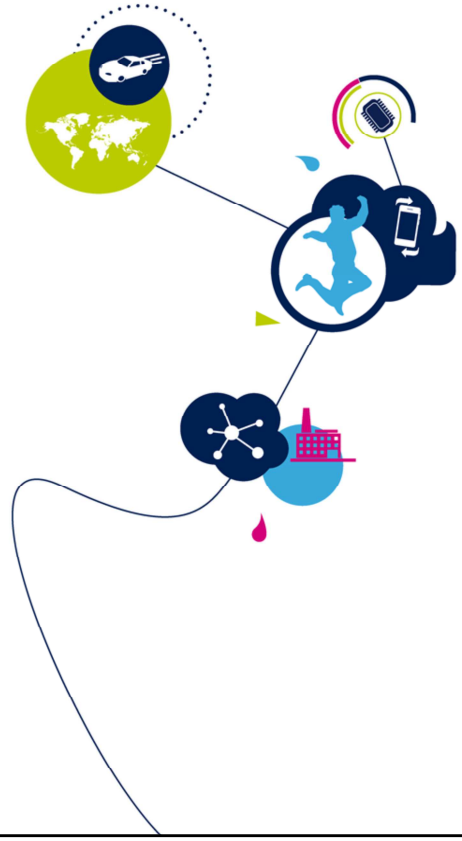
STM32H7x3/x5/x7 - RCC

Reset and Clock Controller

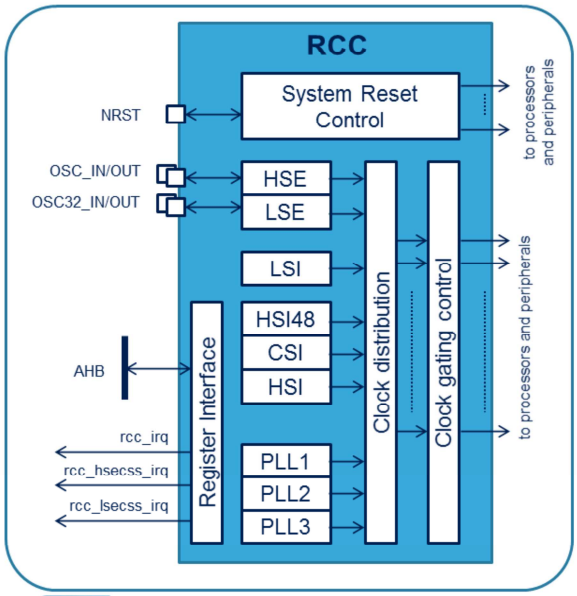
Revision 1.0



life.augmented

Hello, and welcome to this presentation of the Reset and Clock Controller (RCC) for STM32H7x3/x5/x7microcontrollers.
The RCC is linked to the Power Controller block (PWR).
Before going through this slide set, please have a look at the slides concerning the PWR block.

Overview



- The STM32H7x3/x5/x7 Reset and Clock Controller (RCC) manages:
  - The generation of all the clocks,
    - PLLs, RC oscillators, Crystal oscillators…
  - The gating of all the clocks,
  - The control of all the system and peripheral resets.

Application benefits

- High flexibility in choice of clock sources to meet consumption and accuracy requirements.
- Safe and flexible reset management

The STM32H7x3/x5/x7 reset and clock controller manages the reset, the system and peripheral clocks generation.
STM32H7x3/x5/x7 devices embed 4 internal oscillators, 2 oscillators for an external crystal or resonator, and 3 phase-locked loops (PLL).
Many peripherals have their own clock, independent of the system clock.
The STM32H7x3/x5/x7 RCC provides high flexibility in the choice of clock sources, which allows the system designer to meet both power consumption and accuracy requirements.
The numerous independent peripheral clocks allow a designer to adjust the system power consumption without impacting the communication baud rates, and also to keep certain peripherals active in low-power mode.

Reset key features

Safe and flexible reset management without external components

- The RCC generates several types of reset:
  - Power-on reset (rst_por)
  - System reset (nreset)
  - Backup domain reset (rst_vsw)
  - Domain resets (rst_d1, rst_d2)

life.augmented

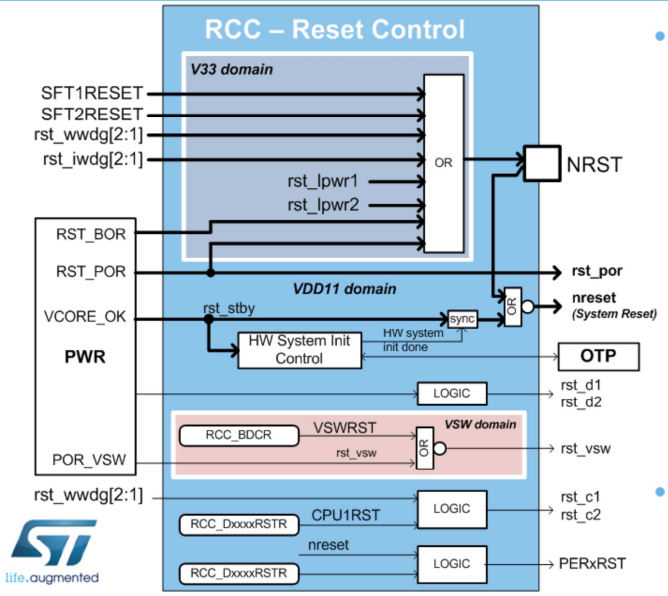
Safe and flexible reset management without any need for external components reduces application costs.
The RCC manages several types of resets: the power reset, the system reset, D1 and D2 domain reset, the local resets, and the backup domain reset.

Reset sources

4

No external components are needed due to internal filter and power monitoring

System reset sources can reset external components



- System reset sources:
  - Power-on reset (rst_por)
  - Brown-out reset
  - Low level on the NRST pin (external reset)
  - WWDG[2:1] timeout events
  - IWDG[2:1]timeout events
  - A software reset (SFT[2:1]RESET)
  - Low-power mode security reset (rst_lpwr[2:1])
  - Exit from Standby
- Power-on reset source (rst_por)
  - Voltage comparators on VDD and VDD11

Thanks to the voltage monitoring function included in the PWR block, the filters embedded in the NRST pad, and the RCC reset control, the amount of external components is reduced to a single external capacitor connected on the NRST pin.

The first type of reset is the System reset, which resets all the registers except certain registers for the Reset and Clock Controller and Power Controller. It also does not reset the Backup domain.

Many sources can generate a system reset:

- An invalid voltage on the V_{DD} or V_{CORE} supply (see PWR block for details),
- An invalid voltage on V_{DD} due to the brown-out function. The brown-out function allows the user to choose its own threshold levels for the V_{DD} supply

(see PWR block for details),

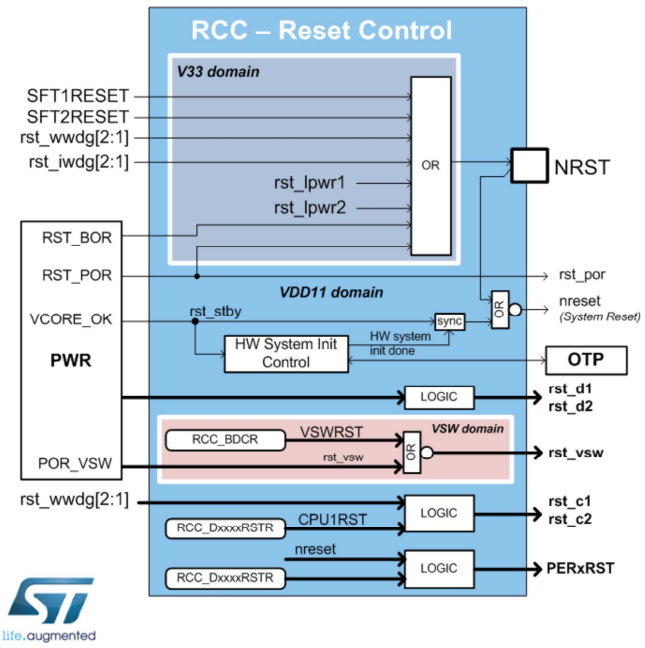
- An exit from Standby mode
- A low-level on the NRST pad
- A timeout from one of the Independent Watchdogs (IWDG[2:1])
- A timeout from one of the window watchdogs if allowed by the reset scope bits (WW[2:1]RSC)
- Software reset request initiated by the Cortex M7 (named CPU1) or Cortex M4 (named CPU2)
- A low-power-mode security reset (which is generated when Stop or Standby mode is entered but is prohibited by the option byte configuration)

Note as well that a System reset asserts the NRST pad, which can be used to reset external components on the board.

The reset source flag can be found in the RCC Control and Status register.

The power-on reset is generated when the V_{DD} or V_{CORE} voltages are lower than a certain voltage threshold.

Reset sources



- Backup domain reset sources
  - BDRST bit in RCC_BDCR register
  - V_{DD} or V_{BAT} power on, if both supplies have previously been powered off
- Domain reset sources
  - Exit from DSTANDBY mode
- Peripherals reset source
  - PERxRST bit in RCC registers
  - System reset
- RCC offers flags in order to identify the reset source.

The Backup domain reset occurs:

- When the BDRST bit in the RCC_BDCR register is set to ‘1’ by the application, as shown in the figure.
- When the V_{DD} and V_{BAT} are powered on if both supplies have previously been powered off.

A D1 or D2 domain reset occurs when one of the two domains has been switched off.

In addition, the peripherals have individual reset control bits.

It is also possible to reset the CPU1 and its associated window watchdog WWDG1 (rst_c1), and to reset the CPU2 and its associated window watchdog WWDG2 (rst_c2).

Reset coverage

6

- The **Power-on reset** resets all the logic located in the V_{DD} and V_{CORE} domains. Back-up domain logic is not affected.
- The **System reset** resets all registers except certain RCC registers, PWR registers, and the Backup domain.
- The **Backup domain reset** resets Backup domain RTC registers, Backup registers, and the RCC_BDCR register.
- The **Domain resets** reset all the logic on the V_{CORE} supply line of the domain (D1 or D2).

life.augmented

The power-on reset is the reset having the largest coverage.
The power-on reset resets all the logic located in the V_{DD} and V_{CORE} domains except those in the Backup domain powered by V_{BAT} which contains the RTC and the external low-speed oscillator.
Note that the power-on reset also triggers the system reset, so the NRST pad is asserted during a power-on reset.
The system reset resets most of the logic located in the V_{DD} and V_{CORE} domains except certain resources located in the RCC and PWR blocks. The backup domain is not affected by this reset.
The backup domain reset resets the Backup domain powered by V_{BAT} which contains the RTC and the external low-speed oscillator.
The Domain reset resets the complete D1 and D2

domains when the corresponding power switch is activated.

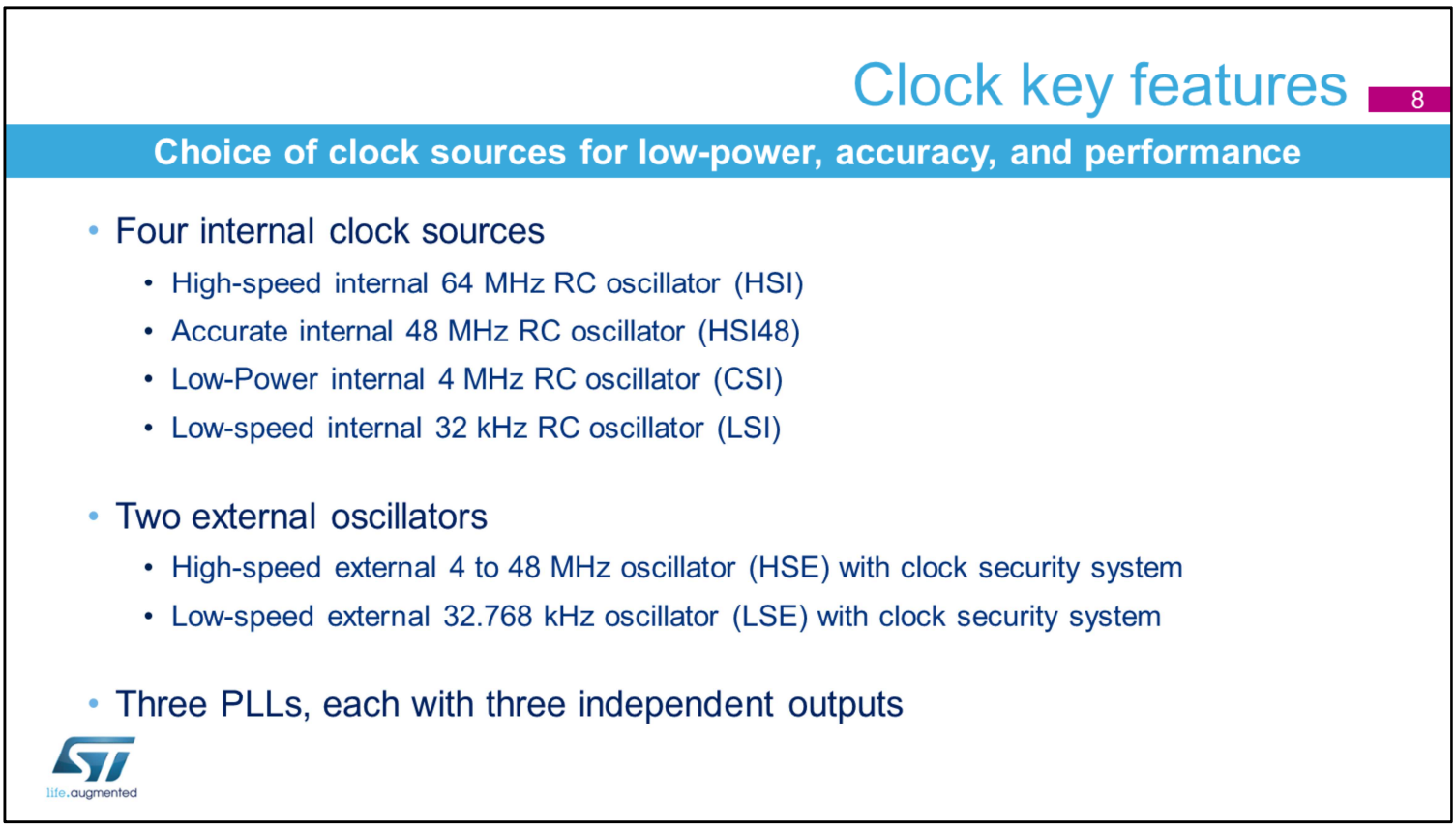
Hold Boot Function

Possibility to choose which processor boots first !

- The HOLD_BOOT function allows to put one of the two CPUs on hold after a system reset or when the system exits from STANDBY mode.
- The Hold Boot function is provided in order to support configurations where one CPU is master of the system and completes the initialization of the complete product (system clock, voltage scaling...) before activating the second CPU.

life.augmented

The Hold Boot function can be used to set one of the two processors to HOLD after a system reset or an exit from Standby mode.
This function can be useful if one processor has to BOOT in advance for example to perform the system initialization.
Note that when one of the processors is in HOLD, it does not prevent the system from entering Standby mode.



The RCC offers a large choice of clock sources, which can be selected depending on low-power, accuracy, and performance requirements.

STM32H7x3/x5/x7 device embeds 4 internal RC oscillators:

- a high-speed internal RC oscillator (HSI) which can work at 64, 32, 16 or 8 MHz,
- a low-power internal RC oscillator (CSI), working at 4 MHz,
- an accurate RC oscillator, working at 48 MHz,
- a low-speed internal 32 kHz RC oscillator (LSI).

STM32H7x3/x5/x7 devices embed 2 oscillators for use with an external crystal or resonator:

- a high-speed external 4 to 48 MHz oscillator (HSE)

with a clock security system and

- a low-speed external 32.768 kHz oscillator (LSE) also with a clock security system.

STM32H7x3/x5/x7 device also embeds 3 phase-locked loops, each with three independent outputs for clocking different peripherals at different frequencies.

High-Speed Internal (HSI) clock

1% accuracy, high-speed, and fast wakeup time

Parameters	Values
Start-up time	2 µs (max.)
Consumption (typ.)	300 µA (typ.)

- HSI 64 MHz, factory- and user-trimmed
- HSI can be selected as
  - Wakeup clock from STOP mode
  - Backup clock for Clock Security System (CSS)
- Can be automatically started when exiting STOP mode.
- Can remain activated during STOP mode, to avoid the start-up penalty.
- Some peripherals can enable the HSI during STOP mode when they need a kernel clock to detect wakeup event conditions.

life.augmented

The high-speed internal oscillator (HIS) is a 64 MHz RC oscillator which provides fast wakeup times. The HSI is trimmed during production testing, and can also be user-trimmed.
A dedicated divider can generate a 32, 16 or 8 MHz clock.
The HSI can be selected as a clock at wakeup from system Stop, and as the backup clock if an HSE failure is detected by the Clock Security System.
The HSI can remain powered when the system goes to STOP mode in order to speed up the wakeup time.
Certain peripherals such as the I2Cs and U(S)ART/LPUART can request the activation of the HSI in system Stop mode in order to generate wakeup events.
The HSI is enabled by the peripheral only for the wakeup sequence detection and remains disabled outside of this

wakeup sequence.

Low-Power Internal (CSI) clock

Low-power, and fast wakeup time

Parameters	Values
Start-up time	1 µs (typ.)
Consumption (typ.)	23 µA (typ.)

- **CSI** 4 MHz, factory- and user-trimmed
- CSI can be selected as wakeup clock from STOP mode
- Can be automatically started when exiting STOP mode
- Can remain activated during STOP mode, to avoid the start-up penalty.
- Some peripherals can enable the CSI during STOP mode when they need a kernel clock to detect wakeup evet conditions.

life.augmented

The low-power internal oscillator (CSI) is a 4 MHz RC oscillator which provides fast wakeup times. The CSI is trimmed during production testing, and can also be user-trimmed.
The CSI can be selected as a clock at wakeup from system Stop.
The CSI can remain powered when the system goes to Stop mode in order the speed-up the wakeup time.
Certain peripherals such as the I2Cs and U(S)ART/LPUART can request the activation of the CSI in system STOP mode in order to generate wakeup events.
The CSI is enabled by the peripheral only for the wakeup sequence detection and remains disabled outside of this wakeup sequence.

High-Speed External (HSE) clock

Safe crystal oscillator system clock

Parameters	Values
Start-up time	2 ms (typ.)
Consumption* (typical)	0.45 mA (typ.)

* VDD=3 V, Rm=30 Ω CL=10 pF at 16 MHz

- **HSE** 4-48MHz
  - External source (Bypass mode) up to 50 MHz
  - External crystal oscillator / ceramic resonator (4 to 48 MHz)
- **Clock Security System (CSS)**
  - Automatic detection of HSE failure with
    - Non-maskable interrupt generation
    - Break input to TIM1/TIM8/TIM15/TIM16/TIM17 => critical applications such as motor control can be put in a safe state.
  - Backup clock is HSI => application software does not stop in case of crystal **oscillator** failure

life.augmented

The high-speed external oscillator (HSE) provides a safe crystal oscillator system clock.
The HSE supports a 4 to 48 MHz external crystal or ceramic resonator, and also an external source in bypass mode.
A clock security system automatically detects an HSE failure. When detected, a Non-Maskable Interrupt is generated, and a break input can be sent to timers in order to put critical applications such as motor control in a safe state. When an HSE failure is detected, the system clock is automatically switched to HSI, so the application software does not stop in case of a crystal oscillator failure.

Low-Speed Internal (LSI) clock

Ultra-low power internal 32 kHz oscillator
Available in all modes except VBAT mode

- The LSI can be used for RTC, LPUARTs, LPTIMs, and IWDGs.

	LSI 32 kHz
Accuracy (typ.)	+/- 1.8 %
Consumption (typ.)	130 nA

life.augmented

STM32H7x3/x5/x7 devices embed an ultra-low-power 32 kHz RC oscillator (LSI), which is available in all modes except VBAT mode.
The LSI can be used to clock the RTC, LPUARTs, the low-power timers, and the independent watchdogs. The accuracy of the LSI is plus or minus 1.8%. The LSI consumption is typically 130 nA.

Low-Speed External (LSE) clock

32.768 kHz configurable for low-power or high-drive

Available in all power modes and in VBAT mode

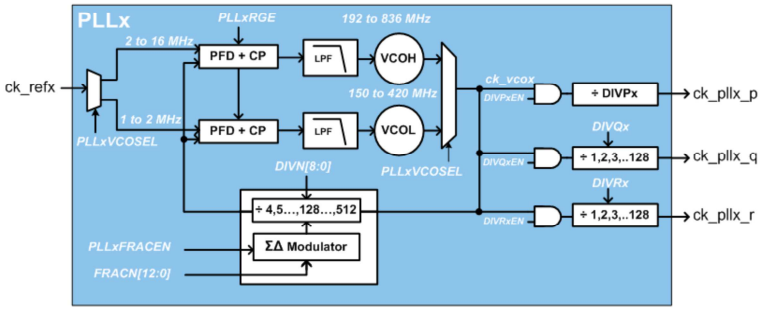
- The LSE can be used with external crystal oscillator or resonator, or with external clock source in bypass mode
- **Clock Security System on LSE**: Available in all modes except VBAT mode.
- The LSE can be used for RTC, U(S)ARTs, LPUART, LPTIMs.

Mode	Maximum critical crystal gm (µA/V)	Consumption (nA)
Ultra-low power	0.5	290
Medium-low driving	0.75	390
Medium-high driving	1.7	550
High driving	2.7	900

life.augmented

The 32.768 kHz low-speed external oscillator can be used with external quartz or resonator, or with an external clock source in bypass mode. The oscillator driving capability is programmable. Four modes are available, from ultra-low power mode with a consumption of only 290 nanoamps, to high-driving mode.
A clock security system monitors the failure of the LSE oscillator. In case of failure, the application can switch from the RTC clock to the LSI clock.
The clock security system is functional in all modes except VBAT mode. It is also functional under reset.
The LSE can be used to clock the RTC, the USARTs or low-power UART peripherals, and the low-power timers.

Wide input range, accurate output frequency



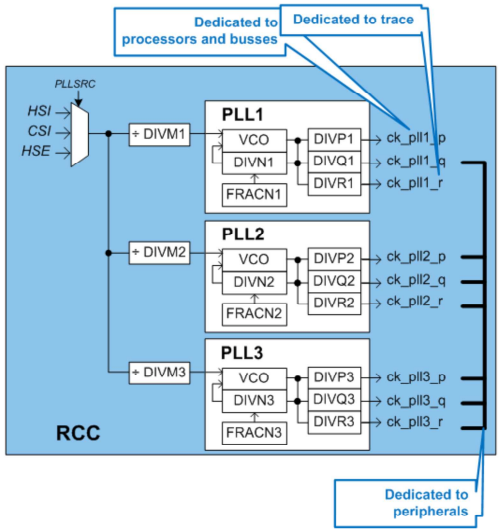
- Three PLLs:
  - Wide input frequency range
    - 1 to 2 MHz with VCOL
    - 2 to 16 MHz with VCLH
  - Wide VCO output frequency range
    - 150 to 420 MHz with VCOL
    - 192 to 836 MHz with VCOH
  - 13-bit fractional multiplication factor
    - Programmable on-the-fly
  - 3 Outputs per PLL
    - With post-divider range from 1 to 128

life.augmented

The PLLs embedded in STM32H7x3/x5/x7 devices provides a flexible way to generate the required frequency for the system or peripheral clocks.
They offer a wide input frequency range from 1 to 16 MHz.
If the input frequency is between 1 and 2 MHz, then the user has to select the VCOL path.
If the input frequency is between 2 and 16 MHz, then the user has to select the VCOH path.
The two VCOs also have a smart frequency range: 150 to 420 MHz for VCOL and 192 to 836 MHz for VCOH.
The PLLs also provides 3 different outputs which are all derived from the VCO output via post-dividers (DIVP, DIVQ and DIVR).
In addition, it is possible to change the values of the post-divider without disabling the PLLs. The application just needs to disable the corresponding post-divider,

change the division ratio, and re-enable the post-divider. In order to get a duty cycle close to 50%, the application has to program the post-dividers to even values.
In addition, the PLLs can be switched to fractional mode, allowing a high precision in the VCO frequency.
The 13-bit fractional divider can be changed without disabling the PLLs. This feature can be used to perform accurate clock drift compensation.

Simplified clock distribution

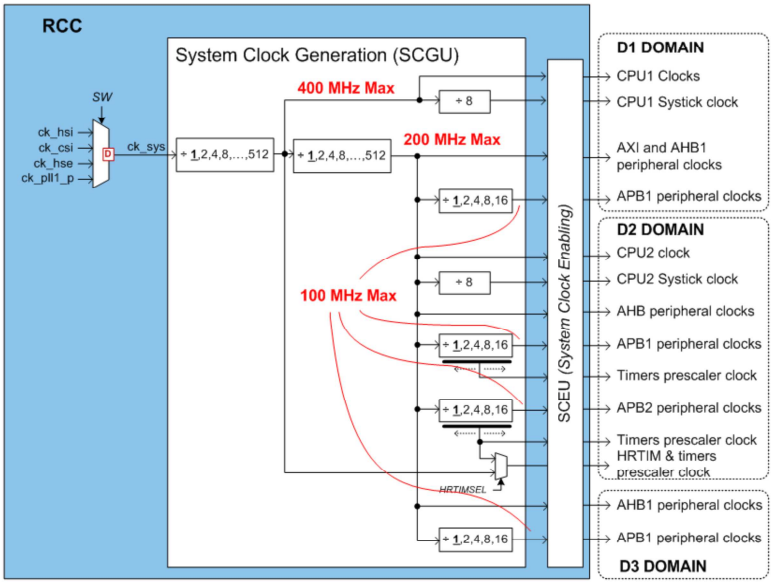


- Clocks generated by the PLLs:
  - Possible clock sources: HSE, HSI and CSI
  - A dedicated pre-divider in from off each PLL
  - 7 outputs dedicated to the generation of peripheral kernel clock
  - 1 output dedicated to processors and busses
  - 1 output dedicated to debug trace

life.augmented

The PLLs share the same clock source: HSI, CSI or HSE.
Each PLL has a dedicated pre-divider in order to adjust the reference clock for each PLL.
PLL2 and PLL3 are dedicated to the generation of the kernel clocks for the peripherals. Certain peripherals such as the SAI, USB, etc. may require a specific frequency.
The PLL1 P output is used as system clock for processors and busses, the PLL1 R is dedicated for the debug trace, and the PLL1 Q is also used as kernel clock for peripherals.
Note that the PLLs clock source cannot be changed if one of the PLL is enabled.

Simplified clock tree

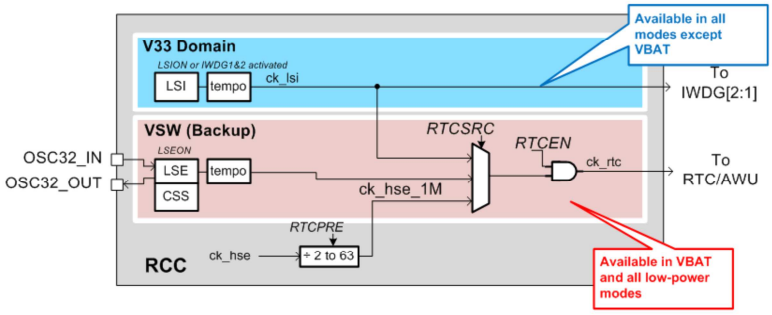


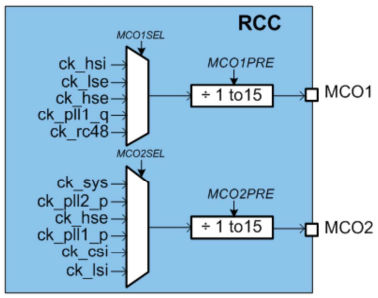
- Dynamic switch for system clock selection
- System clock source can be:
  - HSI (default after reset)
  - CSI
  - HSE
  - PLL1_P output
- Dynamic frequency dividers allow easy frequency adjust

life.augmented

The system clock can be derived from the HSI, CSI, HSE or the output of DIVP of PLL1.
The switch used to select the system clock is dynamic, meaning that it is possible to change the frequency on-the-fly, without generating timing violations.
In addition, all the pre-scalers presented in the figure are dynamic, so they can be changed on-the-fly as well, making the frequency scaling operation very simple.
The clock of the CPU1 does not exceed 480 MHz.
The AHB clocks do not exceed 240 MHz, as well as the CPU2 speed.
The APB clocks do not exceed 120 MHz.
Note that HRTIM and certain timers can use clock frequencies up to 480 MHz.

Simplified clock tree



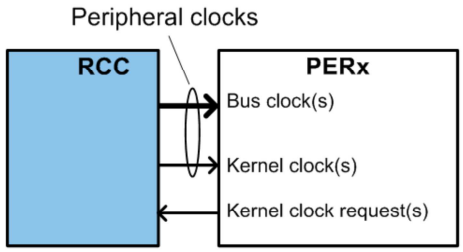


- Clock source for RTC/AWU can be:
  - LSE
  - LSI
  - HSE divided
- Two clock outputs: MCO1 and MCO2 offering a big choice of clock source.

The clock for the RTC/AWU can be selected from among: LSE, LSI or HSE divided.
Note that when HSE divided is used as a kernel clock for the RTC/AWU, the frequency cannot exceed 1 MHz.
Note as well that if LSI is used as a kernel clock for the RTC/AWU, the RTC/AWU will no longer be clocked if the V_{DD} and V_{CORE} supply are removed.
If HSE divided is used as a kernel clock for the RTC/AWU, the RTC/AWU will no longer be clocked if the system goes to Stop mode or if the V_{CORE} supply is removed.
The LSE can remain enabled in all low-power modes and in V_{BAT} mode.
The RCC also provides two clock output signals: MCO1 and MCO2.
A prescaler allows the application to adapt the frequency to the pads' capability.

Peripheral Clock Distribution

Bus speed can be changed without affecting peripherals



- Peripherals generally receive:
  - One or several bus clocks
  - One or several kernel clocks
- The bus clock allows the access to peripheral control and data stream.
- The kernel clock is generally used by the peripheral to handle its external interface: I2S, SPI, USB, Ethernet…

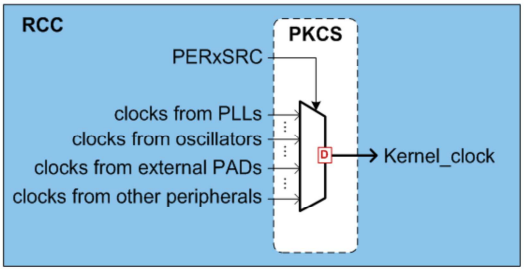
life.augmented

Many STM32H7x3/x5/x7 peripherals have different clocks for the data and control streams via the processor bus interface, as well as the clock for the specific peripheral interface.
Generally the clocks for the data and control streams via the processor bus interface are named ‘Bus clock(s)’, and the clock(s) for the peripheral specific interface are named ‘kernel clocks’.
The peripheral clocks represent the clocks received by the peripheral: ‘bus clock(s)’ and ‘kernel clock(s)’.
Having a separate bus clock and kernel clock allows the application to change the interconnect and processor working frequency without affecting the peripheral.
For certain peripherals it is also possible to disable the bus clock as long as the peripheral does not need to transfer data to the system.
So it gives a good flexibility on the frequency selection

for the bus processor and memories, and the real need of the peripheral interface.
For example: The UARTs have a kernel clock which is used, among other things, by the baud rate generator for the serial interface communication, and an APB clock for the register interface.
In addition, certain peripherals are able to request the kernel clock when they detect specific events.

Peripheral Clock Distribution

Kernel clock switches are dynamic and glitch-free



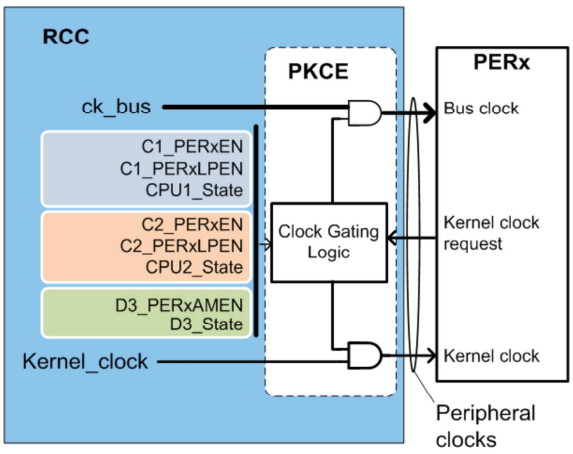
- Most of the peripherals receiving a kernel clock have a clock switch to select the optimal kernel clock.
  - The clock switch is dynamic and glitch-free.
  - Several clock sources are proposed depending on the peripheral's requirements
  - The kernel clock switches can be configured via RCC registers

life.augmented

The distribution of the kernel clocks is not detailed in this presentation. Please refer to the reference manual for specific details.
Most of the peripherals receiving a kernel clock have a dynamic clock switch to select the optimal clock source.
The proposed clock sources generally come from:
one of the 7 PLL outputs dedicated to the kernel clock generation
internal or external oscillators: this is mandatory for peripherals needing a kernel clock when the system is in Stop mode
Pads: for example on peripherals using an external PHY, but also for audio as well, in order to use a clock reference from an external device
other internal peripherals: for synchronization between blocks.
The dynamic switching facilitates the transition from one source to another.

RCC registers are used to configure the kernel clocks for all the peripherals.

Peripheral Clock Gating



- The peripheral clocks gating is controlled by several parameters:
  - Enables bits (C1_PERxEN, C2_PERxEN)
  - Low-power enables bits (C1_PERxLPEN, C2_PERxLPEN)
  - Processor and domain states
  - Autonomous bits (D3_PERxAMEN)
- Setting the bit C1_PERxEN to '1' indicates that the peripheral PERx is enabled for the CPU1.
- Setting the bit C2_PERxEN to '1' indicates that the peripheral PERx is enabled for the CPU2.

Peripherals generally receive:

- One or several bus clocks
- One or several kernel clocks

Each processor can control the clock gating of the peripheral clocks via dedicated registers located in the RCC.

The gating of the peripheral clocks depends on several parameters:

- The clock enable bits, each processor have a dedicated control bit for that, named C1_PERxEN and C2_PERxEN, for simplification
- The low-power clock enable bits
- The processors states (CRUN, CSLEEP or CSTOP)
- The autonomous bits for peripherals located in D3

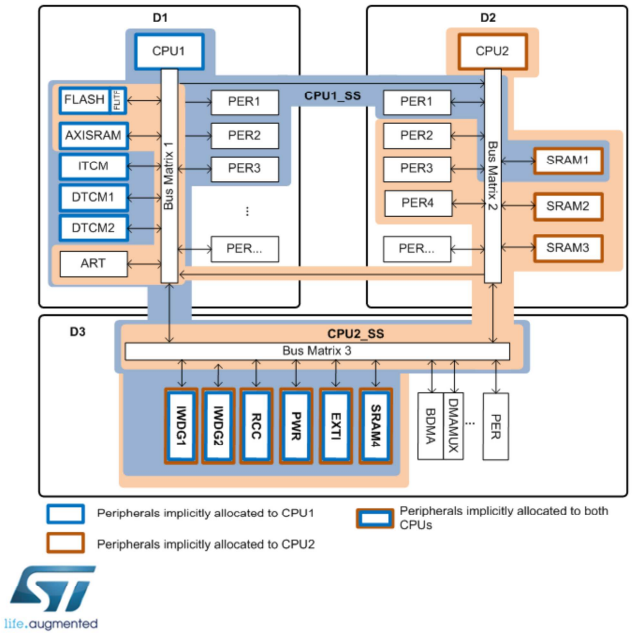
domain

Setting the bit C1_PERxEN to ‘1’ indicates that the peripheral PERx is enabled for the CPU1.

Setting the bit C2_PERxEN to ‘1’ indicates that the peripheral PERx is enabled for the CPU2.

This operation is named “allocating” a peripheral.

Peripheral Allocation



- The allocation informs the RCC that the CPU1 or CPU2 enabled a peripheral. This information is used for low-power modes.
- Prior to using a peripheral, the CPUs have to allocate (i.e. enable) it.
- The FLASH, AXISRAM, ITCM, DTCM1 and DTCM2 are implicitly allocated (i.e. enabled for) to CPU1.
- The D2SRAM1, D2SRAM2 and D2SRAM3 are implicitly allocated (i.e. enabled for) to CPU2.
- The IWDG1, IWDG2, RCC, PWR, EXTI and D3SRAM4 are implicitly allocated (i.e. enabled for) to both CPUs.

It is important to note that the RCC offers two register sets, allowing each processor to enable (or allocate) peripherals.

The peripheral allocation informs the RCC that the CPU1 or CPU2 enabled a peripheral. This information is used by the RCC for the clock control in low-power modes.

So before using a peripheral, the CPUs have to allocate it.

The same peripheral can be allocated by both processors, it is up to the application to avoid resource conflicts.

Certain peripherals are implicitly allocated to a specific processor:

- **The FLASH, AXISRAM, ITCM, DTCM1 and DTCM2**

are implicitly allocated (i.e. enabled for) to CPU1, the CPU2 can allocate any of them, but by default they are not allocated to CPU2.

- **The D2SRAM1, D2SRAM2 and D2SRAM3 are implicitly allocated (i.e. enabled for) to CPU2, the CPU1 can allocate any of them, but by default they are not allocated to CPU1.**

Certain other peripherals are allocated to both processors. This is the case for the IWDG1, IWDG2, RCC, PWR, EXTI and D3SRAM4.

When a CPU allocates a peripheral, this peripheral is ‘linked’ to the processor state for the low-power modes.

The CPU, plus the peripherals allocated by this CPU, and the associated interconnect is considered by the RCC as a CPU sub-system.

The D1 and D2 domain core voltages can be switched-off.

To give a simple example of how the RCC uses the peripheral allocation, we can state that the RCC will not allow a domain to be switched off if one of the peripherals of this domain is used by the processor of the other domain which is not switched off.

System States

22

System States	D1 States	D2 States	D3 States
RUN	DRUN/DSTOP/DSTANDBY	DRUN/DSTOP/DSTANDBY	DRUN
STOP	DSTOP/DSTANDBY	DSTOP/DSTANDBY	DSTOP
STANDBY	DSTANDBY	DSTANDBY	DSTANDBY

- A domain is in DRUN when its bus matrix is clocked. Its bus matrix will be clocked if a CPU which is in CRUN/CSLEEP allocated a peripheral on this domain.
- A domain is in DSTOP when its bus matrix is no longer clocked. The CPU of this domain is in CSTOP, the other CPU is not allocating peripheral on this domain, or the CPU is also in CSTOP.
- A domain is in DSTANDBY when its V_{CORE} supply is switched off.

The following table gives a simplified view of the system states versus domain states.

- A domain is in DRUN mode when its bus matrix is clocked. Its bus matrix will be clocked if a CPU which is in CRUN/CSLEEP mode allocated a peripheral on this domain.
- A domain is in DSTOP mode when its bus matrix is no longer clocked. The CPU of this domain is in CSTOP mode, the other CPU is not allocating peripheral on this domain, or the CPU is also in CSTOP mode.
- A domain is in DSTANDBY mode when its V_{CORE} supply is switched off.

Note that it is possible to keep only the D3 domain in DRUN mode while other domains are in DSTANDBY

mode.

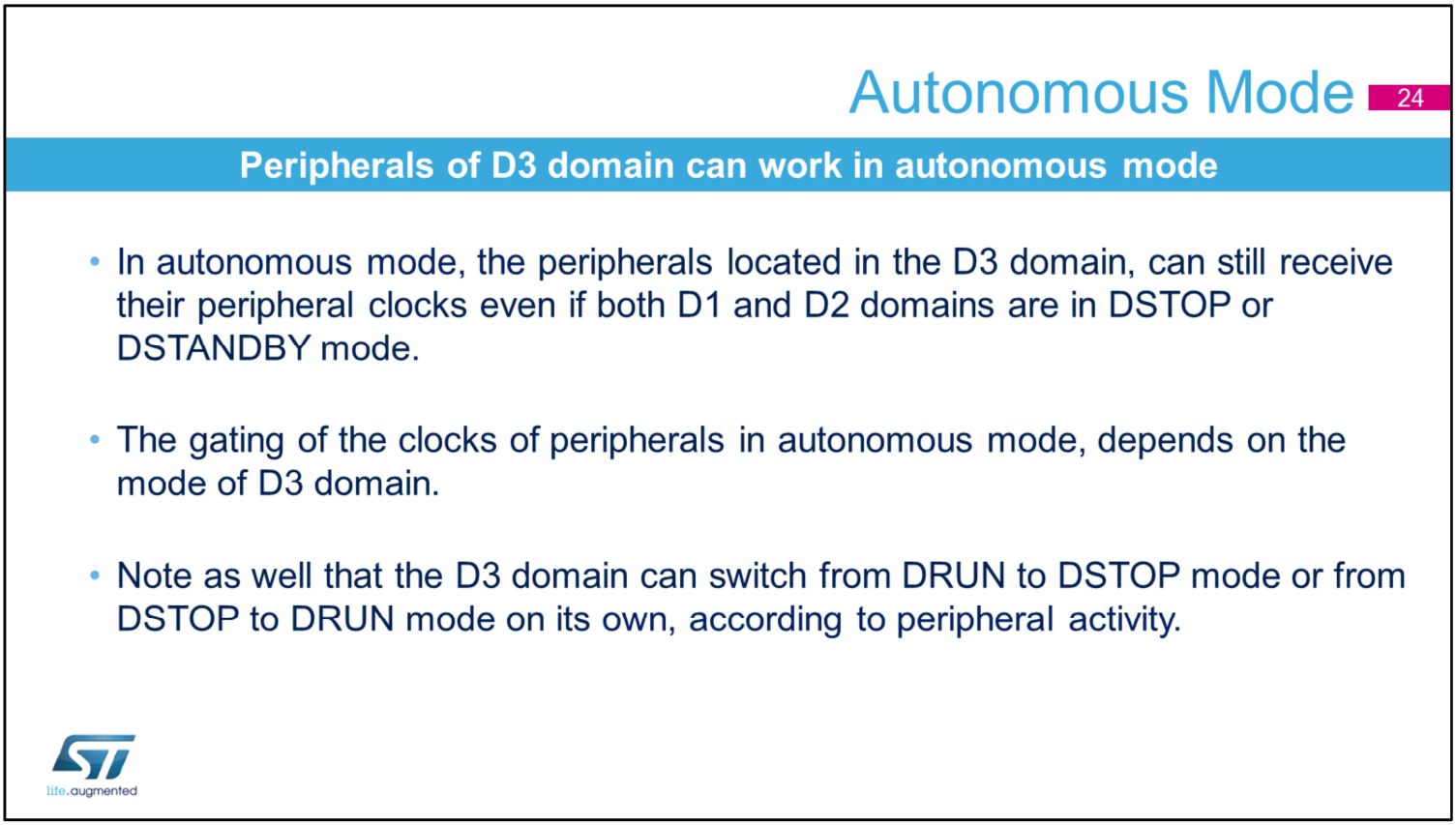
For more details on system states, please refer to the PWR training slides.

System States

- System restart:
  - The HSI is selected as system and peripheral clock
  - All PLLs are switched OFF
- System STOP restart:
  - The application can select HSI or CSI as system clock ,
  - The application can select HSI or CSI as peripheral clock ,
- DSTOP or CSTOP restart:
  - The clock settings used before going to DSTOP/STOP are kept

life.augmented

Note that when the system restarts for example after a system reset, the HSI is selected as the system and peripheral clock.
The CSI, HSI48 and HSE are off, as well as the PLLs.
The LSI is still working (if it was previously enabled).
After a system stop, the PLLs are switched off. The application can choose either the HSI or CSI as the system clock and peripheral clock.
The HSI48 and HSE are off.
After a D1 or D2 domain exits DSTOP or CSTOP mode, the RCC maintains the same clock setting as the one used before going to low-power mode.
CSLEEP mode does not affect the clock settings, but only acts on the clock gating.



The autonomous mode offers the possibility to provide the peripheral clocks to peripherals located in D3, even:

- if the CPU to which they are allocated is in CSTOP mode.
- if both D1 and D2 domains are in DSTOP or DSTANDBY mode.

The gating of the clocks of peripherals in autonomous mode, depends on the mode of D3 domain:

- if the D3 Domain is in DRUN mode, the peripheral will receive its peripheral clocks
- if the D3 Domain is in DSTOP mode, the peripheral clocks are gated, but the peripherals can still request the kernel clock if they have this capability.

The autonomous mode does not prevent the D3 domain

from entering DSTOP or DSTANDBY mode.

The D3 domain is able to switch alternatively from DRUN to DSTOP mode, and then from DSTOP to DRUN mode according to peripheral activity.

For example, if the system is expecting messages via I2C4; the complete system can be put in Stop mode.

When the I2C4 detects a START bit, then it will generate a “kernel clock request”.

This request enables the HSI or CSI, and a kernel clock is provided only to the requester. Then the I2C4 can decode the incoming message.

Several cases may occur:

- if the device address of the incoming message does not match, then the I2C4 releases its “kernel clock request” until a new START condition is detected.
- if the device address of the incoming message matches, the message has to be stored in the D3 local memory. The I2C4 is able to generate a wake-up event on address match in order to switch the D3 domain to DRUN mode. When the D3 domain is in DRUN mode, the bus interface clock is activated, allowing the I2C4 to transfer the message into memory via DMA1.
    - If the D3 local memory is not full, then the D3 domain will go back to DSTOP mode, without any CPU activation.
    - If the D3 local memory is full, the DMA1 can generate a wake-up event in order to activate CPU1 or CPU2.

Please refer to D3 Domain low-power slide set for details.

Kernel Clock Requests

- Some peripherals such as the UARTs and I2Cs are able to asynchronously detect events requesting a kernel clock for the processing.
- The RCC can provide on demand a kernel clock to those peripherals, when the CPU allocating the peripheral is in CSTOP or if the system is in STOP mode.
- The kernel clock request will work if the selected kernel clock is: HSI or CSI.
- In order to speed-up the activation time, the HSI or CSI can be maintained activated during system STOP mode.
- In system STOP mode, the LSI and LSE clocks are still available for peripherals.

life.augmented

Certain peripherals such as the UARTs and I2Cs are able to asynchronously detect events requesting a kernel clock for the processing.
The RCC can provide on demand a kernel clock to those peripherals, when the CPU allocating the peripheral is in CSTOP mode or if the system is in STOP mode.
The kernel clock request will work if the selected kernel clock is: HSI or CSI.
In order to speed-up the activation time, the HSI or CSI can be maintained activated during system STOP mode.
In system STOP mode, the LSI and LSE clocks are still available for peripherals.

Peripheral Clock gating

- A **kernel clock** is provided to the peripherals PERx of **D1** or **D2** domains if one of the following conditions is met:
  1. When the CPU to which the peripheral is allocated is in CRUN mode **and** PERxEN = '1'.
  2. When the CPU to which the peripheral is allocated is in CSLEEP mode **and** PERxLPEN = '1'.
  3. When the CPU to which the peripheral is allocated is in CSTOP mode, **and** PERxAMEN = '1', **and** the peripheral is generating a kernel clock request **and** the kernel clock source is HSI or CSI.
  4. When the CPU to which the peripheral is allocated is in CSTOP mode, **and** PERxAMEN = '1', **and** the kernel clock source of the peripheral is LSE or LSI.
- The **bus interface** clock will be provided to the peripherals only when Conditions 1 or 2 are met.

life.augmented

A kernel clock is provided to the peripherals located in D1 or D2 domains if one of the following conditions is met:

1. When the CPU to which the peripheral is allocated is in CRUN mode and if its Enable bit is set to '1'.
2. When the CPU to which the peripheral is allocated is in CSLEEP mode and if its LP-Enable bit is set to '1'.
3. When the CPU to which the peripheral is allocated is in CSTOP mode, and if its LP-Enable bit is set to '1' and the peripheral is generating a kernel clock request and the kernel clock source is HSI or CSI.
4. When the CPU to which the peripheral is allocated is in CSTOP mode, and if its LP-Enable bit is set to '1', and the kernel clock source of the peripheral is LSE or LSI.

The bus interface clock will be provided to the peripherals only when Conditions 1 or 2 are met.

Peripheral Clock gating

27

- A **kernel clock** is provided to the peripherals PERx of **D3** domain if one of the following conditions is met:
  1. When the CPU to which the peripheral is allocated is in CRUN mode **and** PERxEN = '1'.
  2. When the CPU to which the peripheral is allocated is in CSLEEP mode **and** PERxLPEN = '1'.
  3. When the CPU to which the peripheral is allocated is in CSTOP mode, **and** the D3 domain is in DRUN mode, **and** PERxAMEN = '1'.
  4. When the CPU to which the peripheral is allocated is in CSTOP mode, **and** the D3 domain is in DSTOP mode, **and** PERxAMEN = '1', **and** the peripheral is generating a kernel clock request **and** the kernel clock source is HSI or CSI.
  5. When the CPU to which the peripheral is allocated is in CSTOP mode, **and** the D3 domain is in DSTOP mode, **and** PERxAMEN = '1', **and** the kernel clock source of the peripheral is LSE or LSI.
- The **bus interface** clock will be provided to the peripherals only when Conditions 1, 2 or 3 are met.

life.augmented

A kernel clock is provided to the peripherals PERx of the D3 domain if one of the following conditions is met:

1. When the CPU to which the peripheral is allocated is in CRUN mode and if its Enable bit is set to '1'.
2. When the CPU to which the peripheral is allocated is in CSLEEP mode and PERxLPEN = '1'.
3. When the CPU to which the peripheral is allocated is in CSTOP mode, and the D3 domain is in DRUN, and if its LP-Enable bit is set to '1'.
4. When the CPU to which the peripheral is allocated is in CSTOP mode, and the D3 domain is in DSTOP, and if its LP-Enable bit is set to '1', and the peripheral is generating a kernel clock request and the kernel clock source is HSI or CSI.
5. When the CPU to which the peripheral is allocated is

in CSTOP mode, and the D3 domain is in DSTOP mode, and if its LP-Enable bit is set to ‘1’, and the kernel clock source of the peripheral is LSE or LSI.

The bus interface clock will be provided to the peripherals only when Conditions 1, 2 or 3 are met.

Interrupts

Interrupt event	Description
LSE clock security system	Set when a failure is detected in the LSE oscillator
HSE clock security system	Set when a failure is detected in the HSE oscillator
PLL1 ready interrupt flag	Clock ready caused by PLL1 lock
PLL2 ready interrupt flag	Clock ready caused by PLL2 lock
PLL3 ready interrupt flag	Clock ready caused by PLL3 lock
HSE ready	Clock ready caused by the HSE oscillator
HSI ready	Clock ready caused by the HSI oscillator
CSI ready	Clock ready caused by the CSI oscillator
HSI48 ready	Clock ready caused by the HSI48 oscillator
LSE ready	Clock ready caused by the LSE oscillator
LSI ready	Clock ready caused by the LSI oscillator

This slide lists the RCC interrupts. The LSE and HSE clock security systems, the PLL ready, and all 6 oscillator ready signals can generate an interrupt.

Related peripherals

- Refer to these trainings linked to this peripheral, if needed:
  - STM32H7 Power control (PWR)
  - STM32H7 Asynchronous Interrupts and Event Controller (EXTI)

life.augmented

In addition to this training, you may find the Power Control and Interrupt Controller training useful.