifically the energy cost in sleep, and the transition to and from active mode. This benchmark uses a common set of workloads that are portable across 8-, 16-, and 32-bit microcontrollers. The Core Profile runs on a one-second duty cycle combining these workloads with an extended period of inactivity to enable the use of microcontroller low-power modes.

During the active portion of the test, the benchmark does the following:

- Generate 20 GPIO pulses.
- Perform an 8-bit linear interpolation.
- Perform a 16-bit integration (filter).
- Compute a 7-segment LCD binary conversion (saving state).
- Search for a substring in a string.
- Perform a small bubble-sort.
- Permute the bits of a string based on input and previous state.

The figure below shows the ULPMark-CP benchmark operating in long periods of sleep, followed by a brief wake-up to perform minimal processing, mimicking a sleepy-edge node conserving energy (see [5] for more details).

Figure 8. **ULPMark-CP use case description**



5.2 Use-case constraints

The ULPMark-CP imposes the following constraints:

- The application wakes up every second, based on an accurate RTC to perform some operations on a set of data.
- The data need to be maintained from one processing period to the next. This requires data retention during the inactive phase.
- The number of processing cycles required by the application, once compiled using the compiler strongest optimization option, is about 10000 cycles every second.

There are no particular constraints regarding the reactivity of the system, in order to serve external events or the RTC periodic interrupts.

5.3 Process phase optimization

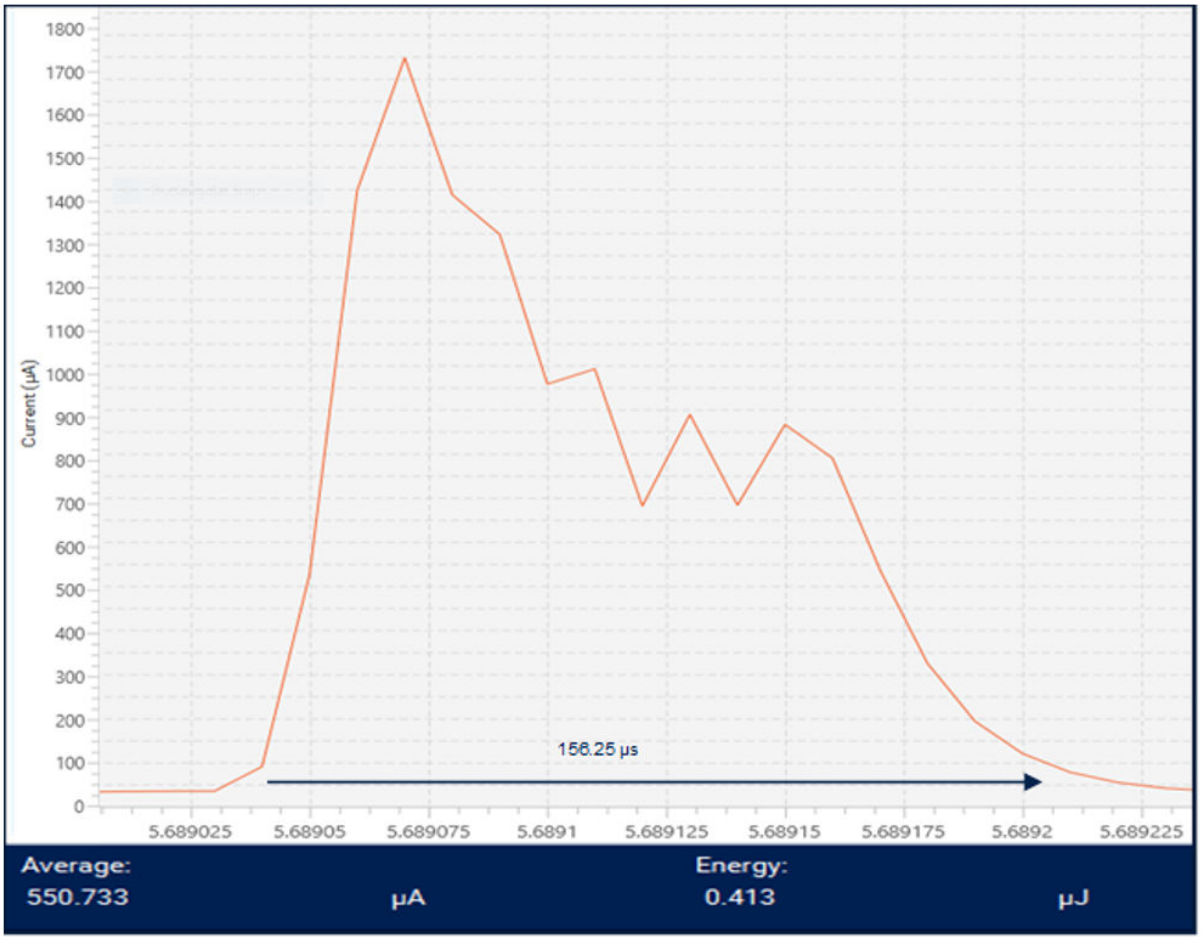
The active portion of the benchmark is only running for ~3% of the total runtime. The user can therefore use the power optimum point corresponding to the voltage scaling range 4 at 16 MHz, using the internal MSIS. When using the MSIS at 16 MHz nominal frequency, it consumes less than any other solution using the PLL.

To further optimize power consumption in active phase, the following steps can be applied:

1. Power down SRAM1, SRAM2, SRAM3, SRAM4 in Run mode.
2. Configure system clock as MSIS voltage scaling range 4 at 16 MHz.
3. Put data in BKPSRAM.
4. Disable unused AHB/APB clock.
5. Put all GPIO pins in analog mode.
6. Power down Flash Bank 2.
7. Enable the flash memory low-power mode.
8. Configure compiler optimization (high speed, no size constraints).

The figure below shows ULPMark-CP in active phase. It takes ~156 µs between wake-up and entering in low-power mode with 550 µA current consumption in Run mode at system clock MSIS 16 MHz.

Figure 9. STM32U575/U585 ULPMark-CP in active phase



Average: 550.733 µA

Energy: 0.413 µJ

5.4 Inactive phase optimization

The ULPMark-CP benchmark requires to keep the data variables unchanged in low-power mode from one run to the next. Therefore, the Shutdown mode cannot be used because it does not provide enough data retention capability for this application.

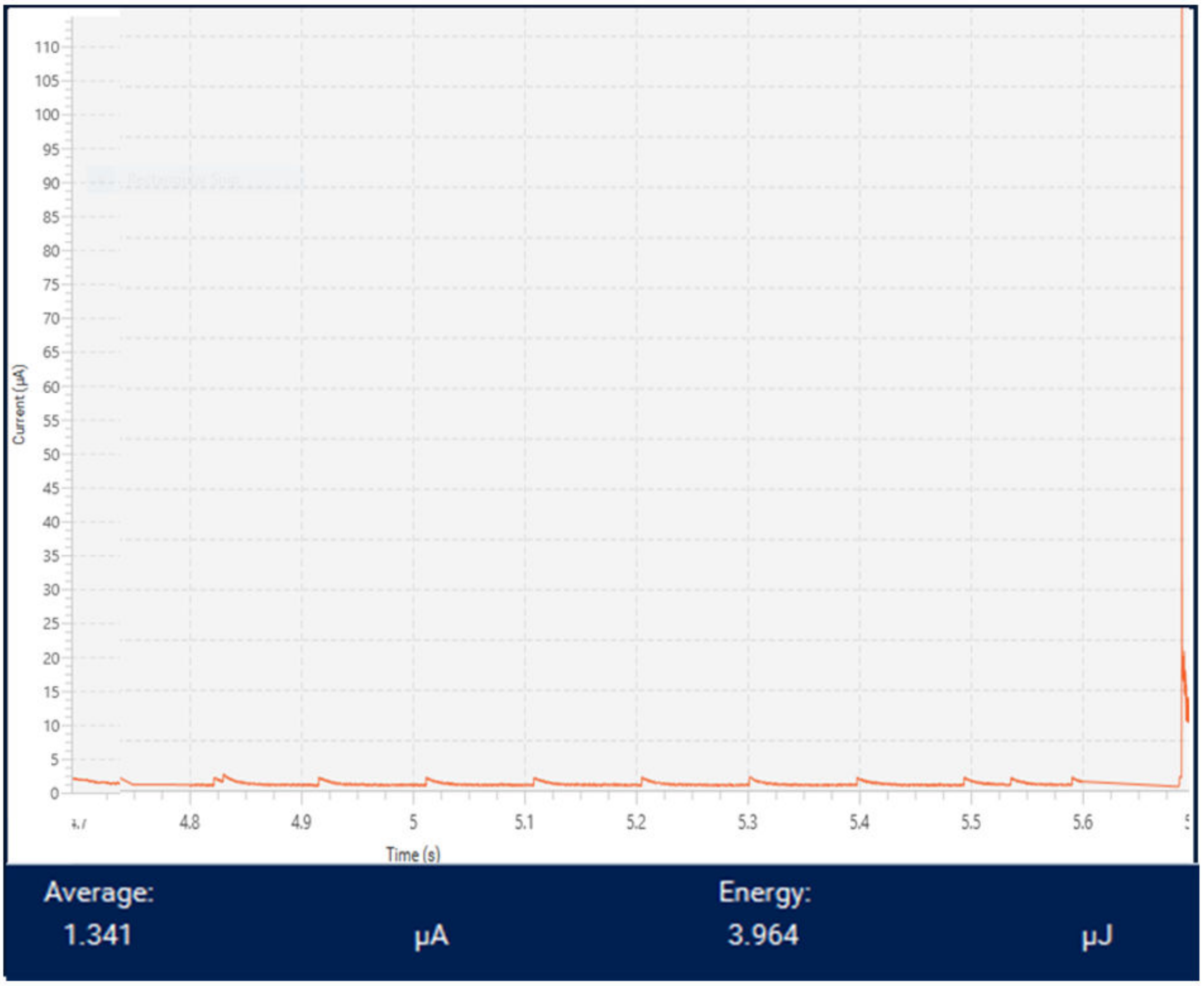
Stop 3 mode can be used with 8-Kbyte SRAM2 or 2-Kbyte BKPSRAM retention and with the lowest wake-up time. However, the most power-efficient solution uses Standby mode with RTC and 8-Kbyte SRAM2 or 2-Kbyte BKPSRAM retention.

To further optimize power consumption in inactive phase, the following steps can be applied:

1. Enable SMPS regulator.
2. Enable ULPMEN bit.
3. Power down unused SRAMs in Stop 3 mode.
4. Disable all AHB/APB peripheral clocks (except AHB3/APB3 for RTC).

The figure below shows ULPMark-CP in inactive phase in Standby mode with an average current consumption of 1.34 µA.

Figure 10. STM32U575/U585 ULPMark-CP in inactive phase with Standby mode

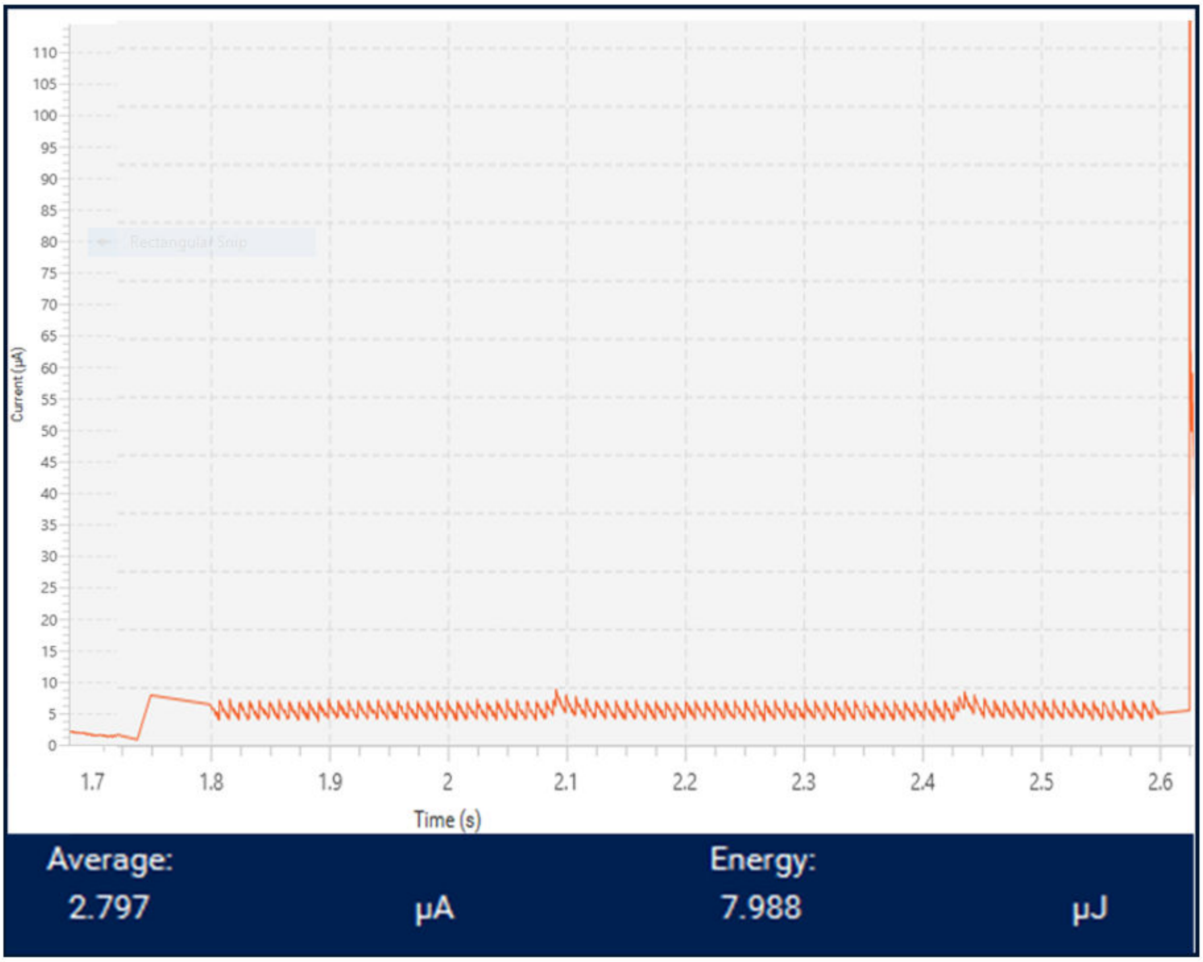


Average:
1.341 µA

Energy:
3.964 µJ

The figure below shows ULPMark-CP in inactive phase in Stop 3 mode with an average current consumption of 2.97 µA.

Figure 11. **STM32U575/U585 ULPMark-CP in inactive phase with Stop 3 mode**

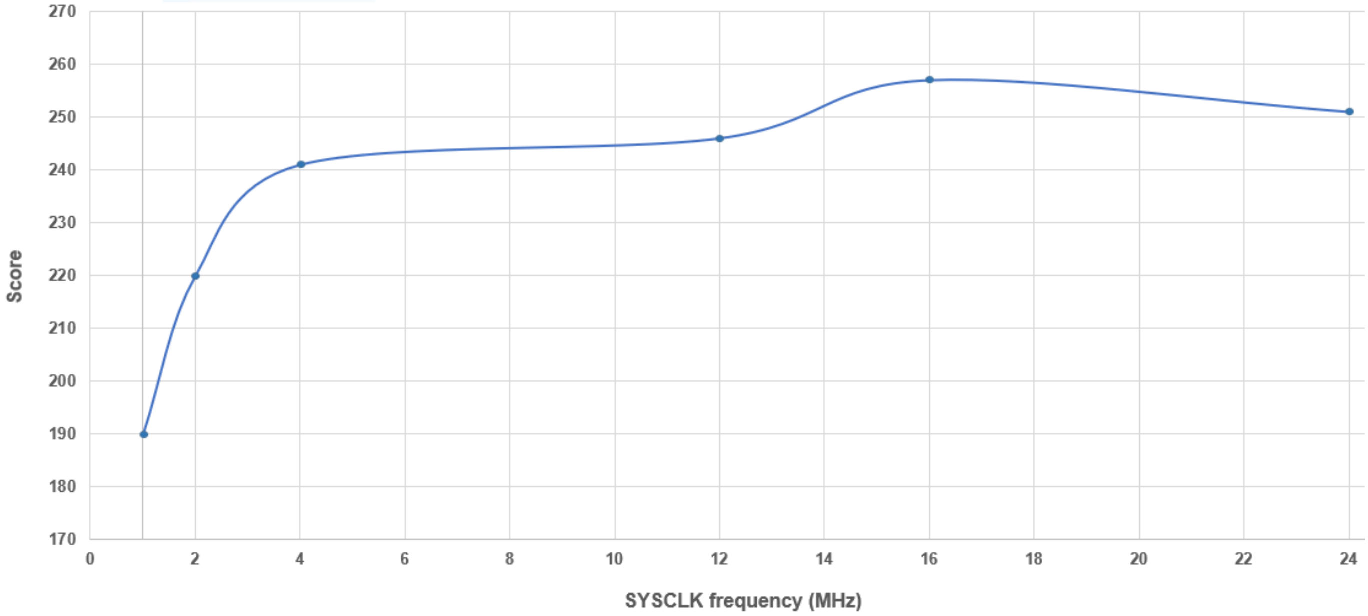


In the case of the ULPBench benchmark with a wake-up period of one second and a processing load of about 10 Kcycles per period, the optimum choice is to use Standby mode for the inactive phase, associated with the voltage scaling range 4 mode at 16 MHz (from the MSIS) for the process phase.

5.5 STM32U575/U585 measurement results

The figure below shows the ULPMark-CP score measurements at different system clock frequencies used during the process phase. It uses Standby mode with BKPSRAM retention and RTC for the inactive phase.

Figure 12. STM32U575/U585 ULPMark-CP measurements versus frequency



The results are provided with different clock configurations; for frequencies below 24 MHz, the MSI is used (because automatically started when waking up from Standby mode).

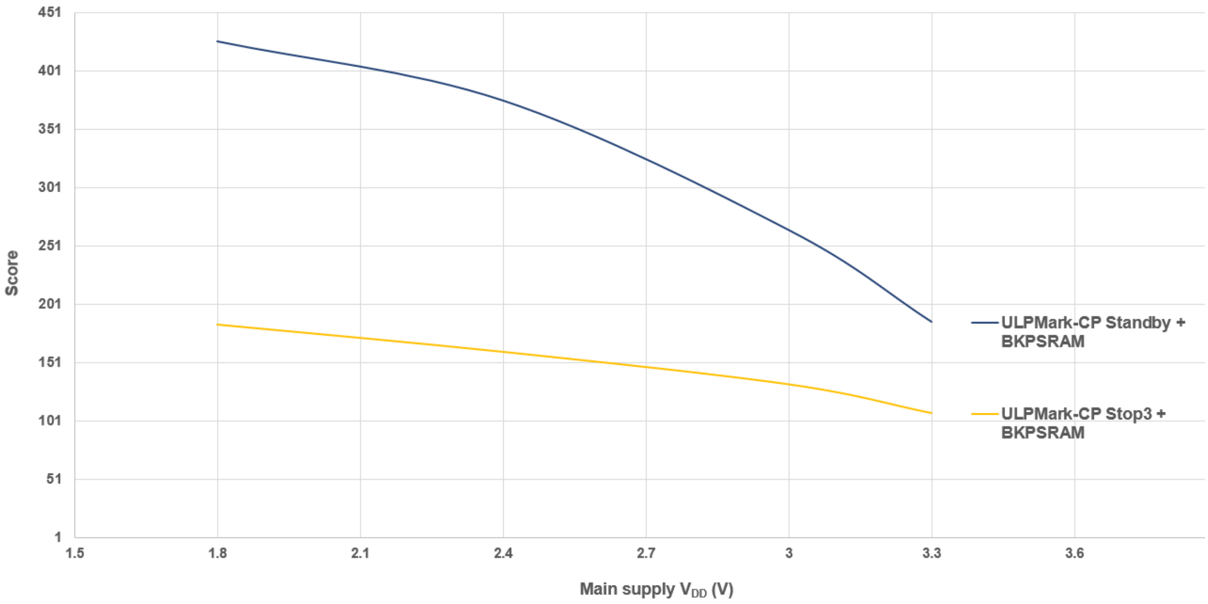
As already seen in the theoretical study, the voltage scaling range 4 at 16 MHz using the MSIS provides the best results.

5.6 Performance evolution with voltage range

EEMBC defined base-test defaults on 3.0 V, but this voltage is not representative of battery-supplied low-power applications. When the LDO regulator is used, the gain in power when moving to a lower voltage is significant compared to staying at 3.0 V. On the contrary, power consumption decreases when increasing V_{DD} with the SMPS regulator. This is why EEMBC introduced a comparison on user-defined voltage, letting manufacturers choose ideal operating conditions.

The figure below shows the ULPMark-CP score measurements at different voltages.

Figure 13. STM32U575/U585 equivalent score versus supply voltage



The curves above correspond to the voltage scaling range 4 at 16 MHz MSIS mode. It clearly shows the advantages of a voltage reduction.

6 Conclusion

The main ultralow-power features of the STM32U5 series microcontrollers are presented in this application note. They show the benefits offered by this microcontroller family to reduce current consumption in embedded systems. These devices offer a large choice of options for optimizing both performance and power consumption, whatever the application.

This document provides guidelines based on experiments and quantitative results to quickly select the best running and low-power modes, according to the characteristics and constraints of the end-user application.

Revision history

Table 6. **Document revision history**

Date	Version	Changes
6-Oct-2021	1	Initial release.
16-Feb-2023	2	Updated: • Section Introduction • Section 1 General information • Section 2 Energy efficiency processing • Section 2.2 ICACHE efficiency • Section 2.3 Flash memory efficiency • Section 3 STM32U5 series ultralow-power feature • Section 3.1 Low-power modes • Section 3.2.2 FLASH in Run and Sleep modes • Section 3.2.3 Power control optimization • Section 3.2.6 Internal SRAMs • Section 3.2.7 Peripherals clock gating in Run and Stop modes • Section 3.3 Power optimization of peripherals using LPBAM • Section 3.3.1 Peripherals supporting autonomous mode • Section 4.2 Inactive phase • Section 4.3 Sleep mode selection • Section 4.5 Impact of transition • Section 5.3 Process phase optimization • Section 5.4 Inactive phase optimization • Section 5.5 STM32U575/U585 measurement results • Section 5.6 Performance evolution with voltage range • Section 6 Conclusion
27-Jun-2023	3	Updated: • Section Introduction • Section 1 General information: list of datasheets updated • Low-power background autonomous mode (LPBAM) • Section 3.3 Power optimization of peripherals using LPBAM • Section 3.3.1 Peripherals supporting autonomous mode • Grammar, spelling and punctation in all sections • Title, to comply with new AN title rules

Contents

List of tables

List of figures