51 HAL RCC Generic Driver

51.1 RCC Firmware driver registers structures

51.1.1 RCC_PLLInitTypeDef

Data Fields

- ***uint32_t PLLState***
- ***uint32_t PLLSource***
- ***uint32_t PLLM***
- ***uint32_t PLLN***
- ***uint32_t PLLP***
- ***uint32_t PLLQ***
- ***uint32_t PLLR***

Field Documentation

- ***uint32_t RCC_PLLInitTypeDef::PLLState***
  The new state of the PLL. This parameter can be a value of ***RCC_PLL_Config***
- ***uint32_t RCC_PLLInitTypeDef::PLLSource***
  RCC_PLLSource: PLL entry clock source. This parameter must be a value of ***RCC_PLL_Clock_Source***
- ***uint32_t RCC_PLLInitTypeDef::PLLM***
  PLLM: Division factor for PLL VCO input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16 on STM32L4Rx/STM32L4Sx devices. This parameter must be a number between Min_Data = 1 and Max_Data = 8 on the other devices
- ***uint32_t RCC_PLLInitTypeDef::PLLN***
  PLLN: Multiplication factor for PLL VCO output clock. This parameter must be a number between Min_Data = 8 and Max_Data = 86
- ***uint32_t RCC_PLLInitTypeDef::PLLP***
  PLLP: Division factor for SAI clock. This parameter must be a value of ***RCC_PLLP_Clock_Divider***
- ***uint32_t RCC_PLLInitTypeDef::PLLQ***
  PLLQ: Division factor for SDMMC1, RNG and USB clocks. This parameter must be a value of ***RCC_PLLQ_Clock_Divider***
- ***uint32_t RCC_PLLInitTypeDef::PLLR***
  PLLR: Division for the main system clock. User have to set the PLLR parameter correctly to not exceed max frequency 80MHZ. This parameter must be a value of ***RCC_PLLR_Clock_Divider***

51.1.2 RCC_OscInitTypeDef

Data Fields

- ***uint32_t OscillatorType***
- ***uint32_t HSEState***
- ***uint32_t LSEState***
- ***uint32_t HSIState***
- ***uint32_t HSICalibrationValue***
- ***uint32_t LSIState***
- ***uint32_t MSIState***
- ***uint32_t MSICalibrationValue***

- ***uint32_t MSIClockRange***
- ***uint32_t HSI48State***
- ***RCC_PLLInitTypeDef PLL***

Field Documentation

- ***uint32_t RCC_OscInitTypeDef::OscillatorType***
  The oscillators to be configured. This parameter can be a value of ***RCC_Oscillator_Type***
- ***uint32_t RCC_OscInitTypeDef::HSEState***
  The new state of the HSE. This parameter can be a value of ***RCC_HSE_Config***
- ***uint32_t RCC_OscInitTypeDef::LSEState***
  The new state of the LSE. This parameter can be a value of ***RCC_LSE_Config***
- ***uint32_t RCC_OscInitTypeDef::HSIState***
  The new state of the HSI. This parameter can be a value of ***RCC_HSI_Config***
- ***uint32_t RCC_OscInitTypeDef::HSICalibrationValue***
  The calibration trimming value (default is RCC_HSICALIBRATION_DEFAULT). This parameter must be a number between Min_Data = 0x00 and Max_Data = 0x1F on STM32L43x/STM32L44x/STM32L47x/STM32L48x devices. This parameter must be a number between Min_Data = 0x00 and Max_Data = 0x7F on the other devices
- ***uint32_t RCC_OscInitTypeDef::LSIState***
  The new state of the LSI. This parameter can be a value of ***RCC_LSI_Config***
- ***uint32_t RCC_OscInitTypeDef::MSIState***
  The new state of the MSI. This parameter can be a value of ***RCC_MSI_Config***
- ***uint32_t RCC_OscInitTypeDef::MSICalibrationValue***
  The calibration trimming value (default is RCC_MSICALIBRATION_DEFAULT). This parameter must be a number between Min_Data = 0x00 and Max_Data = 0xFF
- ***uint32_t RCC_OscInitTypeDef::MSIClockRange***
  The MSI frequency range. This parameter can be a value of ***RCC_MSI_Clock_Range***
- ***uint32_t RCC_OscInitTypeDef::HSI48State***
  The new state of the HSI48 (only applicable to STM32L43x/STM32L44x/STM32L49x/STM32L4Ax devices). This parameter can be a value of ***RCC_HSI48_Config***
- ***RCC_PLLInitTypeDef RCC_OscInitTypeDef::PLL***
  Main PLL structure parameters

51.1.3 RCC_ClkInitTypeDef

Data Fields

- ***uint32_t ClockType***
- ***uint32_t SYSCLKSource***
- ***uint32_t AHBCLKDivider***
- ***uint32_t APB1CLKDivider***
- ***uint32_t APB2CLKDivider***

Field Documentation

- ***uint32_t RCC_ClkInitTypeDef::ClockType***
  The clock to be configured. This parameter can be a value of ***RCC_System_Clock_Type***
- ***uint32_t RCC_ClkInitTypeDef::SYSCLKSource***
  The clock source used as system clock (SYSCLK). This parameter can be a value of ***RCC_System_Clock_Source***
- ***uint32_t RCC_ClkInitTypeDef::AHBCLKDivider***
  The AHB clock (HCLK) divider. This clock is derived from the system clock (SYSCLK). This parameter can be a value of ***RCC_AHB_Clock_Source***

- ***uint32_t RCC_ClkInitTypeDef::APB1CLKDivider***
  The APB1 clock (PCLK1) divider. This clock is derived from the AHB clock (HCLK). This parameter can be a value of ***RCC_APB1_APB2_Clock_Source***
- ***uint32_t RCC_ClkInitTypeDef::APB2CLKDivider***
  The APB2 clock (PCLK2) divider. This clock is derived from the AHB clock (HCLK). This parameter can be a value of ***RCC_APB1_APB2_Clock_Source***

51.2 RCC Firmware driver API description

51.2.1 RCC specific features

After reset the device is running from Multiple Speed Internal oscillator (4 MHz) with Flash 0 wait state. Flash prefetch buffer, D-Cache and I-Cache are disabled, and all peripherals are off except internal SRAM, Flash and JTAG.

- There is no prescaler on High speed (AHBs) and Low speed (APBs) busses: all peripherals mapped on these busses are running at MSI speed.
- The clock for all peripherals is switched off, except the SRAM and FLASH.
- All GPIOs are in analog mode, except the JTAG pins which are assigned to be used for debug purpose.

Once the device started from reset, the user application has to:

- Configure the clock source to be used to drive the System clock (if the application needs higher frequency/performance)
- Configure the System clock frequency and Flash settings
- Configure the AHB and APB busses prescalers
- Enable the clock for the peripheral(s) to be used
- Configure the clock source(s) for peripherals which clocks are not derived from the System clock (SAIx, RTC, ADC, USB OTG FS/SDMMC1/RNG)

51.2.2 Initialization and de-initialization functions

This section provides functions allowing to configure the internal and external oscillators (HSE, HSI, LSE, MSI, LSI, PLL, CSS and MCO) and the System busses clocks (SYSCLK, AHB, APB1 and APB2).

Internal/external clock and PLL configuration

- HSI (high-speed internal): 16 MHz factory-trimmed RC used directly or through the PLL as System clock source.
- MSI (Mutiple Speed Internal): Its frequency is software trimmable from 100KHZ to 48MHZ. It can be used to generate the clock for the USB OTG FS (48 MHz). The number of flash wait states is automatically adjusted when MSI range is updated with HAL_RCC_OscConfig() and the MSI is used as System clock source.
- LSI (low-speed internal): 32 KHz low consumption RC used as IWDG and/or RTC clock source.
- HSE (high-speed external): 4 to 48 MHz crystal oscillator used directly or through the PLL as System clock source. Can be used also optionally as RTC clock source.
- LSE (low-speed external): 32.768 KHz oscillator used optionally as RTC clock source.
- PLL (clocked by HSI, HSE or MSI) providing up to three independent output clocks:
  - The first output is used to generate the high speed system clock (up to 80MHz).
  - The second output is used to generate the clock for the USB OTG FS (48 MHz), the random analog generator (<=48 MHz) and the SDMMC1 (<= 48 MHz).
  - The third output is used to generate an accurate clock to achieve high-quality audio performance on SAI interface.

- PLLSAI1 (clocked by HSI, HSE or MSI) providing up to three independent output clocks:
  - The first output is used to generate SAR ADC1 clock.
  - The second output is used to generate the clock for the USB OTG FS (48 MHz), the random analog generator (<=48 MHz) and the SDMMC1 (<= 48 MHz).
  - The Third output is used to generate an accurate clock to achieve high-quality audio performance on SAI interface.
- PLLSAI2 (clocked by HSI , HSE or MSI) providing up to two independent output clocks:
  - The first output is used to generate SAR ADC2 clock.
  - The second output is used to generate an accurate clock to achieve high-quality audio performance on SAI interface.
- CSS (Clock security system): once enabled, if a HSE clock failure occurs (HSE used directly or through PLL as System clock source), the System clock is automatically switched to HSI and an interrupt is generated if enabled. The interrupt is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.
- MCO (microcontroller clock output): used to output MSI, LSI, HSI, LSE, HSE or main PLL clock (through a configurable prescaler) on PA8 pin.

System, AHB and APB busses clocks configuration

- Several clock sources can be used to drive the System clock (SYSCLK): MSI, HSI, HSE and main PLL. The AHB clock (HCLK) is derived from System clock through configurable prescaler and used to clock the CPU, memory and peripherals mapped on AHB bus (DMA, GPIO...). APB1 (PCLK1) and APB2 (PCLK2) clocks are derived from AHB clock through configurable prescalers and used to clock the peripherals mapped on these busses. You can use "HAL_RCC_GetSysClockFreq()" function to retrieve the frequencies of these clocks. All the peripheral clocks are derived from the System clock (SYSCLK) except: SAI: the SAI clock can be derived either from a specific PLL (PLLSAI1) or (PLLSAI2) or from an external clock mapped on the SAI_CKIN pin. You have to use HAL_RCCEx_PeriphCLKConfig() function to configure this clock.RTC: the RTC clock can be derived either from the LSI, LSE or HSE clock divided by 2 to 31. You have to use __HAL_RCC_RTC_ENABLE() and HAL_RCCEx_PeriphCLKConfig() function to configure this clock.USB OTG FS, SDMMC1 and RNG: USB OTG FS requires a frequency equal to 48 MHz to work correctly, while the SDMMC1 and RNG peripherals require a frequency equal or lower than to 48 MHz. This clock is derived of the main PLL or PLLSAI1 through PLLQ divider. You have to enable the peripheral clock and use HAL_RCCEx_PeriphCLKConfig() function to configure this clock.IWDG clock which is always the LSI clock.
- The maximum frequency of the SYSCLK, HCLK, PCLK1 and PCLK2 is 80 MHz. The clock source frequency should be adapted depending on the device voltage range as listed in the Reference Manual "Clock source frequency versus voltage scaling" chapter.

This section contains the following APIs:

- ***HAL_RCC_DeInit()***
- ***HAL_RCC_OscConfig()***
- ***HAL_RCC_ClockConfig()***

51.2.3 Peripheral Control functions

This subsection provides a set of functions allowing to:

- Ouput clock to MCO pin.
- Retrieve current clock frequencies.

- Enable the Clock Security System.

This section contains the following APIs:

- ***HAL_RCC_MCOConfig()***
- ***HAL_RCC_GetSysClockFreq()***
- ***HAL_RCC_GetHCLKFreq()***
- ***HAL_RCC_GetPCLK1Freq()***
- ***HAL_RCC_GetPCLK2Freq()***
- ***HAL_RCC_GetOscConfig()***
- ***HAL_RCC_GetClockConfig()***
- ***HAL_RCC_EnableCSS()***
- ***HAL_RCC_NMI_IRQHandler()***
- ***HAL_RCC_CSSCallback()***

51.2.4 Detailed description of functions

HAL_RCC_DeInit

Function name **void HAL_RCC_DeInit (void)**

Function description Reset the RCC clock configuration to the default reset state.

Return values
- **None:**

Notes
- The default reset state of the clock configuration is given below: MSI ON and used as system clock sourceHSE, HSI, PLL, PLLSAI1 and PLLISAI2 OFFAHB, APB1 and APB2 prescaler set to 1.CSS, MCO1 OFFAll interrupts disabled
- This function doesn't modify the configuration of the Peripheral clocksLSI, LSE and RTC clocks

HAL_RCC_OscConfig

Function name **HAL_StatusTypeDef HAL_RCC_OscConfig (RCC_OscInitTypeDef * RCC_OscInitStruct)**

Function description Initialize the RCC Oscillators according to the specified parameters in the RCC_OscInitTypeDef.

Parameters
- **RCC_OscInitStruct:** pointer to an RCC_OscInitTypeDef structure that contains the configuration information for the RCC Oscillators.

Return values
- **HAL:** status

Notes
- The PLL is not disabled when used as system clock.
- Transitions LSE Bypass to LSE On and LSE On to LSE Bypass are not supported by this macro. User should request a transition to LSE Off first and then LSE On or LSE Bypass.
- Transition HSE Bypass to HSE On and HSE On to HSE Bypass are not supported by this macro. User should request a transition to HSE Off first and then HSE On or HSE Bypass.

HAL_RCC_ClockConfig

Function name **HAL_StatusTypeDef HAL_RCC_ClockConfig (RCC_ClkInitTypeDef * RCC_ClkInitStruct, uint32_t FLatency)**

Function description: Initialize the CPU, AHB and APB busses clocks according to the specified parameters in the RCC_ClkInitStruct.

Parameters:

- **RCC_ClkInitStruct:** pointer to an RCC_OscInitTypeDef structure that contains the configuration information for the RCC peripheral.
- **FLatency:** FLASH Latency This parameter can be one of the following values:
  - FLASH_LATENCY_0 FLASH 0 Latency cycle
  - FLASH_LATENCY_1 FLASH 1 Latency cycle
  - FLASH_LATENCY_2 FLASH 2 Latency cycles
  - FLASH_LATENCY_3 FLASH 3 Latency cycles
  - FLASH_LATENCY_4 FLASH 4 Latency cycles
  - FLASH_LATENCY_5 FLASH 5 Latency cycles
  - FLASH_LATENCY_6 FLASH 6 Latency cycles
  - FLASH_LATENCY_7 FLASH 7 Latency cycles
  - FLASH_LATENCY_8 FLASH 8 Latency cycles
  - FLASH_LATENCY_9 FLASH 9 Latency cycles
  - FLASH_LATENCY_10 FLASH 10 Latency cycles
  - FLASH_LATENCY_11 FLASH 11 Latency cycles
  - FLASH_LATENCY_12 FLASH 12 Latency cycles
  - FLASH_LATENCY_13 FLASH 13 Latency cycles
  - FLASH_LATENCY_14 FLASH 14 Latency cycles
  - FLASH_LATENCY_15 FLASH 15 Latency cycles

Return values:

- **None:**

Notes:

- The SystemCoreClock CMSIS variable is used to store System Clock Frequency and updated by HAL_RCC_GetHCLKFreq() function called within this function
- The MSI is used by default as system clock source after startup from Reset, wake-up from STANDBY mode. After restart from Reset, the MSI frequency is set to its default value 4 MHz.
- The HSI can be selected as system clock source after from STOP modes or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled).
- A switch from one clock source to another occurs only if the target clock source is ready (clock stable after startup delay or PLL locked). If a clock source which is not yet ready is selected, the switch will occur when the clock source is ready.
- You can use HAL_RCC_GetClockConfig() function to know which clock is currently used as system clock source.
- Depending on the device voltage range, the software has to set correctly HPRE[3:0] bits to ensure that HCLK not exceed the maximum allowed frequency (for more details refer to section above "Initialization/de-initialization functions")

HAL_RCC_MCOConfig

Function name: **void HAL_RCC_MCOConfig (uint32_t RCC_MCOx, uint32_t RCC_MCOSource, uint32_t RCC_MCODiv)**

Function description: Select the clock source to output on MCO pin(PA8).

Parameters

- **RCC_MCOx:** specifies the output direction for the clock source. For STM32L4xx family this parameter can have only one value:
  - RCC_MCO1 Clock source to output on MCO1 pin(PA8).
- **RCC_MCOSource:** specifies the clock source to output. This parameter can be one of the following values:
  - RCC_MCO1SOURCE_NOCLOCK MCO output disabled, no clock on MCO
  - RCC_MCO1SOURCE_SYSCLK system clock selected as MCO source
  - RCC_MCO1SOURCE_MSI MSI clock selected as MCO source
  - RCC_MCO1SOURCE_HSI HSI clock selected as MCO source
  - RCC_MCO1SOURCE_HSE HSE clock selected as MCO sourcee
  - RCC_MCO1SOURCE_PLLCLK main PLL clock selected as MCO source
  - RCC_MCO1SOURCE_LSI LSI clock selected as MCO source
  - RCC_MCO1SOURCE_LSE LSE clock selected as MCO source
- **RCC_MCODiv:** specifies the MCO prescaler. This parameter can be one of the following values:
  - RCC_MCODIV_1 no division applied to MCO clock
  - RCC_MCODIV_2 division by 2 applied to MCO clock
  - RCC_MCODIV_4 division by 4 applied to MCO clock
  - RCC_MCODIV_8 division by 8 applied to MCO clock
  - RCC_MCODIV_16 division by 16 applied to MCO clock

Return values

- **None:**

Notes

- PA8 should be configured in alternate function mode.

HAL_RCC_EnableCSS

Function name **void HAL_RCC_EnableCSS (void)**

Function description Enable the Clock Security System.

Return values

- **None:**

Notes

- If a failure is detected on the HSE oscillator clock, this oscillator is automatically disabled and an interrupt is generated to inform the software about the failure (Clock Security System Interrupt, CSSI), allowing the MCU to perform rescue operations. The CSSI is linked to the Cortex-M4 NMI (Non-Maskable Interrupt) exception vector.
- The Clock Security System can only be cleared by reset.

HAL_RCC_GetSysClockFreq

Function name **uint32_t HAL_RCC_GetSysClockFreq (void)**

Function description Return the SYSCLK frequency.

Return values

- **SYSCLK:** frequency

Notes
- The system frequency computed by this function is not the real frequency in the chip. It is calculated based on the predefined constant and the selected clock source:
- If SYSCLK source is MSI, function returns values based on MSI Value as defined by the MSI range.
- If SYSCLK source is HSI, function returns values based on HSI_VALUE(*)
- If SYSCLK source is HSE, function returns values based on HSE_VALUE(**)
- If SYSCLK source is PLL, function returns values based on HSE_VALUE(**), HSI_VALUE(*) or MSI Value multiplied/divided by the PLL factors.
- (*) HSI_VALUE is a constant defined in stm32l4xx_hal_conf.h file (default value 16 MHz) but the real value may vary depending on the variations in voltage and temperature.
- (**) HSE_VALUE is a constant defined in stm32l4xx_hal_conf.h file (default value 8 MHz), user has to ensure that HSE_VALUE is same as the real frequency of the crystal used. Otherwise, this function may have wrong result.
- The result of this function could be not correct when using fractional value for HSE crystal.
- This function can be used by the user application to compute the baudrate for the communication peripherals or configure other parameters.
- Each time SYSCLK changes, this function must be called to update the right SYSCLK value. Otherwise, any configuration based on this function will be incorrect.

HAL_RCC_GetHCLKFreq

Function name **uint32_t HAL_RCC_GetHCLKFreq (void)**

Function description Return the HCLK frequency.

Return values
- **HCLK:** frequency in Hz

Notes
- Each time HCLK changes, this function must be called to update the right HCLK value. Otherwise, any configuration based on this function will be incorrect.
- The SystemCoreClock CMSIS variable is used to store System Clock Frequency.

HAL_RCC_GetPCLK1Freq

Function name **uint32_t HAL_RCC_GetPCLK1Freq (void)**

Function description Return the PCLK1 frequency.

Return values
- **PCLK1:** frequency in Hz

Notes
- Each time PCLK1 changes, this function must be called to update the right PCLK1 value. Otherwise, any configuration based on this function will be incorrect.

HAL_RCC_GetPCLK2Freq

Function name	**uint32_t HAL_RCC_GetPCLK2Freq (void)**
Function description	Return the PCLK2 frequency.
Return values	• **PCLK2:** frequency in Hz
Notes	• Each time PCLK2 changes, this function must be called to update the right PCLK2 value. Otherwise, any configuration based on this function will be incorrect.

HAL_RCC_GetOscConfig

Function name	**void HAL_RCC_GetOscConfig (RCC_OscInitTypeDef * RCC_OscInitStruct)**
Function description	Configure the RCC_OscInitStruct according to the internal RCC configuration registers.
Parameters	• **RCC_OscInitStruct:** pointer to an RCC_OscInitTypeDef structure that will be configured.
Return values	• **None:**

HAL_RCC_GetClockConfig

Function name	**void HAL_RCC_GetClockConfig (RCC_ClkInitTypeDef * RCC_ClkInitStruct, uint32_t * pFLatency)**
Function description	Configure the RCC_ClkInitStruct according to the internal RCC configuration registers.
Parameters	• **RCC_ClkInitStruct:** pointer to an RCC_ClkInitTypeDef structure that will be configured. • **pFLatency:** Pointer on the Flash Latency.
Return values	• **None:**

HAL_RCC_NMI_IRQHandler

Function name	**void HAL_RCC_NMI_IRQHandler (void)**
Function description	Handle the RCC Clock Security System interrupt request.
Return values	• **None:**
Notes	• This API should be called under the NMI_Handler().

HAL_RCC_CSSCallback

Function name	**void HAL_RCC_CSSCallback (void)**
Function description	RCC Clock Security System interrupt callback.
Return values	• **none:**

51.3 RCC Firmware driver defines

51.3.1 RCC

AHB1 Peripheral Clock Sleep Enable Disable

__HAL_RCC_DMA1_CLK_SLEEP_ENABLE

__HAL_RCC_DMA2_CLK_SLEEP_ENABLE

__HAL_RCC_DMAMUX1_CLK_SLEEP_ENABLE

__HAL_RCC_FLASH_CLK_SLEEP_ENABLE

__HAL_RCC_SRAM1_CLK_SLEEP_ENABLE

__HAL_RCC_CRC_CLK_SLEEP_ENABLE

__HAL_RCC_TSC_CLK_SLEEP_ENABLE

__HAL_RCC_DMA2D_CLK_SLEEP_ENABLE

__HAL_RCC_GFXMMU_CLK_SLEEP_ENABLE

__HAL_RCC_DMA1_CLK_SLEEP_DISABLE

__HAL_RCC_DMA2_CLK_SLEEP_DISABLE

__HAL_RCC_DMAMUX1_CLK_SLEEP_DISABLE

__HAL_RCC_FLASH_CLK_SLEEP_DISABLE

__HAL_RCC_SRAM1_CLK_SLEEP_DISABLE

__HAL_RCC_CRC_CLK_SLEEP_DISABLE

__HAL_RCC_TSC_CLK_SLEEP_DISABLE

__HAL_RCC_DMA2D_CLK_SLEEP_DISABLE

__HAL_RCC_GFXMMU_CLK_SLEEP_DISABLE

AHB1 Peripheral Clock Sleep Enabled or Disabled Status

__HAL_RCC_DMA1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DMA2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DMAMUX1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_FLASH_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SRAM1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_CRC_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TSC_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DMA2D_IS_CLK_SLEEP_ENABLED

__HAL_RCC_GFXMMU_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DMA1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_DMA2_IS_CLK_SLEEP_DISABLED

__HAL_RCC_DMAMUX1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_FLASH_IS_CLK_SLEEP_DISABLED

__HAL_RCC_SRAM1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_CRC_IS_CLK_SLEEP_DISABLED

__HAL_RCC_TSC_IS_CLK_SLEEP_DISABLED

__HAL_RCC_DMA2D_IS_CLK_SLEEP_DISABLED

__HAL_RCC_GFXMMU_IS_CLK_SLEEP_DISABLED

AHB1 Peripheral Force Release Reset

__HAL_RCC_AHB1_FORCE_RESET

__HAL_RCC_DMA1_FORCE_RESET

__HAL_RCC_DMA2_FORCE_RESET

__HAL_RCC_DMAMUX1_FORCE_RESET

__HAL_RCC_FLASH_FORCE_RESET

__HAL_RCC_CRC_FORCE_RESET

__HAL_RCC_TSC_FORCE_RESET

__HAL_RCC_DMA2D_FORCE_RESET

__HAL_RCC_GFXMMU_FORCE_RESET

__HAL_RCC_AHB1_RELEASE_RESET

__HAL_RCC_DMA1_RELEASE_RESET

__HAL_RCC_DMA2_RELEASE_RESET

__HAL_RCC_DMAMUX1_RELEASE_RESET

__HAL_RCC_FLASH_RELEASE_RESET

__HAL_RCC_CRC_RELEASE_RESET

__HAL_RCC_TSC_RELEASE_RESET

__HAL_RCC_DMA2D_RELEASE_RESET

__HAL_RCC_GFXMMU_RELEASE_RESET

AHB1 Peripheral Clock Enable Disable

__HAL_RCC_DMA1_CLK_ENABLE

__HAL_RCC_DMA2_CLK_ENABLE

__HAL_RCC_DMAMUX1_CLK_ENABLE

__HAL_RCC_FLASH_CLK_ENABLE

__HAL_RCC_CRC_CLK_ENABLE

__HAL_RCC_TSC_CLK_ENABLE

__HAL_RCC_DMA2D_CLK_ENABLE

__HAL_RCC_GFXMMU_CLK_ENABLE

__HAL_RCC_DMA1_CLK_DISABLE

__HAL_RCC_DMA2_CLK_DISABLE

__HAL_RCC_DMAMUX1_CLK_DISABLE

__HAL_RCC_FLASH_CLK_DISABLE

__HAL_RCC_CRC_CLK_DISABLE

__HAL_RCC_TSC_CLK_DISABLE

__HAL_RCC_DMA2D_CLK_DISABLE

__HAL_RCC_GFXMMU_CLK_DISABLE

AHB1 Peripheral Clock Enabled or Disabled Status

__HAL_RCC_DMA1_IS_CLK_ENABLED

__HAL_RCC_DMA2_IS_CLK_ENABLED

__HAL_RCC_DMAMUX1_IS_CLK_ENABLED

__HAL_RCC_FLASH_IS_CLK_ENABLED

__HAL_RCC_CRC_IS_CLK_ENABLED

__HAL_RCC_TSC_IS_CLK_ENABLED

__HAL_RCC_DMA2D_IS_CLK_ENABLED

__HAL_RCC_GFXMMU_IS_CLK_ENABLED

__HAL_RCC_DMA1_IS_CLK_DISABLED

__HAL_RCC_DMA2_IS_CLK_DISABLED

__HAL_RCC_DMAMUX1_IS_CLK_DISABLED

__HAL_RCC_FLASH_IS_CLK_DISABLED

__HAL_RCC_CRC_IS_CLK_DISABLED

__HAL_RCC_TSC_IS_CLK_DISABLED

__HAL_RCC_DMA2D_IS_CLK_DISABLED

__HAL_RCC_GFXMMU_IS_CLK_DISABLED

AHB2 Peripheral Clock Enabled or Disabled Status

__HAL_RCC_GPIOA_IS_CLK_ENABLED

__HAL_RCC_GPIOB_IS_CLK_ENABLED

__HAL_RCC_GPIOC_IS_CLK_ENABLED

__HAL_RCC_GPIOD_IS_CLK_ENABLED

__HAL_RCC_GPIOE_IS_CLK_ENABLED

__HAL_RCC_GPIOF_IS_CLK_ENABLED

__HAL_RCC_GPIOG_IS_CLK_ENABLED

__HAL_RCC_GPIOH_IS_CLK_ENABLED

__HAL_RCC_GPIOI_IS_CLK_ENABLED

__HAL_RCC_USB_OTG_FS_IS_CLK_ENABLED

__HAL_RCC_ADC_IS_CLK_ENABLED

__HAL_RCC_DCMI_IS_CLK_ENABLED

__HAL_RCC_AES_IS_CLK_ENABLED

__HAL_RCC_HASH_IS_CLK_ENABLED

__HAL_RCC_RNG_IS_CLK_ENABLED

__HAL_RCC_GPIOA_IS_CLK_DISABLED

__HAL_RCC_GPIOB_IS_CLK_DISABLED

__HAL_RCC_GPIOC_IS_CLK_DISABLED

__HAL_RCC_GPIOD_IS_CLK_DISABLED

__HAL_RCC_GPIOE_IS_CLK_DISABLED

__HAL_RCC_GPIOF_IS_CLK_DISABLED

__HAL_RCC_GPIOG_IS_CLK_DISABLED

__HAL_RCC_GPIOH_IS_CLK_DISABLED

__HAL_RCC_GPIOI_IS_CLK_DISABLED

__HAL_RCC_USB_OTG_FS_IS_CLK_DISABLED

__HAL_RCC_ADC_IS_CLK_DISABLED

__HAL_RCC_DCMI_IS_CLK_DISABLED

__HAL_RCC_AES_IS_CLK_DISABLED

__HAL_RCC_HASH_IS_CLK_DISABLED

__HAL_RCC_RNG_IS_CLK_DISABLED

AHB2 Peripheral Clock Sleep Enable Disable

__HAL_RCC_GPIOA_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOB_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOC_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOD_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOE_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOF_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOG_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOH_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOI_CLK_SLEEP_ENABLE

__HAL_RCC_SRAM2_CLK_SLEEP_ENABLE

__HAL_RCC_USB_OTG_FS_CLK_SLEEP_ENABLE

__HAL_RCC_ADC_CLK_SLEEP_ENABLE

__HAL_RCC_DCMI_CLK_SLEEP_ENABLE

__HAL_RCC_AES_CLK_SLEEP_ENABLE

__HAL_RCC_HASH_CLK_SLEEP_ENABLE

__HAL_RCC_RNG_CLK_SLEEP_ENABLE

__HAL_RCC_OSPIM_CLK_SLEEP_ENABLE

__HAL_RCC_SDMMC1_CLK_SLEEP_ENABLE

__HAL_RCC_GPIOA_CLK_SLEEP_DISABLE

__HAL_RCC_GPIOB_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOC_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOD_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOE_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOF_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOG_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOH_CLK_SLEEP_DISABLE
__HAL_RCC_GPIOI_CLK_SLEEP_DISABLE
__HAL_RCC_SRAM2_CLK_SLEEP_DISABLE
__HAL_RCC_USB_OTG_FS_CLK_SLEEP_DISABLE
__HAL_RCC_ADC_CLK_SLEEP_DISABLE
__HAL_RCC_DCMI_CLK_SLEEP_DISABLE
__HAL_RCC_AES_CLK_SLEEP_DISABLE
__HAL_RCC_HASH_CLK_SLEEP_DISABLE
__HAL_RCC_RNG_CLK_SLEEP_DISABLE
__HAL_RCC_OSPIM_CLK_SLEEP_DISABLE
__HAL_RCC_SDMMC1_CLK_SLEEP_DISABLE

AHB2 Peripheral Clock Sleep Enabled or Disabled Status

__HAL_RCC_GPIOA_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOB_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOC_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOD_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOE_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOF_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOG_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOH_IS_CLK_SLEEP_ENABLED
__HAL_RCC_GPIOI_IS_CLK_SLEEP_ENABLED
__HAL_RCC_SRAM2_IS_CLK_SLEEP_ENABLED
__HAL_RCC_USB_OTG_FS_IS_CLK_SLEEP_ENABLED
__HAL_RCC_ADC_IS_CLK_SLEEP_ENABLED
__HAL_RCC_DCMI_IS_CLK_SLEEP_ENABLED
__HAL_RCC_AES_IS_CLK_SLEEP_ENABLED
__HAL_RCC_HASH_IS_CLK_SLEEP_ENABLED
__HAL_RCC_RNG_IS_CLK_SLEEP_ENABLED
__HAL_RCC_OSPIM_IS_CLK_SLEEP_ENABLED
__HAL_RCC_SDMMC1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_GPIOA_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOB_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOC_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOD_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOE_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOF_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOG_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOH_IS_CLK_SLEEP_DISABLED
__HAL_RCC_GPIOI_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SRAM2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_USB_OTG_FS_IS_CLK_SLEEP_DISABLED
__HAL_RCC_ADC_IS_CLK_SLEEP_DISABLED
__HAL_RCC_DCMI_IS_CLK_SLEEP_DISABLED
__HAL_RCC_AES_IS_CLK_SLEEP_DISABLED
__HAL_RCC_HASH_IS_CLK_SLEEP_DISABLED
__HAL_RCC_RNG_IS_CLK_SLEEP_DISABLED
__HAL_RCC_OSPIM_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SDMMC1_IS_CLK_SLEEP_DISABLED

AHB2 Peripheral Force Release Reset

__HAL_RCC_AHB2_FORCE_RESET
__HAL_RCC_GPIOA_FORCE_RESET
__HAL_RCC_GPIOB_FORCE_RESET
__HAL_RCC_GPIOC_FORCE_RESET
__HAL_RCC_GPIOD_FORCE_RESET
__HAL_RCC_GPIOE_FORCE_RESET
__HAL_RCC_GPIOF_FORCE_RESET
__HAL_RCC_GPIOG_FORCE_RESET
__HAL_RCC_GPIOH_FORCE_RESET
__HAL_RCC_GPIOI_FORCE_RESET
__HAL_RCC_USB_OTG_FS_FORCE_RESET
__HAL_RCC_ADC_FORCE_RESET
__HAL_RCC_DCMI_FORCE_RESET
__HAL_RCC_AES_FORCE_RESET
__HAL_RCC_HASH_FORCE_RESET
__HAL_RCC_RNG_FORCE_RESET
__HAL_RCC_OSPIM_FORCE_RESET

__HAL_RCC_SDMMC1_FORCE_RESET
__HAL_RCC_AHB2_RELEASE_RESET
__HAL_RCC_GPIOA_RELEASE_RESET
__HAL_RCC_GPIOB_RELEASE_RESET
__HAL_RCC_GPIOC_RELEASE_RESET
__HAL_RCC_GPIOD_RELEASE_RESET
__HAL_RCC_GPIOE_RELEASE_RESET
__HAL_RCC_GPIOF_RELEASE_RESET
__HAL_RCC_GPIOG_RELEASE_RESET
__HAL_RCC_GPIOH_RELEASE_RESET
__HAL_RCC_GPIOI_RELEASE_RESET
__HAL_RCC_USB_OTG_FS_RELEASE_RESET
__HAL_RCC_ADC_RELEASE_RESET
__HAL_RCC_DCMI_RELEASE_RESET
__HAL_RCC_AES_RELEASE_RESET
__HAL_RCC_HASH_RELEASE_RESET
__HAL_RCC_RNG_RELEASE_RESET
__HAL_RCC_OSPIM_RELEASE_RESET
__HAL_RCC_SDMMC1_RELEASE_RESET

AHB2 Peripheral Clock Enable Disable

__HAL_RCC_GPIOA_CLK_ENABLE
__HAL_RCC_GPIOB_CLK_ENABLE
__HAL_RCC_GPIOC_CLK_ENABLE
__HAL_RCC_GPIOD_CLK_ENABLE
__HAL_RCC_GPIOE_CLK_ENABLE
__HAL_RCC_GPIOF_CLK_ENABLE
__HAL_RCC_GPIOG_CLK_ENABLE
__HAL_RCC_GPIOH_CLK_ENABLE
__HAL_RCC_GPIOI_CLK_ENABLE
__HAL_RCC_USB_OTG_FS_CLK_ENABLE
__HAL_RCC_ADC_CLK_ENABLE
__HAL_RCC_DCMI_CLK_ENABLE
__HAL_RCC_AES_CLK_ENABLE
__HAL_RCC_HASH_CLK_ENABLE
__HAL_RCC_RNG_CLK_ENABLE
__HAL_RCC_OSPIM_CLK_ENABLE

__HAL_RCC_SDMMC1_CLK_ENABLE

__HAL_RCC_GPIOA_CLK_DISABLE

__HAL_RCC_GPIOB_CLK_DISABLE

__HAL_RCC_GPIOC_CLK_DISABLE

__HAL_RCC_GPIOD_CLK_DISABLE

__HAL_RCC_GPIOE_CLK_DISABLE

__HAL_RCC_GPIOF_CLK_DISABLE

__HAL_RCC_GPIOG_CLK_DISABLE

__HAL_RCC_GPIOH_CLK_DISABLE

__HAL_RCC_GPIOI_CLK_DISABLE

__HAL_RCC_USB_OTG_FS_CLK_DISABLE

__HAL_RCC_ADC_CLK_DISABLE

__HAL_RCC_DCMI_CLK_DISABLE

__HAL_RCC_AES_CLK_DISABLE

__HAL_RCC_HASH_CLK_DISABLE

__HAL_RCC_RNG_CLK_DISABLE

__HAL_RCC_OSPIM_CLK_DISABLE

__HAL_RCC_SDMMC1_CLK_DISABLE

AHB3 Peripheral Clock Enable Disable

__HAL_RCC_FMC_CLK_ENABLE

__HAL_RCC_OSPI1_CLK_ENABLE

__HAL_RCC_OSPI2_CLK_ENABLE

__HAL_RCC_FMC_CLK_DISABLE

__HAL_RCC_OSPI1_CLK_DISABLE

__HAL_RCC_OSPI2_CLK_DISABLE

AHB3 Peripheral Clock Enabled or Disabled Status

__HAL_RCC_FMC_IS_CLK_ENABLED

__HAL_RCC_FMC_IS_CLK_DISABLED

AHB3 Peripheral Clock Sleep Enable Disable

__HAL_RCC_OSPI1_CLK_SLEEP_ENABLE

__HAL_RCC_OSPI2_CLK_SLEEP_ENABLE

__HAL_RCC_FMC_CLK_SLEEP_ENABLE

__HAL_RCC_OSPI1_CLK_SLEEP_DISABLE

__HAL_RCC_OSPI2_CLK_SLEEP_DISABLE

__HAL_RCC_FMC_CLK_SLEEP_DISABLE

AHB3 Peripheral Clock Sleep Enabled or Disabled Status

__HAL_RCC_OSPI1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_OSPI2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_FMC_IS_CLK_SLEEP_ENABLED

__HAL_RCC_OSPI1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_OSPI2_IS_CLK_SLEEP_DISABLED

__HAL_RCC_FMC_IS_CLK_SLEEP_DISABLED

AHB3 Peripheral Force Release Reset

__HAL_RCC_AHB3_FORCE_RESET

__HAL_RCC_FMC_FORCE_RESET

__HAL_RCC_OSPI1_FORCE_RESET

__HAL_RCC_OSPI2_FORCE_RESET

__HAL_RCC_AHB3_RELEASE_RESET

__HAL_RCC_FMC_RELEASE_RESET

__HAL_RCC_OSPI1_RELEASE_RESET

__HAL_RCC_OSPI2_RELEASE_RESET

AHB Clock Source

RCC_SYSCLK_DIV1	SYSCLK not divided
RCC_SYSCLK_DIV2	SYSCLK divided by 2
RCC_SYSCLK_DIV4	SYSCLK divided by 4
RCC_SYSCLK_DIV8	SYSCLK divided by 8
RCC_SYSCLK_DIV16	SYSCLK divided by 16
RCC_SYSCLK_DIV64	SYSCLK divided by 64
RCC_SYSCLK_DIV128	SYSCLK divided by 128
RCC_SYSCLK_DIV256	SYSCLK divided by 256
RCC_SYSCLK_DIV512	SYSCLK divided by 512

APB1 APB2 Clock Source

RCC_HCLK_DIV1	HCLK not divided
RCC_HCLK_DIV2	HCLK divided by 2
RCC_HCLK_DIV4	HCLK divided by 4
RCC_HCLK_DIV8	HCLK divided by 8
RCC_HCLK_DIV16	HCLK divided by 16

APB1 Peripheral Clock Enable Disable

__HAL_RCC_TIM2_CLK_ENABLE

__HAL_RCC_TIM3_CLK_ENABLE

__HAL_RCC_TIM4_CLK_ENABLE

__HAL_RCC_TIM5_CLK_ENABLE
__HAL_RCC_TIM6_CLK_ENABLE
__HAL_RCC_TIM7_CLK_ENABLE
__HAL_RCC_RTCAPB_CLK_ENABLE
__HAL_RCC_WWDG_CLK_ENABLE
__HAL_RCC_SPI2_CLK_ENABLE
__HAL_RCC_SPI3_CLK_ENABLE
__HAL_RCC_USART2_CLK_ENABLE
__HAL_RCC_USART3_CLK_ENABLE
__HAL_RCC_UART4_CLK_ENABLE
__HAL_RCC_UART5_CLK_ENABLE
__HAL_RCC_I2C1_CLK_ENABLE
__HAL_RCC_I2C2_CLK_ENABLE
__HAL_RCC_I2C3_CLK_ENABLE
__HAL_RCC_I2C4_CLK_ENABLE
__HAL_RCC_CRS_CLK_ENABLE
__HAL_RCC_CAN1_CLK_ENABLE
__HAL_RCC_PWR_CLK_ENABLE
__HAL_RCC_DAC1_CLK_ENABLE
__HAL_RCC_OPAMP_CLK_ENABLE
__HAL_RCC_LPTIM1_CLK_ENABLE
__HAL_RCC_LPUART1_CLK_ENABLE
__HAL_RCC_LPTIM2_CLK_ENABLE
__HAL_RCC_TIM2_CLK_DISABLE
__HAL_RCC_TIM3_CLK_DISABLE
__HAL_RCC_TIM4_CLK_DISABLE
__HAL_RCC_TIM5_CLK_DISABLE
__HAL_RCC_TIM6_CLK_DISABLE
__HAL_RCC_TIM7_CLK_DISABLE
__HAL_RCC_RTCAPB_CLK_DISABLE
__HAL_RCC_SPI2_CLK_DISABLE
__HAL_RCC_SPI3_CLK_DISABLE
__HAL_RCC_USART2_CLK_DISABLE
__HAL_RCC_USART3_CLK_DISABLE
__HAL_RCC_UART4_CLK_DISABLE
__HAL_RCC_UART5_CLK_DISABLE

__HAL_RCC_I2C1_CLK_DISABLE
__HAL_RCC_I2C2_CLK_DISABLE
__HAL_RCC_I2C3_CLK_DISABLE
__HAL_RCC_I2C4_CLK_DISABLE
__HAL_RCC_CRS_CLK_DISABLE
__HAL_RCC_CAN1_CLK_DISABLE
__HAL_RCC_PWR_CLK_DISABLE
__HAL_RCC_DAC1_CLK_DISABLE
__HAL_RCC_OPAMP_CLK_DISABLE
__HAL_RCC_LPTIM1_CLK_DISABLE
__HAL_RCC_LPUART1_CLK_DISABLE
__HAL_RCC_LPTIM2_CLK_DISABLE

APB1 Peripheral Clock Enabled or Disabled Status

__HAL_RCC_TIM2_IS_CLK_ENABLED
__HAL_RCC_TIM3_IS_CLK_ENABLED
__HAL_RCC_TIM4_IS_CLK_ENABLED
__HAL_RCC_TIM5_IS_CLK_ENABLED
__HAL_RCC_TIM6_IS_CLK_ENABLED
__HAL_RCC_TIM7_IS_CLK_ENABLED
__HAL_RCC_RTCAPB_IS_CLK_ENABLED
__HAL_RCC_WWDG_IS_CLK_ENABLED
__HAL_RCC_SPI2_IS_CLK_ENABLED
__HAL_RCC_SPI3_IS_CLK_ENABLED
__HAL_RCC_USART2_IS_CLK_ENABLED
__HAL_RCC_USART3_IS_CLK_ENABLED
__HAL_RCC_UART4_IS_CLK_ENABLED
__HAL_RCC_UART5_IS_CLK_ENABLED
__HAL_RCC_I2C1_IS_CLK_ENABLED
__HAL_RCC_I2C2_IS_CLK_ENABLED
__HAL_RCC_I2C3_IS_CLK_ENABLED
__HAL_RCC_I2C4_IS_CLK_ENABLED
__HAL_RCC_CRS_IS_CLK_ENABLED
__HAL_RCC_CAN1_IS_CLK_ENABLED
__HAL_RCC_PWR_IS_CLK_ENABLED
__HAL_RCC_DAC1_IS_CLK_ENABLED
__HAL_RCC_OPAMP_IS_CLK_ENABLED

__HAL_RCC_LPTIM1_IS_CLK_ENABLED
__HAL_RCC_LPUART1_IS_CLK_ENABLED
__HAL_RCC_LPTIM2_IS_CLK_ENABLED
__HAL_RCC_TIM2_IS_CLK_DISABLED
__HAL_RCC_TIM3_IS_CLK_DISABLED
__HAL_RCC_TIM4_IS_CLK_DISABLED
__HAL_RCC_TIM5_IS_CLK_DISABLED
__HAL_RCC_TIM6_IS_CLK_DISABLED
__HAL_RCC_TIM7_IS_CLK_DISABLED
__HAL_RCC_RTCAPB_IS_CLK_DISABLED
__HAL_RCC_WWDG_IS_CLK_DISABLED
__HAL_RCC_SPI2_IS_CLK_DISABLED
__HAL_RCC_SPI3_IS_CLK_DISABLED
__HAL_RCC_USART2_IS_CLK_DISABLED
__HAL_RCC_USART3_IS_CLK_DISABLED
__HAL_RCC_UART4_IS_CLK_DISABLED
__HAL_RCC_UART5_IS_CLK_DISABLED
__HAL_RCC_I2C1_IS_CLK_DISABLED
__HAL_RCC_I2C2_IS_CLK_DISABLED
__HAL_RCC_I2C3_IS_CLK_DISABLED
__HAL_RCC_I2C4_IS_CLK_DISABLED
__HAL_RCC_CRS_IS_CLK_DISABLED
__HAL_RCC_CAN1_IS_CLK_DISABLED
__HAL_RCC_PWR_IS_CLK_DISABLED
__HAL_RCC_DAC1_IS_CLK_DISABLED
__HAL_RCC_OPAMP_IS_CLK_DISABLED
__HAL_RCC_LPTIM1_IS_CLK_DISABLED
__HAL_RCC_LPUART1_IS_CLK_DISABLED
__HAL_RCC_LPTIM2_IS_CLK_DISABLED

APB1 Peripheral Clock Sleep Enable Disable

__HAL_RCC_TIM2_CLK_SLEEP_ENABLE
__HAL_RCC_TIM3_CLK_SLEEP_ENABLE
__HAL_RCC_TIM4_CLK_SLEEP_ENABLE
__HAL_RCC_TIM5_CLK_SLEEP_ENABLE
__HAL_RCC_TIM6_CLK_SLEEP_ENABLE
__HAL_RCC_TIM7_CLK_SLEEP_ENABLE

__HAL_RCC_RTCAPB_CLK_SLEEP_ENABLE
__HAL_RCC_WWDG_CLK_SLEEP_ENABLE
__HAL_RCC_SPI2_CLK_SLEEP_ENABLE
__HAL_RCC_SPI3_CLK_SLEEP_ENABLE
__HAL_RCC_USART2_CLK_SLEEP_ENABLE
__HAL_RCC_USART3_CLK_SLEEP_ENABLE
__HAL_RCC_UART4_CLK_SLEEP_ENABLE
__HAL_RCC_UART5_CLK_SLEEP_ENABLE
__HAL_RCC_I2C1_CLK_SLEEP_ENABLE
__HAL_RCC_I2C2_CLK_SLEEP_ENABLE
__HAL_RCC_I2C3_CLK_SLEEP_ENABLE
__HAL_RCC_I2C4_CLK_SLEEP_ENABLE
__HAL_RCC_CRS_CLK_SLEEP_ENABLE
__HAL_RCC_CAN1_CLK_SLEEP_ENABLE
__HAL_RCC_PWR_CLK_SLEEP_ENABLE
__HAL_RCC_DAC1_CLK_SLEEP_ENABLE
__HAL_RCC_OPAMP_CLK_SLEEP_ENABLE
__HAL_RCC_LPTIM1_CLK_SLEEP_ENABLE
__HAL_RCC_LPUART1_CLK_SLEEP_ENABLE
__HAL_RCC_LPTIM2_CLK_SLEEP_ENABLE
__HAL_RCC_TIM2_CLK_SLEEP_DISABLE
__HAL_RCC_TIM3_CLK_SLEEP_DISABLE
__HAL_RCC_TIM4_CLK_SLEEP_DISABLE
__HAL_RCC_TIM5_CLK_SLEEP_DISABLE
__HAL_RCC_TIM6_CLK_SLEEP_DISABLE
__HAL_RCC_TIM7_CLK_SLEEP_DISABLE
__HAL_RCC_RTCAPB_CLK_SLEEP_DISABLE
__HAL_RCC_WWDG_CLK_SLEEP_DISABLE
__HAL_RCC_SPI2_CLK_SLEEP_DISABLE
__HAL_RCC_SPI3_CLK_SLEEP_DISABLE
__HAL_RCC_USART2_CLK_SLEEP_DISABLE
__HAL_RCC_USART3_CLK_SLEEP_DISABLE
__HAL_RCC_UART4_CLK_SLEEP_DISABLE
__HAL_RCC_UART5_CLK_SLEEP_DISABLE
__HAL_RCC_I2C1_CLK_SLEEP_DISABLE
__HAL_RCC_I2C2_CLK_SLEEP_DISABLE

__HAL_RCC_I2C3_CLK_SLEEP_DISABLE

__HAL_RCC_I2C4_CLK_SLEEP_DISABLE

__HAL_RCC_CRS_CLK_SLEEP_DISABLE

__HAL_RCC_CAN1_CLK_SLEEP_DISABLE

__HAL_RCC_PWR_CLK_SLEEP_DISABLE

__HAL_RCC_DAC1_CLK_SLEEP_DISABLE

__HAL_RCC_OPAMP_CLK_SLEEP_DISABLE

__HAL_RCC_LPTIM1_CLK_SLEEP_DISABLE

__HAL_RCC_LPUART1_CLK_SLEEP_DISABLE

__HAL_RCC_LPTIM2_CLK_SLEEP_DISABLE

APB1 Peripheral Clock Sleep Enabled or Disabled Status

__HAL_RCC_TIM2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM3_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM4_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM5_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM6_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM7_IS_CLK_SLEEP_ENABLED

__HAL_RCC_RTCAPB_IS_CLK_SLEEP_ENABLED

__HAL_RCC_WWDG_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SPI2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SPI3_IS_CLK_SLEEP_ENABLED

__HAL_RCC_USART2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_USART3_IS_CLK_SLEEP_ENABLED

__HAL_RCC_UART4_IS_CLK_SLEEP_ENABLED

__HAL_RCC_UART5_IS_CLK_SLEEP_ENABLED

__HAL_RCC_I2C1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_I2C2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_I2C3_IS_CLK_SLEEP_ENABLED

__HAL_RCC_I2C4_IS_CLK_SLEEP_ENABLED

__HAL_RCC_CRS_IS_CLK_SLEEP_ENABLED

__HAL_RCC_CAN1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_PWR_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DAC1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_OPAMP_IS_CLK_SLEEP_ENABLED

__HAL_RCC_LPTIM1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_LPUART1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_LPTIM2_IS_CLK_SLEEP_ENABLED
__HAL_RCC_TIM2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM3_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM4_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM5_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM6_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM7_IS_CLK_SLEEP_DISABLED
__HAL_RCC_RTCAPB_IS_CLK_SLEEP_DISABLED
__HAL_RCC_WWDG_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SPI2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SPI3_IS_CLK_SLEEP_DISABLED
__HAL_RCC_USART2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_USART3_IS_CLK_SLEEP_DISABLED
__HAL_RCC_UART4_IS_CLK_SLEEP_DISABLED
__HAL_RCC_UART5_IS_CLK_SLEEP_DISABLED
__HAL_RCC_I2C1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_I2C2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_I2C3_IS_CLK_SLEEP_DISABLED
__HAL_RCC_I2C4_IS_CLK_SLEEP_DISABLED
__HAL_RCC_CRS_IS_CLK_SLEEP_DISABLED
__HAL_RCC_CAN1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_PWR_IS_CLK_SLEEP_DISABLED
__HAL_RCC_DAC1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_OPAMP_IS_CLK_SLEEP_DISABLED
__HAL_RCC_LPTIM1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_LPUART1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_LPTIM2_IS_CLK_SLEEP_DISABLED

APB1 Peripheral Force Release Reset

__HAL_RCC_APB1_FORCE_RESET
__HAL_RCC_TIM2_FORCE_RESET
__HAL_RCC_TIM3_FORCE_RESET
__HAL_RCC_TIM4_FORCE_RESET
__HAL_RCC_TIM5_FORCE_RESET
__HAL_RCC_TIM6_FORCE_RESET
__HAL_RCC_TIM7_FORCE_RESET
__HAL_RCC_SPI2_FORCE_RESET

__HAL_RCC_SPI3_FORCE_RESET
__HAL_RCC_USART2_FORCE_RESET
__HAL_RCC_USART3_FORCE_RESET
__HAL_RCC_UART4_FORCE_RESET
__HAL_RCC_UART5_FORCE_RESET
__HAL_RCC_I2C1_FORCE_RESET
__HAL_RCC_I2C2_FORCE_RESET
__HAL_RCC_I2C3_FORCE_RESET
__HAL_RCC_I2C4_FORCE_RESET
__HAL_RCC_CRS_FORCE_RESET
__HAL_RCC_CAN1_FORCE_RESET
__HAL_RCC_PWR_FORCE_RESET
__HAL_RCC_DAC1_FORCE_RESET
__HAL_RCC_OPAMP_FORCE_RESET
__HAL_RCC_LPTIM1_FORCE_RESET
__HAL_RCC_LPUART1_FORCE_RESET
__HAL_RCC_LPTIM2_FORCE_RESET
__HAL_RCC_APB1_RELEASE_RESET
__HAL_RCC_TIM2_RELEASE_RESET
__HAL_RCC_TIM3_RELEASE_RESET
__HAL_RCC_TIM4_RELEASE_RESET
__HAL_RCC_TIM5_RELEASE_RESET
__HAL_RCC_TIM6_RELEASE_RESET
__HAL_RCC_TIM7_RELEASE_RESET
__HAL_RCC_SPI2_RELEASE_RESET
__HAL_RCC_SPI3_RELEASE_RESET
__HAL_RCC_USART2_RELEASE_RESET
__HAL_RCC_USART3_RELEASE_RESET
__HAL_RCC_UART4_RELEASE_RESET
__HAL_RCC_UART5_RELEASE_RESET
__HAL_RCC_I2C1_RELEASE_RESET
__HAL_RCC_I2C2_RELEASE_RESET
__HAL_RCC_I2C3_RELEASE_RESET
__HAL_RCC_I2C4_RELEASE_RESET
__HAL_RCC_CRS_RELEASE_RESET
__HAL_RCC_CAN1_RELEASE_RESET

__HAL_RCC_PWR_RELEASE_RESET

__HAL_RCC_DAC1_RELEASE_RESET

__HAL_RCC_OPAMP_RELEASE_RESET

__HAL_RCC_LPTIM1_RELEASE_RESET

__HAL_RCC_LPUART1_RELEASE_RESET

__HAL_RCC_LPTIM2_RELEASE_RESET

APB2 Peripheral Clock Enable Disable

__HAL_RCC_SYSCFG_CLK_ENABLE

__HAL_RCC_FIREWALL_CLK_ENABLE

__HAL_RCC_TIM1_CLK_ENABLE

__HAL_RCC_SPI1_CLK_ENABLE

__HAL_RCC_TIM8_CLK_ENABLE

__HAL_RCC_USART1_CLK_ENABLE

__HAL_RCC_TIM15_CLK_ENABLE

__HAL_RCC_TIM16_CLK_ENABLE

__HAL_RCC_TIM17_CLK_ENABLE

__HAL_RCC_SAI1_CLK_ENABLE

__HAL_RCC_SAI2_CLK_ENABLE

__HAL_RCC_DFSDM1_CLK_ENABLE

__HAL_RCC_LTDC_CLK_ENABLE

__HAL_RCC_DSI_CLK_ENABLE

__HAL_RCC_SYSCFG_CLK_DISABLE

__HAL_RCC_TIM1_CLK_DISABLE

__HAL_RCC_SPI1_CLK_DISABLE

__HAL_RCC_TIM8_CLK_DISABLE

__HAL_RCC_USART1_CLK_DISABLE

__HAL_RCC_TIM15_CLK_DISABLE

__HAL_RCC_TIM16_CLK_DISABLE

__HAL_RCC_TIM17_CLK_DISABLE

__HAL_RCC_SAI1_CLK_DISABLE

__HAL_RCC_SAI2_CLK_DISABLE

__HAL_RCC_DFSDM1_CLK_DISABLE

__HAL_RCC_LTDC_CLK_DISABLE

__HAL_RCC_DSI_CLK_DISABLE

APB2 Peripheral Clock Enabled or Disabled Status

__HAL_RCC_SYSCFG_IS_CLK_ENABLED

__HAL_RCC_FIREWALL_IS_CLK_ENABLED
__HAL_RCC_TIM1_IS_CLK_ENABLED
__HAL_RCC_SPI1_IS_CLK_ENABLED
__HAL_RCC_TIM8_IS_CLK_ENABLED
__HAL_RCC_USART1_IS_CLK_ENABLED
__HAL_RCC_TIM15_IS_CLK_ENABLED
__HAL_RCC_TIM16_IS_CLK_ENABLED
__HAL_RCC_TIM17_IS_CLK_ENABLED
__HAL_RCC_SAI1_IS_CLK_ENABLED
__HAL_RCC_SAI2_IS_CLK_ENABLED
__HAL_RCC_DFSDM1_IS_CLK_ENABLED
__HAL_RCC_LTDC_IS_CLK_ENABLED
__HAL_RCC_DSI_IS_CLK_ENABLED
__HAL_RCC_SYSCFG_IS_CLK_DISABLED
__HAL_RCC_TIM1_IS_CLK_DISABLED
__HAL_RCC_SPI1_IS_CLK_DISABLED
__HAL_RCC_TIM8_IS_CLK_DISABLED
__HAL_RCC_USART1_IS_CLK_DISABLED
__HAL_RCC_TIM15_IS_CLK_DISABLED
__HAL_RCC_TIM16_IS_CLK_DISABLED
__HAL_RCC_TIM17_IS_CLK_DISABLED
__HAL_RCC_SAI1_IS_CLK_DISABLED
__HAL_RCC_SAI2_IS_CLK_DISABLED
__HAL_RCC_DFSDM1_IS_CLK_DISABLED
__HAL_RCC_LTDC_IS_CLK_DISABLED
__HAL_RCC_DSI_IS_CLK_DISABLED

APB2 Peripheral Clock Sleep Enable Disable

__HAL_RCC_SYSCFG_CLK_SLEEP_ENABLE
__HAL_RCC_TIM1_CLK_SLEEP_ENABLE
__HAL_RCC_SPI1_CLK_SLEEP_ENABLE
__HAL_RCC_TIM8_CLK_SLEEP_ENABLE
__HAL_RCC_USART1_CLK_SLEEP_ENABLE
__HAL_RCC_TIM15_CLK_SLEEP_ENABLE
__HAL_RCC_TIM16_CLK_SLEEP_ENABLE
__HAL_RCC_TIM17_CLK_SLEEP_ENABLE
__HAL_RCC_SAI1_CLK_SLEEP_ENABLE

__HAL_RCC_SAI2_CLK_SLEEP_ENABLE

__HAL_RCC_DFSDM1_CLK_SLEEP_ENABLE

__HAL_RCC_LTDC_CLK_SLEEP_ENABLE

__HAL_RCC_DSI_CLK_SLEEP_ENABLE

__HAL_RCC_SYSCFG_CLK_SLEEP_DISABLE

__HAL_RCC_TIM1_CLK_SLEEP_DISABLE

__HAL_RCC_SPI1_CLK_SLEEP_DISABLE

__HAL_RCC_TIM8_CLK_SLEEP_DISABLE

__HAL_RCC_USART1_CLK_SLEEP_DISABLE

__HAL_RCC_TIM15_CLK_SLEEP_DISABLE

__HAL_RCC_TIM16_CLK_SLEEP_DISABLE

__HAL_RCC_TIM17_CLK_SLEEP_DISABLE

__HAL_RCC_SAI1_CLK_SLEEP_DISABLE

__HAL_RCC_SAI2_CLK_SLEEP_DISABLE

__HAL_RCC_DFSDM1_CLK_SLEEP_DISABLE

__HAL_RCC_LTDC_CLK_SLEEP_DISABLE

__HAL_RCC_DSI_CLK_SLEEP_DISABLE

APB2 Peripheral Clock Sleep Enabled or Disabled Status

__HAL_RCC_SYSCFG_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SPI1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM8_IS_CLK_SLEEP_ENABLED

__HAL_RCC_USART1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM15_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM16_IS_CLK_SLEEP_ENABLED

__HAL_RCC_TIM17_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SAI1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SAI2_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DFSDM1_IS_CLK_SLEEP_ENABLED

__HAL_RCC_LTDC_IS_CLK_SLEEP_ENABLED

__HAL_RCC_DSI_IS_CLK_SLEEP_ENABLED

__HAL_RCC_SYSCFG_IS_CLK_SLEEP_DISABLED

__HAL_RCC_TIM1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_SPI1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_TIM8_IS_CLK_SLEEP_DISABLED

__HAL_RCC_USART1_IS_CLK_SLEEP_DISABLED

__HAL_RCC_TIM15_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM16_IS_CLK_SLEEP_DISABLED
__HAL_RCC_TIM17_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SAI1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_SAI2_IS_CLK_SLEEP_DISABLED
__HAL_RCC_DFSDM1_IS_CLK_SLEEP_DISABLED
__HAL_RCC_LTDC_IS_CLK_SLEEP_DISABLED
__HAL_RCC_DSI_IS_CLK_SLEEP_DISABLED

APB2 Peripheral Force Release Reset

__HAL_RCC_APB2_FORCE_RESET
__HAL_RCC_SYSCFG_FORCE_RESET
__HAL_RCC_TIM1_FORCE_RESET
__HAL_RCC_SPI1_FORCE_RESET
__HAL_RCC_TIM8_FORCE_RESET
__HAL_RCC_USART1_FORCE_RESET
__HAL_RCC_TIM15_FORCE_RESET
__HAL_RCC_TIM16_FORCE_RESET
__HAL_RCC_TIM17_FORCE_RESET
__HAL_RCC_SAI1_FORCE_RESET
__HAL_RCC_SAI2_FORCE_RESET
__HAL_RCC_DFSDM1_FORCE_RESET
__HAL_RCC_LTDC_FORCE_RESET
__HAL_RCC_DSI_FORCE_RESET
__HAL_RCC_APB2_RELEASE_RESET
__HAL_RCC_SYSCFG_RELEASE_RESET
__HAL_RCC_TIM1_RELEASE_RESET
__HAL_RCC_SPI1_RELEASE_RESET
__HAL_RCC_TIM8_RELEASE_RESET
__HAL_RCC_USART1_RELEASE_RESET
__HAL_RCC_TIM15_RELEASE_RESET
__HAL_RCC_TIM16_RELEASE_RESET
__HAL_RCC_TIM17_RELEASE_RESET
__HAL_RCC_SAI1_RELEASE_RESET
__HAL_RCC_SAI2_RELEASE_RESET
__HAL_RCC_DFSDM1_RELEASE_RESET
__HAL_RCC_LTDC_RELEASE_RESET

__HAL_RCC_DSI_RELEASE_RESET

RCC Backup Domain Reset

__HAL_RCC_BACKUPRESET_FORCE

Description:

- Macros to force or release the Backup domain reset.

Return value:

- None

Notes:

- This function resets the RTC peripheral (including the backup registers) and the RTC clock source selection in RCC_CSR register. The BKPSRAM is not affected by this reset.

__HAL_RCC_BACKUPRESET_RELEASE

RCC Exported Macros

__HAL_RCC_HSI_ENABLE

Description:

- Macros to enable or disable the Internal High Speed 16MHz oscillator (HSI).

Return value:

- None

Notes:

- The HSI is stopped by hardware when entering STOP and STANDBY modes. It is used (enabled by hardware) as system clock source after startup from Reset, wakeup from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled). HSI can not be stopped if it is used as system clock source. In this case, you have to select another source of the system clock then stop the HSI. After enabling the HSI, the application software should wait on HSIRDY flag to be set indicating that HSI clock is stable and can be used as system clock source. This parameter can be: ENABLE or DISABLE. When the HSI is stopped, HSIRDY flag goes low after 6 HSI oscillator clock cycles.

__HAL_RCC_HSI_DISABLE

__HAL_RCC_HSI_CALIBRATION VALUE_ADJUST

Description:

- Macro to adjust the Internal High Speed 16MHz oscillator (HSI) calibration value.

Parameters:

- __HSICALIBRATIONVALUE__: specifies the calibration trimming value (default is RCC_HSICALIBRATION_DEFAULT). This parameter must be a number between 0 and 0x1F (STM32L43x/STM32L44x/STM32L47x/STM32L48x) or 0x7F (for other devices).

Return value:

- None

Notes:

- The calibration is used to compensate for the variations in voltage and temperature that influence the frequency of the internal HSI RC.

__HAL_RCC_HSIAUTOMATIC_START_ENABLE

Description:

- Macros to enable or disable the wakeup the Internal High Speed oscillator (HSI) in parallel to the Internal Multi Speed oscillator (MSI) used at system wakeup.

Return value:

- None

Notes:

- The enable of this function has not effect on the HSION bit. This parameter can be: ENABLE or DISABLE.

__HAL_RCC_HSIAUTOMATIC_START_DISABLE

__HAL_RCC_HSISTOP_ENABLE

Description:

- Macros to enable or disable the force of the Internal High Speed oscillator (HSI) in STOP mode to be quickly available as kernel clock for USARTs and I2Cs.

Return value:

- None

Notes:

- Keeping the HSI ON in STOP mode allows to avoid slowing down the communication speed because of the HSI startup time. The enable of this function has not effect on the HSION bit. This parameter can be: ENABLE or DISABLE.

__HAL_RCC_HSISTOP_DISABLE

__HAL_RCC_MSI_ENABLE

Description:

- Macros to enable or disable the Internal Multi Speed oscillator (MSI).

Return value:

- None

Notes:

- The MSI is stopped by hardware when entering STOP and STANDBY modes. It is used (enabled by hardware) as system clock source after startup from Reset, wakeup from STOP and STANDBY mode, or in case of failure of the HSE used directly or indirectly as system clock (if the Clock Security System CSS is enabled). MSI can not be stopped if it is used as system clock source. In this case, you have to select another source of the system clock then stop the MSI. After enabling the MSI, the application software should wait on MSIRDY flag to be set indicating that MSI clock is stable and can be used as system clock source. When the MSI is stopped, MSIRDY flag goes low after 6 MSI oscillator clock cycles.

__HAL_RCC_MSI_DISABLE

__HAL_RCC_MSI_CALIBRATION VALUE_ADJUST

Description:

- Macro Adjusts the Internal Multi Speed oscillator (MSI) calibration value.

Parameters:

- __MSICALIBRATIONVALUE__: specifies the calibration trimming value (default is RCC_MSICALIBRATION_DEFAULT). This parameter must be a number between 0 and 255.

Return value:

- None

Notes:

- The calibration is used to compensate for the variations in voltage and temperature that influence the frequency of the internal MSI RC. Refer to the Application Note AN3300 for more details on how to calibrate the MSI.

__HAL_RCC_MSI_RANGE_CONFIG

Description:

- Macro configures the Internal Multi Speed oscillator (MSI) clock range in run mode.

Parameters:

- __MSIRANGEVALUE__: specifies the MSI clock range. This parameter must be one of the following values:
  - RCC_MSIRANGE_0 MSI clock is around 100 KHz
  - RCC_MSIRANGE_1 MSI clock is around 200 KHz
  - RCC_MSIRANGE_2 MSI clock is around 400 KHz
  - RCC_MSIRANGE_3 MSI clock is around 800 KHz
  - RCC_MSIRANGE_4 MSI clock is around 1 MHz
  - RCC_MSIRANGE_5 MSI clock is around 2 MHz
  - RCC_MSIRANGE_6 MSI clock is around 4 MHz (default after Reset)
  - RCC_MSIRANGE_7 MSI clock is around 8 MHz
  - RCC_MSIRANGE_8 MSI clock is around 16 MHz
  - RCC_MSIRANGE_9 MSI clock is around 24 MHz
  - RCC_MSIRANGE_10 MSI clock is around 32 MHz
  - RCC_MSIRANGE_11 MSI clock is around 48 MHz

Return value:

- None

Notes:

- After restart from Reset , the MSI clock is around 4 MHz. After stop the startup clock can be MSI (at any of its possible frequencies, the one that was used before entering stop mode) or HSI. After Standby its frequency can be selected between 4 possible values (1, 2, 4 or 8 MHz). MSIRANGE can be modified when MSI is OFF (MSION=0) or when MSI is ready (MSIRDY=1). The MSI clock range after reset can be modified on the fly.

__HAL_RCC_MSI_STANDBY_RANGE_CONFIG

Description:

- Macro configures the Internal Multi Speed oscillator (MSI) clock range after Standby mode After Standby its frequency can be selected between 4 possible values (1, 2, 4 or 8 MHz).

Parameters:

- __MSIRANGEVALUE__: specifies the MSI clock range. This parameter must be one

of the following values:
- RCC_MSIRANGE_4 MSI clock is around 1 MHz
- RCC_MSIRANGE_5 MSI clock is around 2 MHz
- RCC_MSIRANGE_6 MSI clock is around 4 MHz (default after Reset)
- RCC_MSIRANGE_7 MSI clock is around 8 MHz

Return value:

- None

__HAL_RCC_GET_MSI_RANGE

Description:

- Macro to get the Internal Multi Speed oscillator (MSI) clock range in run mode.

Return value:

- MSI: clock range. This parameter must be one of the following values:
  - RCC_MSIRANGE_0 MSI clock is around 100 KHz
  - RCC_MSIRANGE_1 MSI clock is around 200 KHz
  - RCC_MSIRANGE_2 MSI clock is around 400 KHz
  - RCC_MSIRANGE_3 MSI clock is around 800 KHz
  - RCC_MSIRANGE_4 MSI clock is around 1 MHz
  - RCC_MSIRANGE_5 MSI clock is around 2 MHz
  - RCC_MSIRANGE_6 MSI clock is around 4 MHz (default after Reset)
  - RCC_MSIRANGE_7 MSI clock is around 8 MHz
  - RCC_MSIRANGE_8 MSI clock is around 16 MHz
  - RCC_MSIRANGE_9 MSI clock is around 24 MHz
  - RCC_MSIRANGE_10 MSI clock is around 32 MHz
  - RCC_MSIRANGE_11 MSI clock is around 48 MHz

__HAL_RCC_LSI_ENABLE

Description:

- Macros to enable or disable the Internal Low Speed oscillator (LSI).

Return value:

- None

Notes:

- After enabling the LSI, the application

software should wait on LSIRDY flag to be set indicating that LSI clock is stable and can be used to clock the IWDG and/or the RTC. LSI can not be disabled if the IWDG is running. When the LSI is stopped, LSIRDY flag goes low after 6 LSI oscillator clock cycles.

__HAL_RCC_LSI_DISABLE

__HAL_RCC_HSE_CONFIG

Description:

- Macro to configure the External High Speed oscillator (HSE).

Parameters:

- __STATE__: specifies the new state of the HSE. This parameter can be one of the following values:
  - RCC_HSE_OFF Turn OFF the HSE oscillator, HSERDY flag goes low after 6 HSE oscillator clock cycles.
  - RCC_HSE_ON Turn ON the HSE oscillator.
  - RCC_HSE_BYPASS HSE oscillator bypassed with external clock.

Return value:

- None

Notes:

- Transition HSE Bypass to HSE On and HSE On to HSE Bypass are not supported by this macro. User should request a transition to HSE Off first and then HSE On or HSE Bypass. After enabling the HSE (RCC_HSE_ON or RCC_HSE_Bypass), the application software should wait on HSERDY flag to be set indicating that HSE clock is stable and can be used to clock the PLL and/or system clock. HSE state can not be changed if it is used directly or through the PLL as system clock. In this case, you have to select another source of the system clock then change the HSE state (ex. disable it). The HSE is stopped by hardware when entering STOP and STANDBY modes. This function reset the CSSON bit, so if the clock security system(CSS) was previously enabled you have to enable it again after calling this function.

__HAL_RCC_LSE_CONFIG

Description:

- Macro to configure the External Low

Speed oscillator (LSE).

Parameters:

- __STATE__: specifies the new state of the LSE. This parameter can be one of the following values:
  - RCC_LSE_OFF Turn OFF the LSE oscillator, LSERDY flag goes low after 6 LSE oscillator clock cycles.
  - RCC_LSE_ON Turn ON the LSE oscillator.
  - RCC_LSE_BYPASS LSE oscillator bypassed with external clock.

Return value:

- None

Notes:

- Transitions LSE Bypass to LSE On and LSE On to LSE Bypass are not supported by this macro. User should request a transition to LSE Off first and then LSE On or LSE Bypass. As the LSE is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using HAL_PWR_EnableBkUpAccess() function before to configure the LSE (to be done once after reset). After enabling the LSE (RCC_LSE_ON or RCC_LSE_BYPASS), the application software should wait on LSERDY flag to be set indicating that LSE clock is stable and can be used to clock the RTC.

__HAL_RCC_HSI48_ENABLE

Description:

- Macros to enable or disable the Internal High Speed 48MHz oscillator (HSI48).

Return value:

- None

Notes:

- The HSI48 is stopped by hardware when entering STOP and STANDBY modes. After enabling the HSI48, the application software should wait on HSI48RDY flag to be set indicating that HSI48 clock is stable. This parameter can be: ENABLE or DISABLE.

__HAL_RCC_HSI48_DISABLE

__HAL_RCC_RTC_CONFIG

Description:

- Macros to configure the RTC clock

(RTCCLK).

Parameters:

- __RTC_CLKSOURCE__: specifies the RTC clock source. This parameter can be one of the following values:
  - RCC_RTCCLKSOURCE_NO_CLK No clock selected as RTC clock.
  - RCC_RTCCLKSOURCE_LSE LSE selected as RTC clock.
  - RCC_RTCCLKSOURCE_LSI LSI selected as RTC clock.
  - RCC_RTCCLKSOURCE_HSE_DIV32 HSE clock divided by 32 selected

Return value:

- None

Notes:

- As the RTC clock configuration bits are in the Backup domain and write access is denied to this domain after reset, you have to enable write access using the Power Backup Access macro before to configure the RTC clock source (to be done once after reset). Once the RTC clock is configured it cannot be changed unless the Backup domain is reset using __HAL_RCC_BACKUPRESET_FORCE() macro, or by a Power On Reset (POR).
- If the LSE or LSI is used as RTC clock source, the RTC continues to work in STOP and STANDBY modes, and can be used as wakeup source. However, when the HSE clock is used as RTC clock source, the RTC cannot be used in STOP and STANDBY modes. The maximum input clock frequency for RTC is 1MHz (when using HSE as RTC clock source).

__HAL_RCC_GET_RTC_SOURCE

Description:

- Macro to get the RTC clock source.

Return value:

- The: returned value can be one of the following:
  - RCC_RTCCLKSOURCE_NO_CLK No clock selected as RTC clock.
  - RCC_RTCCLKSOURCE_LSE LSE selected as RTC clock.
  - RCC_RTCCLKSOURCE_LSI LSI selected as RTC clock.
  - RCC_RTCCLKSOURCE_HSE_DIV32 HSE clock divided by 32 selected

__HAL_RCC_PLL_ENABLE

Description:

- Macros to enable or disable the main PLL.

Return value:

- None

Notes:

- After enabling the main PLL, the application software should wait on PLLRDY flag to be set indicating that PLL clock is stable and can be used as system clock source. The main PLL can not be disabled if it is used as system clock source The main PLL is disabled by hardware when entering STOP and STANDBY modes.

__HAL_RCC_PLL_DISABLE

__HAL_RCC_PLL_PLLSOURCE_CONFIG

Description:

- Macro to configure the PLL clock source.

Parameters:

- __PLLSOURCE__: specifies the PLL entry clock source. This parameter can be one of the following values:
  - RCC_PLLSOURCE_NONE No clock selected as PLL clock entry
  - RCC_PLLSOURCE_MSI MSI oscillator clock selected as PLL clock entry
  - RCC_PLLSOURCE_HSI HSI oscillator clock selected as PLL clock entry
  - RCC_PLLSOURCE_HSE HSE oscillator clock selected as PLL clock entry

Return value:

- None

Notes:

- This function must be used only when the main PLL is disabled.
- This clock source is common for the main PLL and audio PLL (PLLSAI1 and PLLSAI2).

__HAL_RCC_PLL_PLLM_CONFIG

Description:

- Macro to configure the PLL source division factor M.

Parameters:

- __PLLM__: specifies the division factor for

PLL VCO input clock This parameter must be a number between Min_Data = 1 and Max_Data = 16 on STM32L4Rx/STM32L4Sx devices. This parameter must be a number between Min_Data = 1 and Max_Data = 8 on other devices.

Return value:

- None

Notes:

- This function must be used only when the main PLL is disabled.
- You have to set the PLLM parameter correctly to ensure that the VCO input frequency ranges from 4 to 16 MHz. It is recommended to select a frequency of 16 MHz to limit PLL jitter.

__HAL_RCC_PLL_CONFIG

Description:

- Macro to configure the main PLL clock source, multiplication and division factors.

Parameters:

- __PLLSOURCE__: specifies the PLL entry clock source. This parameter can be one of the following values:
  - RCC_PLLSOURCE_NONE No clock selected as PLL clock entry
  - RCC_PLLSOURCE_MSI MSI oscillator clock selected as PLL clock entry
  - RCC_PLLSOURCE_HSI HSI oscillator clock selected as PLL clock entry
  - RCC_PLLSOURCE_HSE HSE oscillator clock selected as PLL clock entry
- __PLLM__: specifies the division factor for PLL VCO input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16 on STM32L4Rx/STM32L4Sx devices. This parameter must be a number between Min_Data = 1 and Max_Data = 8 on other devices.
- __PLLN__: specifies the multiplication factor for PLL VCO output clock. This parameter must be a number between 8 and 86.
- __PLLP__: specifies the division factor for SAI clock. This parameter must be a number in the range (7 or 17) for STM32L47x/STM32L48x else (2 to 31).

- __PLLQ__: specifies the division factor for OTG FS, SDMMC1 and RNG clocks. This parameter must be in the range (2, 4, 6 or 8).
- __PLLR__: specifies the division factor for the main system clock.

Return value:

- None

Notes:

- This function must be used only when the main PLL is disabled.
- This clock source is common for the main PLL and audio PLL (PLLSAI1 and PLLSAI2).
- You have to set the PLLM parameter correctly to ensure that the VCO input frequency ranges from 4 to 16 MHz. It is recommended to select a frequency of 16 MHz to limit PLL jitter.
- You have to set the PLLN parameter correctly to ensure that the VCO output frequency is between 64 and 344 MHz.
- If the USB OTG FS is used in your application, you have to set the PLLQ parameter correctly to have 48 MHz clock for the USB. However, the SDMMC1 and RNG need a frequency lower than or equal to 48 MHz to work correctly.
- You have to set the PLLR parameter correctly to not exceed 80MHZ. This parameter must be in the range (2, 4, 6 or 8).

__HAL_RCC_GET_PLL_OSCSOURCE

Description:

- Macro to get the oscillator used as PLL clock source.

Return value:

- The: oscillator used as PLL clock source. The returned value can be one of the following:
  - RCC_PLLSOURCE_NONE: No oscillator is used as PLL clock source.
  - RCC_PLLSOURCE_MSI: MSI oscillator is used as PLL clock source.
  - RCC_PLLSOURCE_HSI: HSI oscillator is used as PLL clock source.
  - RCC_PLLSOURCE_HSE: HSE oscillator is used as PLL clock source.

__HAL_RCC_PLLCLKOUT_ENABLE

Description:

- Enable or disable each clock output

(RCC_PLL_SYSCLK,
RCC_PLL_48M1CLK,
RCC_PLL_SAI3CLK)

Parameters:

- __PLLCLOCKOUT__: specifies the PLL clock to be output. This parameter can be one or a combination of the following values:
  - RCC_PLL_SAI3CLK This clock is used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
  - RCC_PLL_48M1CLK This Clock is used to generate the clock for the USB OTG FS (48 MHz), the random analog generator (<=48 MHz) and the SDMMC1 (<= 48 MHz).
  - RCC_PLL_SYSCLK This Clock is used to generate the high speed system clock (up to 80MHz)

Return value:

- None

Notes:

- Enabling/disabling clock outputs RCC_PLL_SAI3CLK and RCC_PLL_48M1CLK can be done at anytime without the need to stop the PLL in order to save power. But RCC_PLL_SYSCLK cannot be stopped if used as System Clock.

__HAL_RCC_PLLCLKOUT_DISABLE

__HAL_RCC_GET_PLLCLKOUT_CONFIG

Description:

- Get clock output enable status (RCC_PLL_SYSCLK, RCC_PLL_48M1CLK, RCC_PLL_SAI3CLK)

Parameters:

- __PLLCLOCKOUT__: specifies the output PLL clock to be checked. This parameter can be one of the following values:
  - RCC_PLL_SAI3CLK This clock is used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
  - RCC_PLL_48M1CLK This Clock is used to generate the clock for the USB OTG FS (48 MHz), the random analog generator (<=48 MHz) and the SDMMC1 (<= 48 MHz).

– RCC_PLL_SYSCLK This Clock is used to generate the high speed system clock (up to 80MHz)

Return value:

- SET: / RESET

__HAL_RCC_SYSCLK_CONFIG

Description:

- Macro to configure the system clock source.

Parameters:

- __SYSCLKSOURCE__: specifies the system clock source. This parameter can be one of the following values:
  – RCC_SYSCLKSOURCE_MSI: MSI oscillator is used as system clock source.
  – RCC_SYSCLKSOURCE_HSI: HSI oscillator is used as system clock source.
  – RCC_SYSCLKSOURCE_HSE: HSE oscillator is used as system clock source.
  – RCC_SYSCLKSOURCE_PLLCLK: PLL output is used as system clock source.

Return value:

- None

__HAL_RCC_GET_SYSCLK_SOURCE

Description:

- Macro to get the clock source used as system clock.

Return value:

- The: clock source used as system clock. The returned value can be one of the following:
  – RCC_SYSCLKSOURCE_STATUS_MSI: MSI used as system clock.
  – RCC_SYSCLKSOURCE_STATUS_HSI: HSI used as system clock.
  – RCC_SYSCLKSOURCE_STATUS_HSE: HSE used as system clock.
  – RCC_SYSCLKSOURCE_STATUS_PLLCLK: PLL used as system clock.

__HAL_RCC_LSEDRIVE_CONFIG

Description:

- Macro to configure the External Low Speed oscillator (LSE) drive capability.

Parameters:

- __LSEDRIVE__: specifies the new state of

the LSE drive capability. This parameter can be one of the following values:
- RCC_LSEDRIVE_LOW LSE oscillator low drive capability.
- RCC_LSEDRIVE_MEDIUMLOW LSE oscillator medium low drive capability.
- RCC_LSEDRIVE_MEDIUMHIGH LSE oscillator medium high drive capability.
- RCC_LSEDRIVE_HIGH LSE oscillator high drive capability.

Return value:

- None

Notes:

- As the LSE is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using HAL_PWR_EnableBkUpAccess() function before to configure the LSE (to be done once after reset).

__HAL_RCC_WAKEUPSTOP_CLK_CONFIG

Description:

- Macro to configure the wake up from stop clock.

Parameters:

- __STOPWUCLK__: specifies the clock source used after wake up from stop. This parameter can be one of the following values:
  - RCC_STOP_WAKEUPCLOCK_MSI MSI selected as system clock source
  - RCC_STOP_WAKEUPCLOCK_HSI HSI selected as system clock source

Return value:

- None

__HAL_RCC_MCO1_CONFIG

Description:

- Macro to configure the MCO clock.

Parameters:

- __MCOCLKSOURCE__: specifies the MCO clock source. This parameter can be one of the following values:
  - RCC_MCO1SOURCE_NOCLOCK MCO output disabled
  - RCC_MCO1SOURCE_SYSCLK System clock selected as MCO source
  - RCC_MCO1SOURCE_MSI MSI clock selected as MCO source

  - RCC_MCO1SOURCE_HSI HSI clock selected as MCO source
  - RCC_MCO1SOURCE_HSE HSE clock selected as MCO sourcee
  - RCC_MCO1SOURCE_PLLCLK Main PLL clock selected as MCO source
  - RCC_MCO1SOURCE_LSI LSI clock selected as MCO source
  - RCC_MCO1SOURCE_LSE LSE clock selected as MCO source
- __MCODIV__: specifies the MCO clock prescaler. This parameter can be one of the following values:
  - RCC_MCODIV_1 MCO clock source is divided by 1
  - RCC_MCODIV_2 MCO clock source is divided by 2
  - RCC_MCODIV_4 MCO clock source is divided by 4
  - RCC_MCODIV_8 MCO clock source is divided by 8
  - RCC_MCODIV_16 MCO clock source is divided by 16

Flags

RCC_FLAG_MSIRDY	MSI Ready flag
RCC_FLAG_HSIRDY	HSI Ready flag
RCC_FLAG_HSERDY	HSE Ready flag
RCC_FLAG_PLLRDY	PLL Ready flag
RCC_FLAG_PLLSAI1RDY	PLLSAI1 Ready flag
RCC_FLAG_PLLSAI2RDY	PLLSAI2 Ready flag
RCC_FLAG_LSERDY	LSE Ready flag
RCC_FLAG_LSECSSD	LSE Clock Security System Interrupt flag
RCC_FLAG_LSIRDY	LSI Ready flag
RCC_FLAG_RMVF	Remove reset flag
RCC_FLAG_FWRST	Firewall reset flag
RCC_FLAG_OBLRST	Option Byte Loader reset flag
RCC_FLAG_PINRST	PIN reset flag
RCC_FLAG_BORRST	BOR reset flag
RCC_FLAG_SFTRST	Software Reset flag
RCC_FLAG_IWDGRST	Independent Watchdog reset flag
RCC_FLAG_WWDGRST	Window watchdog reset flag
RCC_FLAG_LPWRRST	Low-Power reset flag
RCC_FLAG_HSI48RDY	HSI48 Ready flag

Flags Interrupts Management

__HAL_RCC_ENABLE_IT

Description:

- Enable RCC interrupt (Perform Byte access to RCC_CIR[14:8] bits to enable the selected interrupts).

Parameters:

- __INTERRUPT__: specifies the RCC interrupt sources to be enabled. This parameter can be any combination of the following values:
  - RCC_IT_LSIRDY LSI ready interrupt
  - RCC_IT_LSERDY LSE ready interrupt
  - RCC_IT_MSIRDY HSI ready interrupt
  - RCC_IT_HSIRDY HSI ready interrupt
  - RCC_IT_HSERDY HSE ready interrupt
  - RCC_IT_PLLRDY Main PLL ready interrupt
  - RCC_IT_PLLSAI1RDY PLLSAI1 ready interrupt
  - RCC_IT_PLLSAI2RDY PLLSAI2 ready interrupt for devices with PLLSAI2
  - RCC_IT_LSECSS LSE Clock security system interrupt

Return value:

- None

__HAL_RCC_DISABLE_IT

Description:

- Disable RCC interrupt (Perform Byte access to RCC_CIR[14:8] bits to disable the selected interrupts).

Parameters:

- __INTERRUPT__: specifies the RCC interrupt sources to be disabled. This parameter can be any combination of the following values:
  - RCC_IT_LSIRDY LSI ready interrupt
  - RCC_IT_LSERDY LSE ready interrupt
  - RCC_IT_MSIRDY HSI ready interrupt
  - RCC_IT_HSIRDY HSI ready interrupt
  - RCC_IT_HSERDY HSE ready interrupt
  - RCC_IT_PLLRDY Main PLL ready interrupt
  - RCC_IT_PLLSAI1RDY PLLSAI1 ready interrupt
  - RCC_IT_PLLSAI2RDY PLLSAI2 ready interrupt for devices with PLLSAI2
  - RCC_IT_LSECSS LSE Clock security system interrupt

Return value:

- None

__HAL_RCC_CLEAR_IT

Description:

- Clear the RCC's interrupt pending bits (Perform Byte access to RCC_CIR[23:16] bits to clear the selected interrupt pending bits.

Parameters:

- __INTERRUPT__: specifies the interrupt pending bit to clear. This parameter can be any combination of the following values:
  - RCC_IT_LSIRDY LSI ready interrupt
  - RCC_IT_LSERDY LSE ready interrupt
  - RCC_IT_MSIRDY MSI ready interrupt
  - RCC_IT_HSIRDY HSI ready interrupt
  - RCC_IT_HSERDY HSE ready interrupt
  - RCC_IT_PLLRDY Main PLL ready interrupt
  - RCC_IT_PLLSAI1RDY PLLSAI1 ready interrupt
  - RCC_IT_PLLSAI2RDY PLLSAI2 ready interrupt for devices with PLLSAI2
  - RCC_IT_CSS HSE Clock security system interrupt
  - RCC_IT_LSECSS LSE Clock security system interrupt

Return value:

- None

__HAL_RCC_GET_IT

Description:

- Check whether the RCC interrupt has occurred or not.

Parameters:

- __INTERRUPT__: specifies the RCC interrupt source to check. This parameter can be one of the following values:
  - RCC_IT_LSIRDY LSI ready interrupt
  - RCC_IT_LSERDY LSE ready interrupt
  - RCC_IT_MSIRDY MSI ready interrupt
  - RCC_IT_HSIRDY HSI ready interrupt
  - RCC_IT_HSERDY HSE ready interrupt
  - RCC_IT_PLLRDY Main PLL ready interrupt
  - RCC_IT_PLLSAI1RDY PLLSAI1 ready interrupt
  - RCC_IT_PLLSAI2RDY PLLSAI2 ready interrupt for devices with PLLSAI2
  - RCC_IT_CSS HSE Clock security system interrupt
  - RCC_IT_LSECSS LSE Clock security

system interrupt

Return value:

- The: new state of __INTERRUPT__ (TRUE or FALSE).

__HAL_RCC_CLEAR_RESET_FLAGS

Description:

- Set RMVF bit to clear the reset flags.

Return value:

- None

__HAL_RCC_GET_FLAG

Description:

- Check whether the selected RCC flag is set or not.

Parameters:

- __FLAG__: specifies the flag to check. This parameter can be one of the following values:
  - RCC_FLAG_MSIRDY MSI oscillator clock ready
  - RCC_FLAG_HSIRDY HSI oscillator clock ready
  - RCC_FLAG_HSERDY HSE oscillator clock ready
  - RCC_FLAG_PLLRDY Main PLL clock ready
  - RCC_FLAG_PLLSAI1RDY PLLSAI1 clock ready
  - RCC_FLAG_PLLSAI2RDY PLLSAI2 clock ready for devices with PLLSAI2
  - RCC_FLAG_LSERDY LSE oscillator clock ready
  - RCC_FLAG_LSECSSD Clock security system failure on LSE oscillator detection
  - RCC_FLAG_LSIRDY LSI oscillator clock ready
  - RCC_FLAG_BORRST BOR reset
  - RCC_FLAG_OBLRST OBLRST reset
  - RCC_FLAG_PINRST Pin reset
  - RCC_FLAG_FWRST FIREWALL reset
  - RCC_FLAG_RMVF Remove reset Flag
  - RCC_FLAG_SFTRST Software reset
  - RCC_FLAG_IWDGRST Independent Watchdog reset
  - RCC_FLAG_WWDGRST Window Watchdog reset
  - RCC_FLAG_LPWRRST Low Power reset

Return value:

- The: new state of __FLAG__ (TRUE or

FALSE).

HSE Config

RCC_HSE_OFF	HSE clock deactivation
RCC_HSE_ON	HSE clock activation
RCC_HSE_BYPASS	External clock source for HSE clock

HSI48 Config

RCC_HSI48_OFF	HSI48 clock deactivation
RCC_HSI48_ON	HSI48 clock activation

HSI Config

RCC_HSI_OFF	HSI clock deactivation
RCC_HSI_ON	HSI clock activation
RCC_HSICALIBRATION_DEFAULT	

Interrupts

RCC_IT_LSIRDY	LSI Ready Interrupt flag
RCC_IT_LSERDY	LSE Ready Interrupt flag
RCC_IT_MSIRDY	MSI Ready Interrupt flag
RCC_IT_HSIRDY	HSI16 Ready Interrupt flag
RCC_IT_HSERDY	HSE Ready Interrupt flag
RCC_IT_PLLRDY	PLL Ready Interrupt flag
RCC_IT_PLLSAI1RDY	PLLSAI1 Ready Interrupt flag
RCC_IT_PLLSAI2RDY	PLLSAI2 Ready Interrupt flag
RCC_IT_CSS	Clock Security System Interrupt flag
RCC_IT_LSECSS	LSE Clock Security System Interrupt flag
RCC_IT_HSI48RDY	HSI48 Ready Interrupt flag

LSE Drive Config

RCC_LSEDRIVE_LOW	LSE low drive capability
RCC_LSEDRIVE_MEDIUMLOW	LSE medium low drive capability
RCC_LSEDRIVE_MEDIUMHIGH	LSE medium high drive capability
RCC_LSEDRIVE_HIGH	LSE high drive capability

LSE Config

RCC_LSE_OFF	LSE clock deactivation
RCC_LSE_ON	LSE clock activation
RCC_LSE_BYPASS	External clock source for LSE clock

LSI Config

RCC_LSI_OFF	LSI clock deactivation
RCC_LSI_ON	LSI clock activation

MCO1 Clock Source

RCC_MCO1SOURCE_NOCLOCK	MCO1 output disabled, no clock on MCO1
RCC_MCO1SOURCE_SYSCLK	SYSCLK selection as MCO1 source
RCC_MCO1SOURCE_MSI	MSI selection as MCO1 source
RCC_MCO1SOURCE_HSI	HSI selection as MCO1 source
RCC_MCO1SOURCE_HSE	HSE selection as MCO1 source
RCC_MCO1SOURCE_PLLCLK	PLLCLK selection as MCO1 source
RCC_MCO1SOURCE_LSI	LSI selection as MCO1 source
RCC_MCO1SOURCE_LSE	LSE selection as MCO1 source
RCC_MCO1SOURCE_HSI48	HSI48 selection as MCO1 source (STM32L43x/STM32L44x devices)

MCO1 Clock Prescaler

RCC_MCODIV_1	MCO not divided
RCC_MCODIV_2	MCO divided by 2
RCC_MCODIV_4	MCO divided by 4
RCC_MCODIV_8	MCO divided by 8
RCC_MCODIV_16	MCO divided by 16

MCO Index

RCC_MCO1	
RCC_MCO	MCO1 to be compliant with other families with 2 MCOs

MSI Clock Range

RCC_MSIRANGE_0	MSI = 100 KHz
RCC_MSIRANGE_1	MSI = 200 KHz
RCC_MSIRANGE_2	MSI = 400 KHz
RCC_MSIRANGE_3	MSI = 800 KHz
RCC_MSIRANGE_4	MSI = 1 MHz
RCC_MSIRANGE_5	MSI = 2 MHz
RCC_MSIRANGE_6	MSI = 4 MHz
RCC_MSIRANGE_7	MSI = 8 MHz
RCC_MSIRANGE_8	MSI = 16 MHz
RCC_MSIRANGE_9	MSI = 24 MHz
RCC_MSIRANGE_10	MSI = 32 MHz
RCC_MSIRANGE_11	MSI = 48 MHz

MSI Config

RCC_MSI_OFF	MSI clock deactivation
RCC_MSI_ON	MSI clock activation
RCC_MSICALIBRATION_DEFAULT	Default MSI calibration trimming value

Oscillator Type

RCC_OSCILLATORTYPE_NONE	Oscillator configuration unchanged
RCC_OSCILLATORTYPE_HSE	HSE to configure
RCC_OSCILLATORTYPE_HSI	HSI to configure
RCC_OSCILLATORTYPE_LSE	LSE to configure
RCC_OSCILLATORTYPE_LSI	LSI to configure
RCC_OSCILLATORTYPE_MSI	MSI to configure
RCC_OSCILLATORTYPE_HSI48	HSI48 to configure

PLLP Clock Divider

RCC_PLLP_DIV2	PLLP division factor = 2
RCC_PLLP_DIV3	PLLP division factor = 3
RCC_PLLP_DIV4	PLLP division factor = 4
RCC_PLLP_DIV5	PLLP division factor = 5
RCC_PLLP_DIV6	PLLP division factor = 6
RCC_PLLP_DIV7	PLLP division factor = 7
RCC_PLLP_DIV8	PLLP division factor = 8
RCC_PLLP_DIV9	PLLP division factor = 9
RCC_PLLP_DIV10	PLLP division factor = 10
RCC_PLLP_DIV11	PLLP division factor = 11
RCC_PLLP_DIV12	PLLP division factor = 12
RCC_PLLP_DIV13	PLLP division factor = 13
RCC_PLLP_DIV14	PLLP division factor = 14
RCC_PLLP_DIV15	PLLP division factor = 15
RCC_PLLP_DIV16	PLLP division factor = 16
RCC_PLLP_DIV17	PLLP division factor = 17
RCC_PLLP_DIV18	PLLP division factor = 18
RCC_PLLP_DIV19	PLLP division factor = 19
RCC_PLLP_DIV20	PLLP division factor = 20
RCC_PLLP_DIV21	PLLP division factor = 21
RCC_PLLP_DIV22	PLLP division factor = 22
RCC_PLLP_DIV23	PLLP division factor = 23
RCC_PLLP_DIV24	PLLP division factor = 24
RCC_PLLP_DIV25	PLLP division factor = 25
RCC_PLLP_DIV26	PLLP division factor = 26
RCC_PLLP_DIV27	PLLP division factor = 27
RCC_PLLP_DIV28	PLLP division factor = 28

RCC_PLLP_DIV29	PLLP division factor = 29
RCC_PLLP_DIV30	PLLP division factor = 30
RCC_PLLP_DIV31	PLLP division factor = 31

PLLQ Clock Divider

RCC_PLLQ_DIV2	PLLQ division factor = 2
RCC_PLLQ_DIV4	PLLQ division factor = 4
RCC_PLLQ_DIV6	PLLQ division factor = 6
RCC_PLLQ_DIV8	PLLQ division factor = 8

PLLR Clock Divider

RCC_PLLR_DIV2	PLLR division factor = 2
RCC_PLLR_DIV4	PLLR division factor = 4
RCC_PLLR_DIV6	PLLR division factor = 6
RCC_PLLR_DIV8	PLLR division factor = 8

PLLSAI1 Clock Output

RCC_PLLSAI1_SAI1CLK	PLLSAI1CLK selection from PLLSAI1
RCC_PLLSAI1_48M2CLK	PLL48M2CLK selection from PLLSAI1
RCC_PLLSAI1_ADC1CLK	PLLADC1CLK selection from PLLSAI1

PLLSAI2 Clock Output

RCC_PLLSAI2_SAI2CLK	PLLSAI2CLK selection from PLLSAI2
RCC_PLLSAI2_DSICLK	PLLDSICLK selection from PLLSAI2
RCC_PLLSAI2_LTDCCLK	PLLLTDCCLK selection from PLLSAI2

PLL Clock Output

RCC_PLL_SAI3CLK	PLLSAI3CLK selection from main PLL (for devices with PLLSAI2)
RCC_PLL_48M1CLK	PLL48M1CLK selection from main PLL
RCC_PLL_SYSCLK	PLLCLK selection from main PLL

PLL Clock Source

RCC_PLLSOURCE_NONE	No clock selected as PLL entry clock source
RCC_PLLSOURCE_MSI	MSI clock selected as PLL entry clock source
RCC_PLLSOURCE_HSI	HSI clock selected as PLL entry clock source
RCC_PLLSOURCE_HSE	HSE clock selected as PLL entry clock source

PLL Config

RCC_PLL_NONE	PLL configuration unchanged
RCC_PLL_OFF	PLL deactivation
RCC_PLL_ON	PLL activation

RCC RTC Clock Configuration

__HAL_RCC_RTC_ENABLE **Description:**

- Macros to enable or disable the RTC clock.

Return value:

- None

Notes:

- As the RTC is in the Backup domain and write access is denied to this domain after reset, you have to enable write access using HAL_PWR_EnableBkUpAccess() function before to configure the RTC (to be done once after reset). These macros must be used after the RTC clock source was selected.

__HAL_RCC_RTC_DISABLE

RTC Clock Source

RCC_RTCCLKSOURCE_NO_CLK	No clock used as RTC clock
RCC_RTCCLKSOURCE_LSE	LSE oscillator clock used as RTC clock
RCC_RTCCLKSOURCE_LSI	LSI oscillator clock used as RTC clock
RCC_RTCCLKSOURCE_HSE_DIV32	HSE oscillator clock divided by 32 used as RTC clock

Wake-Up from STOP Clock

RCC_STOP_WAKEUPCLOCK_MSI	MSI selection after wake-up from STOP
RCC_STOP_WAKEUPCLOCK_HSI	HSI selection after wake-up from STOP

System Clock Source

RCC_SYSCLKSOURCE_MSI	MSI selection as system clock
RCC_SYSCLKSOURCE_HSI	HSI selection as system clock
RCC_SYSCLKSOURCE_HSE	HSE selection as system clock
RCC_SYSCLKSOURCE_PLLCLK	PLL selection as system clock

System Clock Source Status

RCC_SYSCLKSOURCE_STATUS_MSI	MSI used as system clock
RCC_SYSCLKSOURCE_STATUS_HSI	HSI used as system clock
RCC_SYSCLKSOURCE_STATUS_HSE	HSE used as system clock
RCC_SYSCLKSOURCE_STATUS_PLLCLK	PLL used as system clock

System Clock Type

RCC_CLOCKTYPE_SYSCLK	SYSCLK to configure
RCC_CLOCKTYPE_HCLK	HCLK to configure
RCC_CLOCKTYPE_PCLK1	PCLK1 to configure
RCC_CLOCKTYPE_PCLK2	PCLK2 to configure

Timeout Values

RCC_DBP_TIMEOUT_VALUE

RCC_LSE_TIMEOUT_VALUE

52 HAL RCC Extension Driver

52.1 RCCEx Firmware driver registers structures

52.1.1 RCC_PLLSAI1InitTypeDef

Data Fields

- ***uint32_t PLLSAI1Source***
- ***uint32_t PLLSAI1M***
- ***uint32_t PLLSAI1N***
- ***uint32_t PLLSAI1P***
- ***uint32_t PLLSAI1Q***
- ***uint32_t PLLSAI1R***
- ***uint32_t PLLSAI1ClockOut***

Field Documentation

- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1Source***
  PLLSAI1Source: PLLSAI1 entry clock source. This parameter must be a value of ***RCC_PLL_Clock_Source***
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1M***
  PLLSAI1M: specifies the division factor for PLLSAI1 input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1N***
  PLLSAI1N: specifies the multiplication factor for PLLSAI1 VCO output clock. This parameter must be a number between 8 and 86 or 127 depending on devices.
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1P***
  PLLSAI1P: specifies the division factor for SAI clock. This parameter must be a value of ***RCC_PLLP_Clock_Divider***
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1Q***
  PLLSAI1Q: specifies the division factor for USB/RNG/SDMMC1 clock. This parameter must be a value of ***RCC_PLLQ_Clock_Divider***
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1R***
  PLLSAI1R: specifies the division factor for ADC clock. This parameter must be a value of ***RCC_PLLR_Clock_Divider***
- ***uint32_t RCC_PLLSAI1InitTypeDef::PLLSAI1ClockOut***
  PLLSAIClockOut: specifies PLLSAI1 output clock to be enabled. This parameter must be a value of ***RCC_PLLSAI1_Clock_Output***

52.1.2 RCC_PLLSAI2InitTypeDef

Data Fields

- ***uint32_t PLLSAI2Source***
- ***uint32_t PLLSAI2M***
- ***uint32_t PLLSAI2N***
- ***uint32_t PLLSAI2P***
- ***uint32_t PLLSAI2Q***
- ***uint32_t PLLSAI2R***
- ***uint32_t PLLSAI2ClockOut***

Field Documentation

- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2Source***
  PLLSAI2Source: PLLSAI2 entry clock source. This parameter must be a value of ***RCC_PLL_Clock_Source***
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2M***
  PLLSAI2M: specifies the division factor for PLLSAI2 input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2N***
  PLLSAI2N: specifies the multiplication factor for PLLSAI2 VCO output clock. This parameter must be a number between 8 and 86 or 127 depending on devices.
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2P***
  PLLSAI2P: specifies the division factor for SAI clock. This parameter must be a value of ***RCC_PLLP_Clock_Divider***
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2Q***
  PLLSAI2Q: specifies the division factor for DSI clock. This parameter must be a value of ***RCC_PLLQ_Clock_Divider***
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2R***
  PLLSAI2R: specifies the division factor for ADC clock. This parameter must be a value of ***RCC_PLLR_Clock_Divider***
- ***uint32_t RCC_PLLSAI2InitTypeDef::PLLSAI2ClockOut***
  PLLSAIClockOut: specifies PLLSAI2 output clock to be enabled. This parameter must be a value of ***RCC_PLLSAI2_Clock_Output***

52.1.3 RCC_PeriphCLKInitTypeDef

Data Fields

- ***uint32_t PeriphClockSelection***
- ***RCC_PLLSAI1InitTypeDef PLLSAI1***
- ***RCC_PLLSAI2InitTypeDef PLLSAI2***
- ***uint32_t Usart1ClockSelection***
- ***uint32_t Usart2ClockSelection***
- ***uint32_t Usart3ClockSelection***
- ***uint32_t Uart4ClockSelection***
- ***uint32_t Uart5ClockSelection***
- ***uint32_t Lpuart1ClockSelection***
- ***uint32_t I2c1ClockSelection***
- ***uint32_t I2c2ClockSelection***
- ***uint32_t I2c3ClockSelection***
- ***uint32_t I2c4ClockSelection***
- ***uint32_t Lptim1ClockSelection***
- ***uint32_t Lptim2ClockSelection***
- ***uint32_t Sai1ClockSelection***
- ***uint32_t Sai2ClockSelection***
- ***uint32_t UsbClockSelection***
- ***uint32_t Sdmmc1ClockSelection***
- ***uint32_t RngClockSelection***
- ***uint32_t AdcClockSelection***
- ***uint32_t Dfsdm1ClockSelection***
- ***uint32_t Dfsdm1AudioClockSelection***
- ***uint32_t LtdcClockSelection***
- ***uint32_t DsiClockSelection***
- ***uint32_t OspiClockSelection***
- ***uint32_t RTCClockSelection***

Field Documentation

- ***uint32_t RCC_PeriphCLKInitTypeDef::PeriphClockSelection***
  The Extended Clock to be configured. This parameter can be a value of ***RCCEx_Periph_Clock_Selection***
- ***RCC_PLLSAI1InitTypeDef RCC_PeriphCLKInitTypeDef::PLLSAI1***
  PLLSAI1 structure parameters. This parameter will be used only when PLLSAI1 is selected as Clock Source for SAI1, USB/RNG/SDMMC1 or ADC
- ***RCC_PLLSAI2InitTypeDef RCC_PeriphCLKInitTypeDef::PLLSAI2***
  PLLSAI2 structure parameters. This parameter will be used only when PLLSAI2 is selected as Clock Source for SAI2 or ADC
- ***uint32_t RCC_PeriphCLKInitTypeDef::Usart1ClockSelection***
  Specifies USART1 clock source. This parameter can be a value of ***RCCEx_USART1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Usart2ClockSelection***
  Specifies USART2 clock source. This parameter can be a value of ***RCCEx_USART2_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Usart3ClockSelection***
  Specifies USART3 clock source. This parameter can be a value of ***RCCEx_USART3_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Uart4ClockSelection***
  Specifies UART4 clock source. This parameter can be a value of ***RCCEx_UART4_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Uart5ClockSelection***
  Specifies UART5 clock source. This parameter can be a value of ***RCCEx_UART5_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Lpuart1ClockSelection***
  Specifies LPUART1 clock source. This parameter can be a value of ***RCCEx_LPUART1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::I2c1ClockSelection***
  Specifies I2C1 clock source. This parameter can be a value of ***RCCEx_I2C1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::I2c2ClockSelection***
  Specifies I2C2 clock source. This parameter can be a value of ***RCCEx_I2C2_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::I2c3ClockSelection***
  Specifies I2C3 clock source. This parameter can be a value of ***RCCEx_I2C3_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::I2c4ClockSelection***
  Specifies I2C4 clock source. This parameter can be a value of ***RCCEx_I2C4_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Lptim1ClockSelection***
  Specifies LPTIM1 clock source. This parameter can be a value of ***RCCEx_LPTIM1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Lptim2ClockSelection***
  Specifies LPTIM2 clock source. This parameter can be a value of ***RCCEx_LPTIM2_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Sai1ClockSelection***
  Specifies SAI1 clock source. This parameter can be a value of ***RCCEx_SAI1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Sai2ClockSelection***
  Specifies SAI2 clock source. This parameter can be a value of ***RCCEx_SAI2_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::UsbClockSelection***
  Specifies USB clock source (warning: same source for SDMMC1 and RNG). This parameter can be a value of ***RCCEx_USB_Clock_Source***

- ***uint32_t RCC_PeriphCLKInitTypeDef::Sdmmc1ClockSelection***
  Specifies SDMMC1 clock source (warning: same source for USB and RNG). This parameter can be a value of ***RCCEx_SDMMC1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::RngClockSelection***
  Specifies RNG clock source (warning: same source for USB and SDMMC1). This parameter can be a value of ***RCCEx_RNG_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::AdcClockSelection***
  Specifies ADC interface clock source. This parameter can be a value of ***RCCEx_ADC_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Dfsdm1ClockSelection***
  Specifies DFSDM1 clock source. This parameter can be a value of ***RCCEx_DFSDM1_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::Dfsdm1AudioClockSelection***
  Specifies DFSDM1 audio clock source. This parameter can be a value of ***RCCEx_DFSDM1_Audio_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::LtdcClockSelection***
  Specifies LTDC clock source. This parameter can be a value of ***RCCEx_LTDC_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::DsiClockSelection***
  Specifies DSI clock source. This parameter can be a value of ***RCCEx_DSI_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::OspiClockSelection***
  Specifies OctoSPI clock source. This parameter can be a value of ***RCCEx_OSPI_Clock_Source***
- ***uint32_t RCC_PeriphCLKInitTypeDef::RTCClockSelection***
  Specifies RTC clock source. This parameter can be a value of ***RCC_RTC_Clock_Source***

52.1.4 RCC_CRSInitTypeDef

Data Fields

- ***uint32_t Prescaler***
- ***uint32_t Source***
- ***uint32_t Polarity***
- ***uint32_t ReloadValue***
- ***uint32_t ErrorLimitValue***
- ***uint32_t HSI48CalibrationValue***

Field Documentation

- ***uint32_t RCC_CRSInitTypeDef::Prescaler***
  Specifies the division factor of the SYNC signal. This parameter can be a value of ***RCCEx_CRS_SynchroDivider***
- ***uint32_t RCC_CRSInitTypeDef::Source***
  Specifies the SYNC signal source. This parameter can be a value of ***RCCEx_CRS_SynchroSource***
- ***uint32_t RCC_CRSInitTypeDef::Polarity***
  Specifies the input polarity for the SYNC signal source. This parameter can be a value of ***RCCEx_CRS_SynchroPolarity***
- ***uint32_t RCC_CRSInitTypeDef::ReloadValue***
  Specifies the value to be loaded in the frequency error counter with each SYNC event. It can be calculated in using macro **__HAL_RCC_CRS_RELOADVALUE_CALCULATE(__FTARGET__, __FSYNC__)** This parameter must be a number between 0 and 0xFFFF or a value of ***RCCEx_CRS_ReloadValueDefault*** .

- ***uint32_t RCC_CRSInitTypeDef::ErrorLimitValue***
  Specifies the value to be used to evaluate the captured frequency error value. This parameter must be a number between 0 and 0xFF or a value of ***RCCEx_CRS_ErrorLimitDefault***
- ***uint32_t RCC_CRSInitTypeDef::HSI48CalibrationValue***
  Specifies a user-programmable trimming value to the HSI48 oscillator. This parameter must be a number between 0 and 0x3F or a value of ***RCCEx_CRS_HSI48CalibrationDefault***

52.1.5 RCC_CRSSynchroInfoTypeDef

Data Fields

- ***uint32_t ReloadValue***
- ***uint32_t HSI48CalibrationValue***
- ***uint32_t FreqErrorCapture***
- ***uint32_t FreqErrorDirection***

Field Documentation

- ***uint32_t RCC_CRSSynchroInfoTypeDef::ReloadValue***
  Specifies the value loaded in the Counter reload value. This parameter must be a number between 0 and 0xFFFF
- ***uint32_t RCC_CRSSynchroInfoTypeDef::HSI48CalibrationValue***
  Specifies value loaded in HSI48 oscillator smooth trimming. This parameter must be a number between 0 and 0x3F
- ***uint32_t RCC_CRSSynchroInfoTypeDef::FreqErrorCapture***
  Specifies the value loaded in the .FECAP, the frequency error counter value latched in the time of the last SYNC event. This parameter must be a number between 0 and 0xFFFF
- ***uint32_t RCC_CRSSynchroInfoTypeDef::FreqErrorDirection***
  Specifies the value loaded in the .FEDIR, the counting direction of the frequency error counter latched in the time of the last SYNC event. It shows whether the actual frequency is below or above the target. This parameter must be a value of ***RCCEx_CRS_FreqErrorDirection***

52.2 RCCEx Firmware driver API description

52.2.1 Extended Peripheral Control functions

This subsection provides a set of functions allowing to control the RCC Clocks frequencies.

Important note: Care must be taken when HAL_RCCEx_PeriphCLKConfig() is used to select the RTC clock source; in this case the Backup domain will be reset in order to modify the RTC Clock source, as consequence RTC registers (including the backup registers) are set to their reset values.

This section contains the following APIs:

- ***HAL_RCCEx_PeriphCLKConfig()***
- ***HAL_RCCEx_GetPeriphCLKConfig()***
- ***HAL_RCCEx_GetPeriphCLKFreq()***

52.2.2 Extended clock management functions

This subsection provides a set of functions allowing to control the activation or deactivation of MSI PLL-mode, PLLSAI1, PLLSAI2, LSE CSS, Low speed clock output and clock after wake-up from STOP mode.

This section contains the following APIs:

- ***HAL_RCCEx_EnablePLLSAI1()***
- ***HAL_RCCEx_DisablePLLSAI1()***
- ***HAL_RCCEx_EnablePLLSAI2()***
- ***HAL_RCCEx_DisablePLLSAI2()***
- ***HAL_RCCEx_WakeUpStopCLKConfig()***
- ***HAL_RCCEx_StandbyMSIRangeConfig()***
- ***HAL_RCCEx_EnableLSECSS()***
- ***HAL_RCCEx_DisableLSECSS()***
- ***HAL_RCCEx_EnableLSECSS_IT()***
- ***HAL_RCCEx_LSECSS_IRQHandler()***
- ***HAL_RCCEx_LSECSS_Callback()***
- ***HAL_RCCEx_EnableLSCO()***
- ***HAL_RCCEx_DisableLSCO()***
- ***HAL_RCCEx_EnableMSIPLLMode()***
- ***HAL_RCCEx_DisableMSIPLLMode()***

52.2.3 Extended Clock Recovery System Control functions

For devices with Clock Recovery System feature (CRS), RCC Extention HAL driver can be used as follows:

1. In System clock config, HSI48 needs to be enabled
2. Enable CRS clock in IP MSP init which will use CRS functions
3. Call CRS functions as follows:
   a. Prepare synchronization configuration necessary for HSI48 calibration
      – Default values can be set for frequency Error Measurement (reload and error limit) and also HSI48 oscillator smooth trimming.
      – Macro __HAL_RCC_CRS_RELOADVALUE_CALCULATE can be also used to calculate directly reload value with target and sychronization frequencies values
   b. Call function HAL_RCCEx_CRSConfig which
      – Resets CRS registers to their default values.
      – Configures CRS registers with synchronization configuration
      – Enables automatic calibration and frequency error counter feature Note: When using USB LPM (Link Power Management) and the device is in Sleep mode, the periodic USB SOF will not be generated by the host. No SYNC signal will therefore be provided to the CRS to calibrate the HSI48 on the run. To guarantee the required clock precision after waking up from Sleep mode, the LSE or reference clock on the GPIOs should be used as SYNC signal.
   c. A polling function is provided to wait for complete synchronization
      – Call function HAL_RCCEx_CRSWaitSynchronization()
      – According to CRS status, user can decide to adjust again the calibration or continue application if synchronization is OK
4. User can retrieve information related to synchronization in calling function HAL_RCCEx_CRSGetSynchronizationInfo()
5. Regarding synchronization status and synchronization information, user can try a new calibration in changing synchronization configuration and call again

HAL_RCCEx_CRSConfig. Note: When the SYNC event is detected during the downcounting phase (before reaching the zero value), it means that the actual frequency is lower than the target (and so, that the TRIM value should be incremented), while when it is detected during the upcounting phase it means that the actual frequency is higher (and that the TRIM value should be decremented).

6. In interrupt mode, user can resort to the available macros (__HAL_RCC_CRS_XXX_IT). Interrupts will go through CRS Handler (CRS_IRQn/CRS_IRQHandler)
   - Call function HAL_RCCEx_CRSConfig()
   - Enable CRS_IRQn (thanks to NVIC functions)
   - Enable CRS interrupt (__HAL_RCC_CRS_ENABLE_IT)
   - Implement CRS status management in the following user callbacks called from HAL_RCCEx_CRS_IRQHandler():
     - HAL_RCCEx_CRS_SyncOkCallback()
     - HAL_RCCEx_CRS_SyncWarnCallback()
     - HAL_RCCEx_CRS_ExpectedSyncCallback()
     - HAL_RCCEx_CRS_ErrorCallback()
7. To force a SYNC EVENT, user can use the function HAL_RCCEx_CRSSoftwareSynchronizationGenerate(). This function can be called before calling HAL_RCCEx_CRSConfig (for instance in Systick handler)

This section contains the following APIs:

- ***HAL_RCCEx_CRSConfig()***
- ***HAL_RCCEx_CRSSoftwareSynchronizationGenerate()***
- ***HAL_RCCEx_CRSGetSynchronizationInfo()***
- ***HAL_RCCEx_CRSWaitSynchronization()***
- ***HAL_RCCEx_CRS_IRQHandler()***
- ***HAL_RCCEx_CRS_SyncOkCallback()***
- ***HAL_RCCEx_CRS_SyncWarnCallback()***
- ***HAL_RCCEx_CRS_ExpectedSyncCallback()***
- ***HAL_RCCEx_CRS_ErrorCallback()***

52.2.4 Detailed description of functions

HAL_RCCEx_PeriphCLKConfig

Function name: **HAL_StatusTypeDef HAL_RCCEx_PeriphCLKConfig (RCC_PeriphCLKInitTypeDef * PeriphClkInit)**

Function description: Initialize the RCC extended peripherals clocks according to the specified parameters in the RCC_PeriphCLKInitTypeDef.

Parameters:

- **PeriphClkInit:** pointer to an RCC_PeriphCLKInitTypeDef structure that contains a field PeriphClockSelection which can be a combination of the following values:
  - RCC_PERIPHCLK_RTC RTC peripheral clock
  - RCC_PERIPHCLK_ADC ADC peripheral clock
  - RCC_PERIPHCLK_I2C1 I2C1 peripheral clock
  - RCC_PERIPHCLK_I2C2 I2C2 peripheral clock
  - RCC_PERIPHCLK_I2C3 I2C3 peripheral clock
  - RCC_PERIPHCLK_I2C4 I2C4 peripheral clock (only for devices with I2C4)
  - RCC_PERIPHCLK_LPTIM1 LPTIM1 peripheral clock
  - RCC_PERIPHCLK_LPTIM2 LPTIM2 peripheral clock
  - RCC_PERIPHCLK_LPUART1 LPUART1 peripheral

clock
  - RCC_PERIPHCLK_RNG RNG peripheral clock
  - RCC_PERIPHCLK_SAI1 SAI1 peripheral clock
  - RCC_PERIPHCLK_SAI2 SAI2 peripheral clock (only for devices with SAI2)
  - RCC_PERIPHCLK_SDMMC1 SDMMC1 peripheral clock
  - RCC_PERIPHCLK_USART1 USART1 peripheral clock
  - RCC_PERIPHCLK_USART2 USART1 peripheral clock
  - RCC_PERIPHCLK_USART3 USART1 peripheral clock
  - RCC_PERIPHCLK_UART4 USART1 peripheral clock (only for devices with UART4)
  - RCC_PERIPHCLK_UART5 USART1 peripheral clock (only for devices with UART5)
  - RCC_PERIPHCLK_USB USB peripheral clock (only for devices with USB)
  - RCC_PERIPHCLK_DFSDM1 DFSDM1 peripheral kernel clock (only for devices with DFSDM1)
  - RCC_PERIPHCLK_DFSDM1AUDIO DFSDM1 peripheral audio clock (only for devices with DFSDM1)
  - RCC_PERIPHCLK_LTDC LTDC peripheral clock (only for devices with LTDC)
  - RCC_PERIPHCLK_DSI DSI peripheral clock (only for devices with DSI)
  - RCC_PERIPHCLK_OSPI OctoSPI peripheral clock (only for devices with OctoSPI)

Return values
- **HAL:** status

Notes
- Care must be taken when HAL_RCCEx_PeriphCLKConfig() is used to select the RTC clock source: in this case the access to Backup domain is enabled.

HAL_RCCEx_GetPeriphCLKConfig

Function name **void HAL_RCCEx_GetPeriphCLKConfig (RCC_PeriphCLKInitTypeDef * PeriphClkInit)**

Function description Get the RCC_ClkInitStruct according to the internal RCC configuration registers.

Parameters
- **PeriphClkInit:** pointer to an RCC_PeriphCLKInitTypeDef structure that returns the configuration information for the Extended Peripherals clocks(SAI1, SAI2, LPTIM1, LPTIM2, I2C1, I2C2, I2C3, I2C4, LPUART, USART1, USART2, USART3, UART4, UART5, RTC, ADCx, DFSDMx, SWPMI1, USB, SDMMC1 and RNG).

Return values
- **None:**

HAL_RCCEx_GetPeriphCLKFreq

Function name **uint32_t HAL_RCCEx_GetPeriphCLKFreq (uint32_t PeriphClk)**

Function description Return the peripheral clock frequency for peripherals with clock source from PLLSAIs.

Parameters
- **PeriphClk:** Peripheral clock identifier This parameter can be

one of the following values:
- RCC_PERIPHCLK_RTC RTC peripheral clock
- RCC_PERIPHCLK_ADC ADC peripheral clock
- RCC_PERIPHCLK_I2C1 I2C1 peripheral clock
- RCC_PERIPHCLK_I2C2 I2C2 peripheral clock
- RCC_PERIPHCLK_I2C3 I2C3 peripheral clock
- RCC_PERIPHCLK_I2C4 I2C4 peripheral clock (only for devices with I2C4)
- RCC_PERIPHCLK_LPTIM1 LPTIM1 peripheral clock
- RCC_PERIPHCLK_LPTIM2 LPTIM2 peripheral clock
- RCC_PERIPHCLK_LPUART1 LPUART1 peripheral clock
- RCC_PERIPHCLK_RNG RNG peripheral clock
- RCC_PERIPHCLK_SAI1 SAI1 peripheral clock
- RCC_PERIPHCLK_SAI2 SAI2 peripheral clock (only for devices with SAI2)
- RCC_PERIPHCLK_SDMMC1 SDMMC1 peripheral clock
- RCC_PERIPHCLK_USART1 USART1 peripheral clock
- RCC_PERIPHCLK_USART2 USART1 peripheral clock
- RCC_PERIPHCLK_USART3 USART1 peripheral clock
- RCC_PERIPHCLK_UART4 USART1 peripheral clock (only for devices with UART4)
- RCC_PERIPHCLK_UART5 USART1 peripheral clock (only for devices with UART5)
- RCC_PERIPHCLK_USB USB peripheral clock (only for devices with USB)
- RCC_PERIPHCLK_DFSDM1 DFSDM1 peripheral kernel clock (only for devices with DFSDM1)
- RCC_PERIPHCLK_DFSDM1AUDIO DFSDM1 peripheral audio clock (only for devices with DFSDM1)
- RCC_PERIPHCLK_LTDC LTDC peripheral clock (only for devices with LTDC)
- RCC_PERIPHCLK_DSI DSI peripheral clock (only for devices with DSI)
- RCC_PERIPHCLK_OSPI OctoSPI peripheral clock (only for devices with OctoSPI)

Return values
- **Frequency:** in Hz

Notes
- Return 0 if peripheral clock identifier not managed by this API

HAL_RCCEx_EnablePLLSAI1

Function name **HAL_StatusTypeDef HAL_RCCEx_EnablePLLSAI1 (RCC_PLLSAI1InitTypeDef * PLLSAI1Init)**

Function description Enable PLLSAI1.

Parameters
- **PLLSAI1Init:** pointer to an RCC_PLLSAI1InitTypeDef structure that contains the configuration information for the PLLSAI1

Return values
- **HAL:** status

HAL_RCCEx_DisablePLLSAI1

Function name **HAL_StatusTypeDef HAL_RCCEx_DisablePLLSAI1 (void)**

Function description Disable PLLSAI1.

Return values
- **HAL:** status

HAL_RCCEx_EnablePLLSAI2

Function name **HAL_StatusTypeDef HAL_RCCEx_EnablePLLSAI2 (RCC_PLLSAI2InitTypeDef * PLLSAI2Init)**

Function description Enable PLLSAI2.

Parameters
- **PLLSAI2Init:** pointer to an RCC_PLLSAI2InitTypeDef structure that contains the configuration information for the PLLSAI2

Return values
- **HAL:** status

HAL_RCCEx_DisablePLLSAI2

Function name **HAL_StatusTypeDef HAL_RCCEx_DisablePLLSAI2 (void)**

Function description Disable PLLISAI2.

Return values
- **HAL:** status

HAL_RCCEx_WakeUpStopCLKConfig

Function name **void HAL_RCCEx_WakeUpStopCLKConfig (uint32_t WakeUpClk)**

Function description Configure the oscillator clock source for wakeup from Stop and CSS backup clock.

Parameters
- **WakeUpClk:** Wakeup clock This parameter can be one of the following values:
  - RCC_STOP_WAKEUPCLOCK_MSI MSI oscillator selection
  - RCC_STOP_WAKEUPCLOCK_HSI HSI oscillator selection

Return values
- **None:**

Notes
- This function shall not be called after the Clock Security System on HSE has been enabled.

HAL_RCCEx_StandbyMSIRangeConfig

Function name **void HAL_RCCEx_StandbyMSIRangeConfig (uint32_t MSIRange)**

Function description Configure the MSI range after standby mode.

Parameters
- **MSIRange:** MSI range This parameter can be one of the following values:
  - RCC_MSIRANGE_4 Range 4 around 1 MHz
  - RCC_MSIRANGE_5 Range 5 around 2 MHz
  - RCC_MSIRANGE_6 Range 6 around 4 MHz (reset

value)
  - RCC_MSIRANGE_7 Range 7 around 8 MHz

Return values
- **None:**

Notes
- After Standby its frequency can be selected between 4 possible values (1, 2, 4 or 8 MHz).

HAL_RCCEx_EnableLSECSS

Function name **void HAL_RCCEx_EnableLSECSS (void)**

Function description Enable the LSE Clock Security System.

Return values
- **None:**

Notes
- Prior to enable the LSE Clock Security System, LSE oscillator is to be enabled with HAL_RCC_OscConfig() and the LSE oscillator clock is to be selected as RTC clock with HAL_RCCEx_PeriphCLKConfig().

HAL_RCCEx_DisableLSECSS

Function name **void HAL_RCCEx_DisableLSECSS (void)**

Function description Disable the LSE Clock Security System.

Return values
- **None:**

Notes
- LSE Clock Security System can only be disabled after a LSE failure detection.

HAL_RCCEx_EnableLSECSS_IT

Function name **void HAL_RCCEx_EnableLSECSS_IT (void)**

Function description Enable the LSE Clock Security System Interrupt & corresponding EXTI line.

Return values
- **None:**

Notes
- LSE Clock Security System Interrupt is mapped on RTC EXTI line 19

HAL_RCCEx_LSECSS_IRQHandler

Function name **void HAL_RCCEx_LSECSS_IRQHandler (void)**

Function description Handle the RCC LSE Clock Security System interrupt request.

Return values
- **None:**

HAL_RCCEx_LSECSS_Callback

Function name **void HAL_RCCEx_LSECSS_Callback (void)**

Function description RCCEx LSE Clock Security System interrupt callback.

Return values
- **none:**

HAL_RCCEx_EnableLSCO

Function name	**void HAL_RCCEx_EnableLSCO (uint32_t LSCOSource)**
Function description	Select the Low Speed clock source to output on LSCO pin (PA2).
Parameters	• **LSCOSource:** specifies the Low Speed clock source to output. This parameter can be one of the following values: – RCC_LSCOSOURCE_LSI LSI clock selected as LSCO source – RCC_LSCOSOURCE_LSE LSE clock selected as LSCO source
Return values	• **None:**

HAL_RCCEx_DisableLSCO

Function name	**void HAL_RCCEx_DisableLSCO (void)**
Function description	Disable the Low Speed clock output.
Return values	• **None:**

HAL_RCCEx_EnableMSIPLLMode

Function name	**void HAL_RCCEx_EnableMSIPLLMode (void)**
Function description	Enable the PLL-mode of the MSI.
Return values	• **None:**
Notes	• Prior to enable the PLL-mode of the MSI for automatic hardware calibration LSE oscillator is to be enabled with HAL_RCC_OscConfig().

HAL_RCCEx_DisableMSIPLLMode

Function name	**void HAL_RCCEx_DisableMSIPLLMode (void)**
Function description	Disable the PLL-mode of the MSI.
Return values	• **None:**
Notes	• PLL-mode of the MSI is automatically reset when LSE oscillator is disabled.

HAL_RCCEx_CRSConfig

Function name	**void HAL_RCCEx_CRSConfig (RCC_CRSInitTypeDef * pInit)**
Function description	Start automatic synchronization for polling mode.
Parameters	• **pInit:** Pointer on RCC_CRSInitTypeDef structure
Return values	• **None:**

HAL_RCCEx_CRSSoftwareSynchronizationGenerate

Function name	**void HAL_RCCEx_CRSSoftwareSynchronizationGenerate (void)**

Function description Generate the software synchronization event.

Return values
- **None:**

HAL_RCCEx_CRSGetSynchronizationInfo

Function name **void HAL_RCCEx_CRSGetSynchronizationInfo (RCC_CRSSynchroInfoTypeDef * pSynchroInfo)**

Function description Return synchronization info.

Parameters
- **pSynchroInfo:** Pointer on RCC_CRSSynchroInfoTypeDef structure

Return values
- **None:**

HAL_RCCEx_CRSWaitSynchronization

Function name **uint32_t HAL_RCCEx_CRSWaitSynchronization (uint32_t Timeout)**

Function description Wait for CRS Synchronization status.

Parameters
- **Timeout:** Duration of the timeout

Return values
- **Combination:** of Synchronization status This parameter can be a combination of the following values:
  - RCC_CRS_TIMEOUT
  - RCC_CRS_SYNCOK
  - RCC_CRS_SYNCWARN
  - RCC_CRS_SYNCERR
  - RCC_CRS_SYNCMISS
  - RCC_CRS_TRIMOVF

Notes
- Timeout is based on the maximum time to receive a SYNC event based on synchronization frequency.
- If Timeout set to HAL_MAX_DELAY, HAL_TIMEOUT will be never returned.

HAL_RCCEx_CRS_IRQHandler

Function name **void HAL_RCCEx_CRS_IRQHandler (void)**

Function description Handle the Clock Recovery System interrupt request.

Return values
- **None:**

HAL_RCCEx_CRS_SyncOkCallback

Function name **void HAL_RCCEx_CRS_SyncOkCallback (void)**

Function description RCCEx Clock Recovery System SYNCOK interrupt callback.

Return values
- **none:**

HAL_RCCEx_CRS_SyncWarnCallback

Function name **void HAL_RCCEx_CRS_SyncWarnCallback (void)**

Function description RCCEx Clock Recovery System SYNCWARN interrupt callback.

Return values
- **none:**

HAL_RCCEx_CRS_ExpectedSyncCallback

Function name **void HAL_RCCEx_CRS_ExpectedSyncCallback (void)**

Function description RCCEx Clock Recovery System Expected SYNC interrupt callback.

Return values
- **none:**

HAL_RCCEx_CRS_ErrorCallback

Function name **void HAL_RCCEx_CRS_ErrorCallback (uint32_t Error)**

Function description RCCEx Clock Recovery System Error interrupt callback.

Parameters
- **Error:** Combination of Error status. This parameter can be a combination of the following values:
  - RCC_CRS_SYNCERR
  - RCC_CRS_SYNCMISS
  - RCC_CRS_TRIMOVF

Return values
- **none:**

52.3 RCCEx Firmware driver defines

52.3.1 RCCEx

ADC Clock Source

RCC_ADCCLKSOURCE_NONE

RCC_ADCCLKSOURCE_PLLSAI1

RCC_ADCCLKSOURCE_SYSCLK

RCCEx CRS ErrorLimitDefault

RCC_CRS_ERRORLIMIT_DEFAULT Default Frequency error limit

RCCEx CRS Extended Features

__HAL_RCC_CRS_FREQ_ERROR_COUNTER_ENABLE

Description:
- Enable the oscillator clock for frequency error counter.

Return value:
- None

Notes:
- when the CEN bit is set the CRS_CFGR register becomes write-protected.

__HAL_RCC_CRS_FREQ_ERROR_COUNTER_DISABLE

Description:
- Disable the oscillator clock

for frequency error counter.

Return value:

- None

__HAL_RCC_CRS_AUTOMATIC_CALIB_ENABLE

Description:

- Enable the automatic hardware adjustement of TRIM bits.

Return value:

- None

Notes:

- When the AUTOTRIMEN bit is set the CRS_CFGR register becomes write-protected.

__HAL_RCC_CRS_AUTOMATIC_CALIB_DISABLE

Description:

- Enable or disable the automatic hardware adjustement of TRIM bits.

Return value:

- None

__HAL_RCC_CRS_RELOADVALUE_CALCULATE

Description:

- Macro to calculate reload value to be set in CRS register according to target and sync frequencies.

Parameters:

- __FTARGET__: Target frequency (value in Hz)
- __FSYNC__: Synchronization signal frequency (value in Hz)

Return value:

- None

Notes:

- The RELOAD value should be selected according to the ratio between the target frequency and the frequency of the synchronization source after prescaling. It is then decreased by one in order to reach the expected synchronization on the zero value. The formula is the

following: RELOAD = (fTARGET / fSYNC) -1

RCCEx CRS Flags

RCC_CRS_FLAG_SYNCOK	SYNC event OK flag
RCC_CRS_FLAG_SYNCWARN	SYNC warning flag
RCC_CRS_FLAG_ERR	Error flag
RCC_CRS_FLAG_ESYNC	Expected SYNC flag
RCC_CRS_FLAG_SYNCERR	SYNC error
RCC_CRS_FLAG_SYNCMISS	SYNC missed
RCC_CRS_FLAG_TRIMOVF	Trimming overflow or underflow

RCCEx CRS FreqErrorDirection

RCC_CRS_FREQERRORDIR_UP	Upcounting direction, the actual frequency is above the target
RCC_CRS_FREQERRORDIR_DOWN	Downcounting direction, the actual frequency is below the target

RCCEx CRS HSI48CalibrationDefault

RCC_CRS_HSI48CALIBRATION_DEFAULT	The default value is 32, which corresponds to the middle of the trimming interval. The trimming step is around 67 kHz between two consecutive TRIM steps. A higher TRIM value corresponds to a higher output frequency

RCCEx CRS Interrupt Sources

RCC_CRS_IT_SYNCOK	SYNC event OK
RCC_CRS_IT_SYNCWARN	SYNC warning
RCC_CRS_IT_ERR	Error
RCC_CRS_IT_ESYNC	Expected SYNC
RCC_CRS_IT_SYNCERR	SYNC error
RCC_CRS_IT_SYNCMISS	SYNC missed
RCC_CRS_IT_TRIMOVF	Trimming overflow or underflow

RCCEx CRS ReloadValueDefault

RCC_CRS_RELOADVALUE_DEFAULT	The reset value of the RELOAD field corresponds to a target frequency of 48 MHz and a synchronization signal frequency of 1 kHz (SOF signal from USB).

RCCEx CRS Status

RCC_CRS_NONE

RCC_CRS_TIMEOUT

RCC_CRS_SYNCOK

RCC_CRS_SYNCWARN

RCC_CRS_SYNCERR

RCC_CRS_SYNCMISS

RCC_CRS_TRIMOVF

RCCEx CRS SynchroDivider

RCC_CRS_SYNC_DIV1	Synchro Signal not divided (default)
RCC_CRS_SYNC_DIV2	Synchro Signal divided by 2
RCC_CRS_SYNC_DIV4	Synchro Signal divided by 4
RCC_CRS_SYNC_DIV8	Synchro Signal divided by 8
RCC_CRS_SYNC_DIV16	Synchro Signal divided by 16
RCC_CRS_SYNC_DIV32	Synchro Signal divided by 32
RCC_CRS_SYNC_DIV64	Synchro Signal divided by 64
RCC_CRS_SYNC_DIV128	Synchro Signal divided by 128

RCCEx CRS SynchroPolarity

RCC_CRS_SYNC_POLARITY_RISING	Synchro Active on rising edge (default)
RCC_CRS_SYNC_POLARITY_FALLING	Synchro Active on falling edge

RCCEx CRS SynchroSource

RCC_CRS_SYNC_SOURCE_GPIO	Synchro Signal source GPIO
RCC_CRS_SYNC_SOURCE_LSE	Synchro Signal source LSE
RCC_CRS_SYNC_SOURCE_USB	Synchro Signal source USB SOF (default)

DFSDM1 Audio Clock Source

RCC_DFSDM1AUDIOCLKSOURCE_SAI1

RCC_DFSDM1AUDIOCLKSOURCE_HSI

RCC_DFSDM1AUDIOCLKSOURCE_MSI

DFSDM1 Clock Source

RCC_DFSDM1CLKSOURCE_PCLK2

RCC_DFSDM1CLKSOURCE_SYSCLK

DSI Clock Source

RCC_DSICLKSOURCE_DSIPHY

RCC_DSICLKSOURCE_PLLSAI2

RCCEx Exported Macros

__HAL_RCC_PLLSAI1_CONFIG

Description:

- Macro to configure the PLLSAI1 clock multiplication and division factors.

Parameters:

- __PLLSAI1M__: specifies the division factor of PLLSAI1 input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16.

- __PLLSAI1N__: specifies the multiplication factor for PLLSAI1 VCO output clock. This parameter must be a number between 8 and 86.
- __PLLSAI1P__: specifies the division factor for SAI clock. This parameter must be a number in the range (7 or 17) for STM32L47xxx/L48xxx else (2 to 31). SAI1 clock frequency = f(PLLSAI1) / PLLSAI1P
- __PLLSAI1Q__: specifies the division factor for USB/RNG/SDMMC1 clock. This parameter must be in the range (2, 4, 6 or 8). USB/RNG/SDMMC1 clock frequency = f(PLLSAI1) / PLLSAI1Q
- __PLLSAI1R__: specifies the division factor for SAR ADC clock. This parameter must be in the range (2, 4, 6 or 8). ADC clock frequency = f(PLLSAI1) / PLLSAI1R

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)
- You have to set the PLLSAI1N parameter correctly to ensure that the VCO output frequency is between 64 and 344 MHz. PLLSAI1 clock frequency = f(PLLSAI1) multiplied by PLLSAI1N

__HAL_RCC_PLLSAI1_MULN_CONFIG

Description:

- Macro to configure the PLLSAI1 clock multiplication factor N.

Parameters:

- __PLLSAI1N__: specifies the multiplication factor for PLLSAI1 VCO output clock. This parameter must be a number between 8 and 86.

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)
- You have to set the PLLSAI1N parameter correctly to ensure that the VCO output

frequency is between 64 and 344 MHz. Use to set PLLSAI1 clock frequency = f(PLLSAI1) multiplied by PLLSAI1N

__HAL_RCC_PLLSAI1_DIVM_CONFIG

Description:

- Macro to configure the PLLSAI1 input clock division factor M.

Parameters:

- __PLLSAI1M__: specifies the division factor for PLLSAI1 clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16.

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI1_DIVP_CONFIG

Description:

- Macro to configure the PLLSAI1 clock division factor P.

Parameters:

- __PLLSAI1P__: specifies the division factor for SAI clock. This parameter must be a number in the range (7 or 17) for STM32L47xxx/L48xxx else (2 to 31). Use to set SAI1 clock frequency = f(PLLSAI1) / PLLSAI1P

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI1_DIVQ_CONFIG

Description:

- Macro to configure the PLLSAI1 clock division factor Q.

Parameters:

- __PLLSAI1Q__: specifies the division factor for USB/RNG/SDMMC1 clock. This parameter must be in the range (2, 4, 6 or

8). Use to set USB/RNG/SDMMC1 clock frequency = f(PLLSAI1) / PLLSAI1Q

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI1_DIVR_CONFIG

Description:

- Macro to configure the PLLSAI1 clock division factor R.

Parameters:

- __PLLSAI1R__: specifies the division factor for ADC clock. This parameter must be in the range (2, 4, 6 or 8) Use to set ADC clock frequency = f(PLLSAI1) / PLLSAI1R

Return value:

- None

Notes:

- This function must be used only when the PLLSAI1 is disabled. PLLSAI1 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI1_ENABLE

Description:

- Macros to enable or disable the PLLSAI1.

Return value:

- None

Notes:

- The PLLSAI1 is disabled by hardware when entering STOP and STANDBY modes.

__HAL_RCC_PLLSAI1_DISABLE

__HAL_RCC_PLLSAI1CLKOUT_ENABLE

Description:

- Macros to enable or disable each clock output (PLLSAI1_SAI1, PLLSAI1_USB2 and PLLSAI1_ADC1).

Parameters:

- __PLLSAI1_CLOCKOUT__: specifies the PLLSAI1 clock to be output. This parameter

can be one or a combination of the following values:
- RCC_PLLSAI1_SAI1CLK This clock is used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
- RCC_PLLSAI1_48M2CLK This clock is used to generate the clock for the USB OTG FS (48 MHz), the random number generator (<=48 MHz) and the SDIO (<= 48 MHz).
- RCC_PLLSAI1_ADC1CLK Clock used to clock ADC peripheral.

Return value:

- None

Notes:

- Enabling and disabling those clocks can be done without the need to stop the PLL. This is mainly used to save Power.

__HAL_RCC_PLLSAI1CLKOUT_DISABLE

__HAL_RCC_GET_PLLSAI1CLKOUT_CONFIG

Description:

- Macro to get clock output enable status (PLLSAI1_SAI1, PLLSAI1_USB2 and PLLSAI1_ADC1).

Parameters:

- __PLLSAI1_CLOCKOUT__: specifies the PLLSAI1 clock to be output. This parameter can be one of the following values:
  - RCC_PLLSAI1_SAI1CLK This clock is used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
  - RCC_PLLSAI1_48M2CLK This clock is used to generate the clock for the USB OTG FS (48 MHz), the random number generator (<=48 MHz) and the SDIO (<= 48 MHz).
  - RCC_PLLSAI1_ADC1CLK Clock used to clock ADC peripheral.

Return value:

- SET: / RESET

__HAL_RCC_PLLSAI2_CONFIG

Description:

- Macro to configure the PLLSAI2 clock multiplication and division factors.

Parameters:

- __PLLSAI2M__: specifies the division

factor of PLLSAI2 input clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16.
- __PLLSAI2N__: specifies the multiplication factor for PLLSAI2 VCO output clock. This parameter must be a number between 8 and 86.
- __PLLSAI2P__: specifies the division factor for SAI clock. This parameter must be a number in the range (7 or 17) for STM32L47xxx/L48xxx else (2 to 31). SAI2 clock frequency = f(PLLSAI2) / PLLSAI2P
- __PLLSAI2Q__: specifies the division factor for DSI clock. This parameter must be in the range (2, 4, 6 or 8). DSI clock frequency = f(PLLSAI2) / PLLSAI2Q
- __PLLSAI2R__: specifies the division factor for SAR ADC clock. This parameter must be in the range (2, 4, 6 or 8).

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)
- You have to set the PLLSAI2N parameter correctly to ensure that the VCO output frequency is between 64 and 344 MHz.

__HAL_RCC_PLLSAI2_MULN_CONFIG

Description:

- Macro to configure the PLLSAI2 clock multiplication factor N.

Parameters:

- __PLLSAI2N__: specifies the multiplication factor for PLLSAI2 VCO output clock. This parameter must be a number between 8 and 86.

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)
- You have to set the PLLSAI2N parameter correctly to ensure that the VCO output frequency is between 64 and 344 MHz.

PLLSAI1 clock frequency = f(PLLSAI1) multiplied by PLLSAI2N

__HAL_RCC_PLLSAI2_DIVM_CONFIG

Description:

- Macro to configure the PLLSAI2 input clock division factor M.

Parameters:

- __PLLSAI2M__: specifies the division factor for PLLSAI2 clock. This parameter must be a number between Min_Data = 1 and Max_Data = 16.

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI2_DIVP_CONFIG

Description:

- Macro to configure the PLLSAI2 clock division factor P.

Parameters:

- __PLLSAI2P__: specifies the division factor. This parameter must be a number in the range (7 or 17). Use to set SAI2 clock frequency = f(PLLSAI2) / __PLLSAI2P__

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI2_DIVQ_CONFIG

Description:

- Macro to configure the PLLSAI2 clock division factor Q.

Parameters:

- __PLLSAI2Q__: specifies the division factor for USB/RNG/SDMMC1 clock. This parameter must be in the range (2, 4, 6 or 8). Use to set USB/RNG/SDMMC1 clock frequency = f(PLLSAI2) / PLLSAI2Q

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI2_DIVR_CONFIG

Description:

- Macro to configure the PLLSAI2 clock division factor R.

Parameters:

- __PLLSAI2R__: specifies the division factor. This parameter must be in the range (2, 4, 6 or 8). Use to set ADC clock frequency = f(PLLSAI2) / __PLLSAI2R__

Return value:

- None

Notes:

- This function must be used only when the PLLSAI2 is disabled. PLLSAI2 clock source is common with the main PLL (configured through __HAL_RCC_PLL_CONFIG() macro)

__HAL_RCC_PLLSAI2_ENABLE

Description:

- Macros to enable or disable the PLLSAI2.

Return value:

- None

Notes:

- The PLLSAI2 is disabled by hardware when entering STOP and STANDBY modes.

__HAL_RCC_PLLSAI2_DISABLE

__HAL_RCC_PLLSAI2CLKOUT_ENABLE

Description:

- Macros to enable or disable each clock output (PLLSAI2_SAI2, PLLSAI2_ADC2 and RCC_PLLSAI2_DSICLK).

Parameters:

- __PLLSAI2_CLOCKOUT__: specifies the PLLSAI2 clock to be output. This parameter can be one or a combination of the following values:
  – RCC_PLLSAI2_SAI2CLK This clock is

used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
  - RCC_PLLSAI2_DSICLK Clock used to clock DSI peripheral.

Return value:

- None

Notes:

- Enabling and disabling those clocks can be done without the need to stop the PLL. This is mainly used to save Power.

__HAL_RCC_PLLSAI2CLKOUT_DISABLE

__HAL_RCC_GET_PLLSAI2CLKOUT_CONFIG

Description:

- Macro to get clock output enable status (PLLSAI2_SAI2, PLLSAI2_ADC2 and RCC_PLLSAI2_DSICLK).

Parameters:

- __PLLSAI2_CLOCKOUT__: specifies the PLLSAI2 clock to be output. This parameter can be one of the following values:
  - RCC_PLLSAI2_SAI2CLK This clock is used to generate an accurate clock to achieve high-quality audio performance on SAI interface in case.
  - RCC_PLLSAI2_DSICLK Clock used to clock DSI peripheral.

Return value:

- SET: / RESET

__HAL_RCC_SAI1_CONFIG

Description:

- Macro to configure the SAI1 clock source.

Parameters:

- __SAI1_CLKSOURCE__: defines the SAI1 clock source. This clock is derived from the PLLSAI1, system PLL or external clock (through a dedicated pin). This parameter can be one of the following values:
  - RCC_SAI1CLKSOURCE_PLLSAI1 SAI1 clock = PLLSAI1 "P" clock (PLLSAI1CLK)
  - RCC_SAI1CLKSOURCE_PLL SAI1 clock = PLL "P" clock (PLLSAI3CLK if PLLSAI2 exists, else PLLSAI2CLK)
  - RCC_SAI1CLKSOURCE_PIN SAI1 clock = External Clock (SAI1_EXTCLK)
  - RCC_SAI1CLKSOURCE_HSI SAI1

clock = HSI16

Return value:

- None

__HAL_RCC_GET_SAI1_SOURCE

Description:

- Macro to get the SAI1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_SAI1CLKSOURCE_PLLSAI1 SAI1 clock = PLLSAI1 "P" clock (PLLSAI1CLK)
  - RCC_SAI1CLKSOURCE_PLL SAI1 clock = PLL "P" clock (PLLSAI3CLK if PLLSAI2 exists, else PLLSAI2CLK)
  - RCC_SAI1CLKSOURCE_PIN SAI1 clock = External Clock (SAI1_EXTCLK)

Notes:

- Despite returned values RCC_SAI1CLKSOURCE_PLLSAI1 or RCC_SAI1CLKSOURCE_PLL, HSI16 is automatically set as SAI1 clock source when PLLs are disabled for devices without PLLSAI2.

__HAL_RCC_SAI2_CONFIG

Description:

- Macro to configure the SAI2 clock source.

Parameters:

- __SAI2_CLKSOURCE__: defines the SAI2 clock source. This clock is derived from the PLLSAI2, system PLL or external clock (through a dedicated pin). This parameter can be one of the following values:
  - RCC_SAI2CLKSOURCE_PLLSAI1 SAI2 clock = PLLSAI1 "P" clock (PLLSAI1CLK)
  - RCC_SAI2CLKSOURCE_PLLSAI2 SAI2 clock = PLLSAI2 "P" clock (PLLSAI2CLK)
  - RCC_SAI2CLKSOURCE_PLL SAI2 clock = PLL "P" clock (PLLSAI3CLK)
  - RCC_SAI2CLKSOURCE_PIN SAI2 clock = External Clock (SAI2_EXTCLK)
  - RCC_SAI2CLKSOURCE_HSI SAI2 clock = HSI16

Return value:

- None

__HAL_RCC_GET_SAI2_SOURCE

Description:

- Macro to get the SAI2 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_SAI2CLKSOURCE_PLLSAI1 SAI2 clock = PLLSAI1 "P" clock (PLLSAI1CLK)
  - RCC_SAI2CLKSOURCE_PLLSAI2 SAI2 clock = PLLSAI2 "P" clock (PLLSAI2CLK)
  - RCC_SAI2CLKSOURCE_PLL SAI2 clock = PLL "P" clock (PLLSAI3CLK)
  - RCC_SAI2CLKSOURCE_PIN SAI2 clock = External Clock (SAI2_EXTCLK)

__HAL_RCC_I2C1_CONFIG

Description:

- Macro to configure the I2C1 clock (I2C1CLK).

Parameters:

- __I2C1_CLKSOURCE__: specifies the I2C1 clock source. This parameter can be one of the following values:
  - RCC_I2C1CLKSOURCE_PCLK1 PCLK1 selected as I2C1 clock
  - RCC_I2C1CLKSOURCE_HSI HSI selected as I2C1 clock
  - RCC_I2C1CLKSOURCE_SYSCLK System Clock selected as I2C1 clock

Return value:

- None

__HAL_RCC_GET_I2C1_SOURCE

Description:

- Macro to get the I2C1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_I2C1CLKSOURCE_PCLK1 PCLK1 selected as I2C1 clock
  - RCC_I2C1CLKSOURCE_HSI HSI selected as I2C1 clock
  - RCC_I2C1CLKSOURCE_SYSCLK System Clock selected as I2C1 clock

__HAL_RCC_I2C2_CONFIG

Description:

- Macro to configure the I2C2 clock (I2C2CLK).

Parameters:

- __I2C2_CLKSOURCE__: specifies the I2C2 clock source. This parameter can be one of the following values:
  - RCC_I2C2CLKSOURCE_PCLK1 PCLK1 selected as I2C2 clock
  - RCC_I2C2CLKSOURCE_HSI HSI selected as I2C2 clock
  - RCC_I2C2CLKSOURCE_SYSCLK System Clock selected as I2C2 clock

Return value:

- None

__HAL_RCC_GET_I2C2_SOURCE

Description:

- Macro to get the I2C2 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_I2C2CLKSOURCE_PCLK1 PCLK1 selected as I2C2 clock
  - RCC_I2C2CLKSOURCE_HSI HSI selected as I2C2 clock
  - RCC_I2C2CLKSOURCE_SYSCLK System Clock selected as I2C2 clock

__HAL_RCC_I2C3_CONFIG

Description:

- Macro to configure the I2C3 clock (I2C3CLK).

Parameters:

- __I2C3_CLKSOURCE__: specifies the I2C3 clock source. This parameter can be one of the following values:
  - RCC_I2C3CLKSOURCE_PCLK1 PCLK1 selected as I2C3 clock
  - RCC_I2C3CLKSOURCE_HSI HSI selected as I2C3 clock
  - RCC_I2C3CLKSOURCE_SYSCLK System Clock selected as I2C3 clock

Return value:

- None

__HAL_RCC_GET_I2C3_SOURCE

Description:

- Macro to get the I2C3 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_I2C3CLKSOURCE_PCLK1 PCLK1 selected as I2C3 clock
  - RCC_I2C3CLKSOURCE_HSI HSI selected as I2C3 clock

- RCC_I2C3CLKSOURCE_SYSCLK System Clock selected as I2C3 clock

__HAL_RCC_I2C4_CONFIG

Description:

- Macro to configure the I2C4 clock (I2C4CLK).

Parameters:

- __I2C4_CLKSOURCE__: specifies the I2C4 clock source. This parameter can be one of the following values:
  - RCC_I2C4CLKSOURCE_PCLK1 PCLK1 selected as I2C4 clock
  - RCC_I2C4CLKSOURCE_HSI HSI selected as I2C4 clock
  - RCC_I2C4CLKSOURCE_SYSCLK System Clock selected as I2C4 clock

Return value:

- None

__HAL_RCC_GET_I2C4_SOURCE

Description:

- Macro to get the I2C4 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_I2C4CLKSOURCE_PCLK1 PCLK1 selected as I2C4 clock
  - RCC_I2C4CLKSOURCE_HSI HSI selected as I2C4 clock
  - RCC_I2C4CLKSOURCE_SYSCLK System Clock selected as I2C4 clock

__HAL_RCC_USART1_CONFIG

Description:

- Macro to configure the USART1 clock (USART1CLK).

Parameters:

- __USART1_CLKSOURCE__: specifies the USART1 clock source. This parameter can be one of the following values:
  - RCC_USART1CLKSOURCE_PCLK2 PCLK2 selected as USART1 clock
  - RCC_USART1CLKSOURCE_HSI HSI selected as USART1 clock
  - RCC_USART1CLKSOURCE_SYSCLK System Clock selected as USART1 clock
  - RCC_USART1CLKSOURCE_LSE SE selected as USART1 clock

Return value:

- None

__HAL_RCC_GET_USART1_SOURCE

Description:

- Macro to get the USART1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_USART1CLKSOURCE_PCLK2 PCLK2 selected as USART1 clock
  - RCC_USART1CLKSOURCE_HSI HSI selected as USART1 clock
  - RCC_USART1CLKSOURCE_SYSCLK System Clock selected as USART1 clock
  - RCC_USART1CLKSOURCE_LSE LSE selected as USART1 clock

__HAL_RCC_USART2_CONFIG

Description:

- Macro to configure the USART2 clock (USART2CLK).

Parameters:

- __USART2_CLKSOURCE__: specifies the USART2 clock source. This parameter can be one of the following values:
  - RCC_USART2CLKSOURCE_PCLK1 PCLK1 selected as USART2 clock
  - RCC_USART2CLKSOURCE_HSI HSI selected as USART2 clock
  - RCC_USART2CLKSOURCE_SYSCLK System Clock selected as USART2 clock
  - RCC_USART2CLKSOURCE_LSE LSE selected as USART2 clock

Return value:

- None

__HAL_RCC_GET_USART2_SOURCE

Description:

- Macro to get the USART2 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_USART2CLKSOURCE_PCLK1 PCLK1 selected as USART2 clock
  - RCC_USART2CLKSOURCE_HSI HSI selected as USART2 clock
  - RCC_USART2CLKSOURCE_SYSCLK System Clock selected as USART2 clock

– RCC_USART2CLKSOURCE_LSE LSE selected as USART2 clock

__HAL_RCC_USART3_CONFIG

Description:

- Macro to configure the USART3 clock (USART3CLK).

Parameters:

- __USART3_CLKSOURCE__: specifies the USART3 clock source. This parameter can be one of the following values:
  – RCC_USART3CLKSOURCE_PCLK1 PCLK1 selected as USART3 clock
  – RCC_USART3CLKSOURCE_HSI HSI selected as USART3 clock
  – RCC_USART3CLKSOURCE_SYSCLK System Clock selected as USART3 clock
  – RCC_USART3CLKSOURCE_LSE LSE selected as USART3 clock

Return value:

- None

__HAL_RCC_GET_USART3_SOURCE

Description:

- Macro to get the USART3 clock source.

Return value:

- The: clock source can be one of the following values:
  – RCC_USART3CLKSOURCE_PCLK1 PCLK1 selected as USART3 clock
  – RCC_USART3CLKSOURCE_HSI HSI selected as USART3 clock
  – RCC_USART3CLKSOURCE_SYSCLK System Clock selected as USART3 clock
  – RCC_USART3CLKSOURCE_LSE LSE selected as USART3 clock

__HAL_RCC_UART4_CONFIG

Description:

- Macro to configure the UART4 clock (UART4CLK).

Parameters:

- __UART4_CLKSOURCE__: specifies the UART4 clock source. This parameter can be one of the following values:
  – RCC_UART4CLKSOURCE_PCLK1 PCLK1 selected as UART4 clock
  – RCC_UART4CLKSOURCE_HSI HSI selected as UART4 clock
  – RCC_UART4CLKSOURCE_SYSCLK System Clock selected as UART4

clock
  - RCC_UART4CLKSOURCE_LSE LSE selected as UART4 clock

Return value:

- None

__HAL_RCC_GET_UART4_SOURCE

Description:

- Macro to get the UART4 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_UART4CLKSOURCE_PCLK1 PCLK1 selected as UART4 clock
  - RCC_UART4CLKSOURCE_HSI HSI selected as UART4 clock
  - RCC_UART4CLKSOURCE_SYSCLK System Clock selected as UART4 clock
  - RCC_UART4CLKSOURCE_LSE LSE selected as UART4 clock

__HAL_RCC_UART5_CONFIG

Description:

- Macro to configure the UART5 clock (UART5CLK).

Parameters:

- __UART5_CLKSOURCE__: specifies the UART5 clock source. This parameter can be one of the following values:
  - RCC_UART5CLKSOURCE_PCLK1 PCLK1 selected as UART5 clock
  - RCC_UART5CLKSOURCE_HSI HSI selected as UART5 clock
  - RCC_UART5CLKSOURCE_SYSCLK System Clock selected as UART5 clock
  - RCC_UART5CLKSOURCE_LSE LSE selected as UART5 clock

Return value:

- None

__HAL_RCC_GET_UART5_SOURCE

Description:

- Macro to get the UART5 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_UART5CLKSOURCE_PCLK1 PCLK1 selected as UART5 clock
  - RCC_UART5CLKSOURCE_HSI HSI selected as UART5 clock

- RCC_UART5CLKSOURCE_SYSCLK System Clock selected as UART5 clock
- RCC_UART5CLKSOURCE_LSE LSE selected as UART5 clock

__HAL_RCC_LPUART1_CONFIG

Description:

- Macro to configure the LPUART1 clock (LPUART1CLK).

Parameters:

- __LPUART1_CLKSOURCE__: specifies the LPUART1 clock source. This parameter can be one of the following values:
  - RCC_LPUART1CLKSOURCE_PCLK1 PCLK1 selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_HSI HSI selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_SYSCLK System Clock selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_LSE LSE selected as LPUART1 clock

Return value:

- None

__HAL_RCC_GET_LPUART1_SOURCE

Description:

- Macro to get the LPUART1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_LPUART1CLKSOURCE_PCLK1 PCLK1 selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_HSI HSI selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_SYSCLK System Clock selected as LPUART1 clock
  - RCC_LPUART1CLKSOURCE_LSE LSE selected as LPUART1 clock

__HAL_RCC_LPTIM1_CONFIG

Description:

- Macro to configure the LPTIM1 clock (LPTIM1CLK).

Parameters:

- __LPTIM1_CLKSOURCE__: specifies the LPTIM1 clock source. This parameter can be one of the following values:
  - RCC_LPTIM1CLKSOURCE_PCLK1 PCLK1 selected as LPTIM1 clock
  - RCC_LPTIM1CLKSOURCE_LSI HSI

selected as LPTIM1 clock
- RCC_LPTIM1CLKSOURCE_HSI LSI selected as LPTIM1 clock
- RCC_LPTIM1CLKSOURCE_LSE LSE selected as LPTIM1 clock

Return value:

- None

__HAL_RCC_GET_LPTIM1_SOURCE

Description:

- Macro to get the LPTIM1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_LPTIM1CLKSOURCE_PCLK1 PCLK1 selected as LPUART1 clock
  - RCC_LPTIM1CLKSOURCE_LSI HSI selected as LPUART1 clock
  - RCC_LPTIM1CLKSOURCE_HSI System Clock selected as LPUART1 clock
  - RCC_LPTIM1CLKSOURCE_LSE LSE selected as LPUART1 clock

__HAL_RCC_LPTIM2_CONFIG

Description:

- Macro to configure the LPTIM2 clock (LPTIM2CLK).

Parameters:

- __LPTIM2_CLKSOURCE__: specifies the LPTIM2 clock source. This parameter can be one of the following values:
  - RCC_LPTIM2CLKSOURCE_PCLK1 PCLK1 selected as LPTIM2 clock
  - RCC_LPTIM2CLKSOURCE_LSI HSI selected as LPTIM2 clock
  - RCC_LPTIM2CLKSOURCE_HSI LSI selected as LPTIM2 clock
  - RCC_LPTIM2CLKSOURCE_LSE LSE selected as LPTIM2 clock

Return value:

- None

__HAL_RCC_GET_LPTIM2_SOURCE

Description:

- Macro to get the LPTIM2 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_LPTIM2CLKSOURCE_PCLK1 PCLK1 selected as LPUART1 clock
  - RCC_LPTIM2CLKSOURCE_LSI HSI

selected as LPUART1 clock
- RCC_LPTIM2CLKSOURCE_HSI System Clock selected as LPUART1 clock
- RCC_LPTIM2CLKSOURCE_LSE LSE selected as LPUART1 clock

__HAL_RCC_SDMMC1_CONFIG

Description:

- Macro to configure the SDMMC1 clock.

Parameters:

- __SDMMC1_CLKSOURCE__: specifies the SDMMC1 clock source. This parameter can be one of the following values:
  - RCC_SDMMC1CLKSOURCE_HSI48 HSI48 selected as SDMMC1 clock for devices with HSI48
  - RCC_SDMMC1CLKSOURCE_MSI MSI selected as SDMMC1 clock
  - RCC_SDMMC1CLKSOURCE_PLLSAI1 PLLSAI1 Clock selected as SDMMC1 clock
  - RCC_SDMMC1CLKSOURCE_PLL PLL Clock selected as SDMMC1 clock

Return value:

- None

__HAL_RCC_GET_SDMMC1_SOURCE

Description:

- Macro to get the SDMMC1 clock.

Return value:

- The: clock source can be one of the following values:
  - RCC_SDMMC1CLKSOURCE_HSI48 HSI48 selected as SDMMC1 clock for devices with HSI48
  - RCC_SDMMC1CLKSOURCE_MSI MSI selected as SDMMC1 clock
  - RCC_SDMMC1CLKSOURCE_PLLSAI1 PLLSAI1 "Q" clock (PLL48M2CLK) selected as SDMMC1 clock
  - RCC_SDMMC1CLKSOURCE_PLL PLL "Q" clock (PLL48M1CLK) selected as SDMMC1 clock

__HAL_RCC_RNG_CONFIG

Description:

- Macro to configure the RNG clock.

Parameters:

- __RNG_CLKSOURCE__: specifies the RNG clock source. This parameter can be one of the following values:
  - RCC_RNGCLKSOURCE_MSI MSI

selected as RNG clock
- RCC_RNGCLKSOURCE_PLLSAI1 PLLSAI1 Clock selected as RNG clock
- RCC_RNGCLKSOURCE_PLL PLL Clock selected as RNG clock

Return value:

- None

Notes:

- USB, RNG and SDMMC1 peripherals share the same 48MHz clock source.

__HAL_RCC_GET_RNG_SOURCE

Description:

- Macro to get the RNG clock.

Return value:

- The: clock source can be one of the following values:
  - RCC_RNGCLKSOURCE_MSI MSI selected as RNG clock
  - RCC_RNGCLKSOURCE_PLLSAI1 PLLSAI1 "Q" clock (PLL48M2CLK) selected as RNG clock
  - RCC_RNGCLKSOURCE_PLL PLL "Q" clock (PLL48M1CLK) selected as RNG clock

__HAL_RCC_USB_CONFIG

Description:

- Macro to configure the USB clock (USBCLK).

Parameters:

- __USB_CLKSOURCE__: specifies the USB clock source. This parameter can be one of the following values:
  - RCC_USBCLKSOURCE_MSI MSI selected as USB clock
  - RCC_USBCLKSOURCE_PLLSAI1 PLLSAI1 "Q" clock (PLL48M2CLK) selected as USB clock
  - RCC_USBCLKSOURCE_PLL PLL "Q" clock (PLL48M1CLK) selected as USB clock

Return value:

- None

Notes:

- USB, RNG and SDMMC1 peripherals share the same 48MHz clock source.

__HAL_RCC_GET_USB_SOURCE

Description:

- Macro to get the USB clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_USBCLKSOURCE_MSI MSI selected as USB clock
  - RCC_USBCLKSOURCE_PLLSAI1 PLLSAI1 "Q" clock (PLL48M2CLK) selected as USB clock
  - RCC_USBCLKSOURCE_PLL PLL "Q" clock (PLL48M1CLK) selected as USB clock

__HAL_RCC_ADC_CONFIG

Description:

- Macro to configure the ADC interface clock.

Parameters:

- __ADC_CLKSOURCE__: specifies the ADC digital interface clock source. This parameter can be one of the following values:
  - RCC_ADCCLKSOURCE_NONE No clock selected as ADC clock
  - RCC_ADCCLKSOURCE_PLLSAI1 PLLSAI1 Clock selected as ADC clock
  - RCC_ADCCLKSOURCE_SYSCLK System Clock selected as ADC clock

Return value:

- None

__HAL_RCC_GET_ADC_SOURCE

Description:

- Macro to get the ADC clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_ADCCLKSOURCE_NONE No clock selected as ADC clock
  - RCC_ADCCLKSOURCE_PLLSAI1 PLLSAI1 Clock selected as ADC clock
  - RCC_ADCCLKSOURCE_SYSCLK System Clock selected as ADC clock

__HAL_RCC_DFSDM1_CONFIG

Description:

- Macro to configure the DFSDM1 clock.

Parameters:

- __DFSDM1_CLKSOURCE__: specifies the DFSDM1 clock source. This parameter can be one of the following values:
  - RCC_DFSDM1CLKSOURCE_PCLK2 PCLK2 Clock selected as DFSDM1 clock
  - RCC_DFSDM1CLKSOURCE_SYSCL

K System Clock selected as DFSDM1 clock

Return value:

- None

__HAL_RCC_GET_DFSDM1_SOURCE

Description:

- Macro to get the DFSDM1 clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_DFSDM1CLKSOURCE_PCLK2 PCLK2 Clock selected as DFSDM1 clock
  - RCC_DFSDM1CLKSOURCE_SYSCLK System Clock selected as DFSDM1 clock

__HAL_RCC_DFSDM1AUDIO_CONFIG

Description:

- Macro to configure the DFSDM1 audio clock.

Parameters:

- __DFSDM1AUDIO_CLKSOURCE__: specifies the DFSDM1 audio clock source. This parameter can be one of the following values:
  - RCC_DFSDM1AUDIOCLKSOURCE_SAI1 SAI1 clock selected as DFSDM1 audio clock
  - RCC_DFSDM1AUDIOCLKSOURCE_HSI HSI clock selected as DFSDM1 audio clock
  - RCC_DFSDM1AUDIOCLKSOURCE_MSI MSI clock selected as DFSDM1 audio clock

Return value:

- None

__HAL_RCC_GET_DFSDM1AUDIO_SOURCE

Description:

- Macro to get the DFSDM1 audio clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_DFSDM1AUDIOCLKSOURCE_SAI1 SAI1 clock selected as DFSDM1 audio clock
  - RCC_DFSDM1AUDIOCLKSOURCE_HSI HSI clock selected as DFSDM1 audio clock

– RCC_DFSDM1AUDIOCLKSOURCE_MSI MSI clock selected as DFSDM1 audio clock

__HAL_RCC_LTDC_CONFIG

Description:

- Macro to configure the LTDC clock.

Parameters:

- __LTDC_CLKSOURCE__: specifies the DSI clock source. This parameter can be one of the following values:
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV2 PLLSAI2 divider R divided by 2 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV4 PLLSAI2 divider R divided by 4 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV8 PLLSAI2 divider R divided by 8 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV16 PLLSAI2 divider R divided by 16 clock selected as LTDC clock

Return value:

- None

__HAL_RCC_GET_LTDC_SOURCE

Description:

- Macro to get the LTDC clock source.

Return value:

- The: clock source can be one of the following values:
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV2 PLLSAI2 divider R divided by 2 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV4 PLLSAI2 divider R divided by 4 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV8 PLLSAI2 divider R divided by 8 clock selected as LTDC clock
  – RCC_LTDCCLKSOURCE_PLLSAI2_DIV16 PLLSAI2 divider R divided by 16 clock selected as LTDC clock

__HAL_RCC_DSI_CONFIG

Description:

- Macro to configure the DSI clock.

Parameters:

- __DSI_CLKSOURCE__: specifies the DSI clock source. This parameter can be one of the following values:
  – RCC_DSICLKSOURCE_DSIPHY DSI-

PHY clock selected as DSI clock
  - RCC_DSICLKSOURCE_PLLSAI2 PLLSAI2 R divider clock selected as DSI clock

Return value:

- None

__HAL_RCC_GET_DSI_SOURCE

Description:

- Macro to get the DSI clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_DSICLKSOURCE_DSIPHY DSI-PHY clock selected as DSI clock
  - RCC_DSICLKSOURCE_PLLSAI2 PLLSAI2 R divider clock selected as DSI clock

__HAL_RCC_OSPI_CONFIG

Description:

- Macro to configure the OctoSPI clock.

Parameters:

- __OSPI_CLKSOURCE__: specifies the OctoSPI clock source. This parameter can be one of the following values:
  - RCC_OSPICLKSOURCE_SYSCLK System Clock selected as OctoSPI clock
  - RCC_OSPICLKSOURCE_MSI MSI clock selected as OctoSPI clock
  - RCC_OSPICLKSOURCE_PLL PLL Q divider clock selected as OctoSPI clock

Return value:

- None

__HAL_RCC_GET_OSPI_SOURCE

Description:

- Macro to get the OctoSPI clock source.

Return value:

- The: clock source can be one of the following values:
  - RCC_OSPICLKSOURCE_SYSCLK System Clock selected as OctoSPI clock
  - RCC_OSPICLKSOURCE_MSI MSI clock selected as OctoSPI clock
  - RCC_OSPICLKSOURCE_PLL PLL Q divider clock selected as OctoSPI clock

RCC LSE CSS external interrupt line

RCC_EXTI_LINE_LSECSS External interrupt line 19 connected to the LSE CSS EXTI Line

Flags Interrupts Management

__HAL_RCC_PLLSAI1_ENABLE_IT

Description:

- Enable PLLSAI1RDY interrupt.

Return value:

- None

__HAL_RCC_PLLSAI1_DISABLE_IT

Description:

- Disable PLLSAI1RDY interrupt.

Return value:

- None

__HAL_RCC_PLLSAI1_CLEAR_IT

Description:

- Clear the PLLSAI1RDY interrupt pending bit.

Return value:

- None

__HAL_RCC_PLLSAI1_GET_IT_SOURCE

Description:

- Check whether PLLSAI1RDY interrupt has occurred or not.

Return value:

- TRUE: or FALSE.

__HAL_RCC_PLLSAI1_GET_FLAG

Description:

- Check whether the PLLSAI1RDY flag is set or not.

Return value:

- TRUE: or FALSE.

__HAL_RCC_PLLSAI2_ENABLE_IT

Description:

- Enable PLLSAI2RDY interrupt.

Return value:

- None

__HAL_RCC_PLLSAI2_DISABLE_IT

Description:

- Disable PLLSAI2RDY interrupt.

Return value:

- None

__HAL_RCC_PLLSAI2_CLEAR_IT

Description:

- Clear the PLLSAI2RDY interrupt

	pending bit. **Return value:** • None
__HAL_RCC_PLLSAI2_GET_IT_SOURCE	**Description:** • Check whether the PLLSAI2RDY interrupt has occurred or not. **Return value:** • TRUE: or FALSE.
__HAL_RCC_PLLSAI2_GET_FLAG	**Description:** • Check whether the PLLSAI2RDY flag is set or not. **Return value:** • TRUE: or FALSE.
__HAL_RCC_LSECSS_EXTI_ENABLE_IT	**Description:** • Enable the RCC LSE CSS Extended Interrupt Line. **Return value:** • None
__HAL_RCC_LSECSS_EXTI_DISABLE_IT	**Description:** • Disable the RCC LSE CSS Extended Interrupt Line. **Return value:** • None
__HAL_RCC_LSECSS_EXTI_ENABLE_EVENT	**Description:** • Enable the RCC LSE CSS Event Line. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_DISABLE_EVENT	**Description:** • Disable the RCC LSE CSS Event Line. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_ENABLE_FALLING_EDGE	**Description:** • Enable the RCC LSE CSS Extended Interrupt Falling Trigger. **Return value:** • None.

__HAL_RCC_LSECSS_EXTI_DISABLE_ FALLING_EDGE	**Description:** • Disable the RCC LSE CSS Extended Interrupt Falling Trigger. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_ENABLE_ RISING_ EDGE	**Description:** • Enable the RCC LSE CSS Extended Interrupt Rising Trigger. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_DISABLE_ RISING_ EDGE	**Description:** • Disable the RCC LSE CSS Extended Interrupt Rising Trigger. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_ENABLE_ RISING_ FALLING_EDGE	**Description:** • Enable the RCC LSE CSS Extended Interrupt Rising & Falling Trigger. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_DISABLE_ RISING_ FALLING_EDGE	**Description:** • Disable the RCC LSE CSS Extended Interrupt Rising & Falling Trigger. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_GET_FLAG	**Description:** • Check whether the specified RCC LSE CSS EXTI interrupt flag is set or not. **Return value:** • EXTI: RCC LSE CSS Line Status.
__HAL_RCC_LSECSS_EXTI_CLEAR_FLAG	**Description:** • Clear the RCC LSE CSS EXTI flag. **Return value:** • None.
__HAL_RCC_LSECSS_EXTI_GENERATE_ SWIT	**Description:** • Generate a Software interrupt on the RCC LSE CSS EXTI line.

Return value:

- None.

__HAL_RCC_CRS_ENABLE_IT

Description:

- Enable the specified CRS interrupts.

Parameters:

- __INTERRUPT__: specifies the CRS interrupt sources to be enabled. This parameter can be any combination of the following values:
  - RCC_CRS_IT_SYNCOK SYNC event OK interrupt
  - RCC_CRS_IT_SYNCWARN SYNC warning interrupt
  - RCC_CRS_IT_ERR Synchronization or trimming error interrupt
  - RCC_CRS_IT_ESYNC Expected SYNC interrupt

Return value:

- None

__HAL_RCC_CRS_DISABLE_IT

Description:

- Disable the specified CRS interrupts.

Parameters:

- __INTERRUPT__: specifies the CRS interrupt sources to be disabled. This parameter can be any combination of the following values:
  - RCC_CRS_IT_SYNCOK SYNC event OK interrupt
  - RCC_CRS_IT_SYNCWARN SYNC warning interrupt
  - RCC_CRS_IT_ERR Synchronization or trimming error interrupt
  - RCC_CRS_IT_ESYNC Expected SYNC interrupt

Return value:

- None

__HAL_RCC_CRS_GET_IT_SOURCE

Description:

- Check whether the CRS interrupt has occurred or not.

Parameters:

- __INTERRUPT__: specifies the CRS interrupt source to check. This parameter can be one of the following values:

- RCC_CRS_IT_SYNCOK SYNC event OK interrupt
- RCC_CRS_IT_SYNCWARN SYNC warning interrupt
- RCC_CRS_IT_ERR Synchronization or trimming error interrupt
- RCC_CRS_IT_ESYNC Expected SYNC interrupt

Return value:

- The: new state of __INTERRUPT__ (SET or RESET).

RCC_CRS_IT_ERROR_MASK

Description:

- Clear the CRS interrupt pending bits.

Parameters:

- __INTERRUPT__: specifies the interrupt pending bit to clear. This parameter can be any combination of the following values:
  - RCC_CRS_IT_SYNCOK SYNC event OK interrupt
  - RCC_CRS_IT_SYNCWARN SYNC warning interrupt
  - RCC_CRS_IT_ERR Synchronization or trimming error interrupt
  - RCC_CRS_IT_ESYNC Expected SYNC interrupt
  - RCC_CRS_IT_TRIMOVF Trimming overflow or underflow interrupt
  - RCC_CRS_IT_SYNCERR SYNC error interrupt
  - RCC_CRS_IT_SYNCMISS SYNC missed interrupt

__HAL_RCC_CRS_CLEAR_IT

__HAL_RCC_CRS_GET_FLAG

Description:

- Check whether the specified CRS flag is set or not.

Parameters:

- __FLAG__: specifies the flag to check. This parameter can be one of the following values:
  - RCC_CRS_FLAG_SYNCOK SYNC event OK
  - RCC_CRS_FLAG_SYNCWARN SYNC warning
  - RCC_CRS_FLAG_ERR Error
  - RCC_CRS_FLAG_ESYNC

Expected SYNC
- RCC_CRS_FLAG_TRIMOVF Trimming overflow or underflow
- RCC_CRS_FLAG_SYNCERR SYNC error
- RCC_CRS_FLAG_SYNCMISS SYNC missed

Return value:

- The: new state of _FLAG_ (TRUE or FALSE).

RCC_CRS_FLAG_ERROR_MASK

Description:

- Clear the CRS specified FLAG.

Parameters:

- __FLAG__: specifies the flag to clear. This parameter can be one of the following values:
  - RCC_CRS_FLAG_SYNCOK SYNC event OK
  - RCC_CRS_FLAG_SYNCWARN SYNC warning
  - RCC_CRS_FLAG_ERR Error
  - RCC_CRS_FLAG_ESYNC Expected SYNC
  - RCC_CRS_FLAG_TRIMOVF Trimming overflow or underflow
  - RCC_CRS_FLAG_SYNCERR SYNC error
  - RCC_CRS_FLAG_SYNCMISS SYNC missed

Return value:

- None

Notes:

- RCC_CRS_FLAG_ERR clears RCC_CRS_FLAG_TRIMOVF, RCC_CRS_FLAG_SYNCERR, RCC_CRS_FLAG_SYNCMISS and consequently RCC_CRS_FLAG_ERR

__HAL_RCC_CRS_CLEAR_FLAG

I2C1 Clock Source

RCC_I2C1CLKSOURCE_PCLK1

RCC_I2C1CLKSOURCE_SYSCLK

RCC_I2C1CLKSOURCE_HSI

I2C2 Clock Source

RCC_I2C2CLKSOURCE_PCLK1

RCC_I2C2CLKSOURCE_SYSCLK

RCC_I2C2CLKSOURCE_HSI

I2C3 Clock Source

RCC_I2C3CLKSOURCE_PCLK1

RCC_I2C3CLKSOURCE_SYSCLK

RCC_I2C3CLKSOURCE_HSI

I2C4 Clock Source

RCC_I2C4CLKSOURCE_PCLK1

RCC_I2C4CLKSOURCE_SYSCLK

RCC_I2C4CLKSOURCE_HSI

LPTIM1 Clock Source

RCC_LPTIM1CLKSOURCE_PCLK1

RCC_LPTIM1CLKSOURCE_LSI

RCC_LPTIM1CLKSOURCE_HSI

RCC_LPTIM1CLKSOURCE_LSE

LPTIM2 Clock Source

RCC_LPTIM2CLKSOURCE_PCLK1

RCC_LPTIM2CLKSOURCE_LSI

RCC_LPTIM2CLKSOURCE_HSI

RCC_LPTIM2CLKSOURCE_LSE

LPUART1 Clock Source

RCC_LPUART1CLKSOURCE_PCLK1

RCC_LPUART1CLKSOURCE_SYSCLK

RCC_LPUART1CLKSOURCE_HSI

RCC_LPUART1CLKSOURCE_LSE

Low Speed Clock Source

RCC_LSCOSOURCE_LSI LSI selection for low speed clock output

RCC_LSCOSOURCE_LSE LSE selection for low speed clock output

LTDC Clock Source

RCC_LTDCCLKSOURCE_PLLSAI2_DIV2

RCC_LTDCCLKSOURCE_PLLSAI2_DIV4

RCC_LTDCCLKSOURCE_PLLSAI2_DIV8

RCC_LTDCCLKSOURCE_PLLSAI2_DIV16

OctoSPI Clock Source

RCC_OSPICLKSOURCE_SYSCLK

RCC_OSPICLKSOURCE_MSI

RCC_OSPICLKSOURCE_PLL

Periph Clock Selection

RCC_PERIPHCLK_USART1

RCC_PERIPHCLK_USART2

RCC_PERIPHCLK_USART3

RCC_PERIPHCLK_UART4

RCC_PERIPHCLK_UART5

RCC_PERIPHCLK_LPUART1

RCC_PERIPHCLK_I2C1

RCC_PERIPHCLK_I2C2

RCC_PERIPHCLK_I2C3

RCC_PERIPHCLK_LPTIM1

RCC_PERIPHCLK_LPTIM2

RCC_PERIPHCLK_SAI1

RCC_PERIPHCLK_SAI2

RCC_PERIPHCLK_USB

RCC_PERIPHCLK_ADC

RCC_PERIPHCLK_DFSDM1

RCC_PERIPHCLK_DFSDM1AUDIO

RCC_PERIPHCLK_RTC

RCC_PERIPHCLK_RNG

RCC_PERIPHCLK_SDMMC1

RCC_PERIPHCLK_I2C4

RCC_PERIPHCLK_LTDC

RCC_PERIPHCLK_DSI

RCC_PERIPHCLK_OSPI

RNG Clock Source

RCC_RNGCLKSOURCE_HSI48

RCC_RNGCLKSOURCE_PLLSAI1

RCC_RNGCLKSOURCE_PLL

RCC_RNGCLKSOURCE_MSI

SAI1 Clock Source

RCC_SAI1CLKSOURCE_PLLSAI1

RCC_SAI1CLKSOURCE_PLLSAI2

RCC_SAI1CLKSOURCE_PLL

RCC_SAI1CLKSOURCE_PIN

RCC_SAI1CLKSOURCE_HSI

SAI2 Clock Source

RCC_SAI2CLKSOURCE_PLLSAI1

RCC_SAI2CLKSOURCE_PLLSAI2

RCC_SAI2CLKSOURCE_PLL

RCC_SAI2CLKSOURCE_PIN

RCC_SAI2CLKSOURCE_HSI

SDMMC1 Clock Source

RCC_SDMMC1CLKSOURCE_HSI48

RCC_SDMMC1CLKSOURCE_PLLSAI1

RCC_SDMMC1CLKSOURCE_PLL

RCC_SDMMC1CLKSOURCE_MSI

UART4 Clock Source

RCC_UART4CLKSOURCE_PCLK1

RCC_UART4CLKSOURCE_SYSCLK

RCC_UART4CLKSOURCE_HSI

RCC_UART4CLKSOURCE_LSE

UART5 Clock Source

RCC_UART5CLKSOURCE_PCLK1

RCC_UART5CLKSOURCE_SYSCLK

RCC_UART5CLKSOURCE_HSI

RCC_UART5CLKSOURCE_LSE

USART1 Clock Source

RCC_USART1CLKSOURCE_PCLK2

RCC_USART1CLKSOURCE_SYSCLK

RCC_USART1CLKSOURCE_HSI

RCC_USART1CLKSOURCE_LSE

USART2 Clock Source

RCC_USART2CLKSOURCE_PCLK1

RCC_USART2CLKSOURCE_SYSCLK

RCC_USART2CLKSOURCE_HSI

RCC_USART2CLKSOURCE_LSE

USART3 Clock Source

RCC_USART3CLKSOURCE_PCLK1

RCC_USART3CLKSOURCE_SYSCLK

RCC_USART3CLKSOURCE_HSI

RCC_USART3CLKSOURCE_LSE

USB Clock Source

RCC_USBCLKSOURCE_HSI48

RCC_USBCLKSOURCE_PLLSAI1

RCC_USBCLKSOURCE_PLL

RCC_USBCLKSOURCE_MSI

53 HAL RNG Generic Driver

53.1 RNG Firmware driver registers structures

53.1.1 RNG_InitTypeDef

Data Fields

- ***uint32_t ClockErrorDetection***

Field Documentation

- ***uint32_t RNG_InitTypeDef::ClockErrorDetection***
  Clock error detection

53.1.2 NG_HandleTypeDef

Data Fields

- ***RNG_TypeDef * Instance***
- ***RNG_InitTypeDef Init***
- ***HAL_LockTypeDef Lock***
- ***__IO HAL_RNG_StateTypeDef State***
- ***uint32_t RandomNumber***

Field Documentation

- ***RNG_TypeDef* RNG_HandleTypeDef::Instance***
  Register base address
- ***RNG_InitTypeDef RNG_HandleTypeDef::Init***
  RNG configuration parameters
- ***HAL_LockTypeDef RNG_HandleTypeDef::Lock***
  RNG locking object
- ***__IO HAL_RNG_StateTypeDef RNG_HandleTypeDef::State***
  RNG communication state
- ***uint32_t RNG_HandleTypeDef::RandomNumber***
  Last Generated RNG Data

53.2 RNG Firmware driver API description

53.2.1 How to use this driver

The RNG HAL driver can be used as follows:

1. Enable the RNG controller clock using __HAL_RCC_RNG_CLK_ENABLE() macro in HAL_RNG_MspInit().
2. Activate the RNG peripheral using HAL_RNG_Init() function.
3. Wait until the 32-bit Random Number Generator contains a valid random data using (polling/interrupt) mode.
4. Get the 32 bit random number using HAL_RNG_GenerateRandomNumber() function.

53.2.2 Initialization and de-initialization functions

This section provides functions allowing to:

- Initialize the RNG according to the specified parameters in the RNG_InitTypeDef and create the associated handle