Hello, and welcome to this presentation of the STM32WB’s reset and clock control.
The reset and clock controller manages the various reset mechanisms and the generation of the system and peripheral clocks. STM32WB microcontrollers embed 5 internal oscillators, 2 oscillators for an external crystal or resonator, and 2 phase-locked loops (PLL).

Many peripherals have their own kernel clock, independent of the system clock. The 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.
Safe and flexible reset management without external components reduces application costs.

The RCC manages several types of resets: the power reset, the system reset, the local resets, and the backup domain reset.
Thanks to the voltages monitoring feature included into the power block (PWR), the filters embedded in the NRST pad, and the RCC reset controller, the amount of external components is reduced to a single external capacitor connected to 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 VDD or VFBSMPS supply (see PWR block for details),
  - An invalid voltage on VDD due to brown-out function. The brown-out function allows the user to choose its own threshold levels for the VDD supply (see PWR block for details),
  - An exit from Standby or Shutdown mode
• A low-level on the NRST pad
• A timeout from the independent watchdog
• A timeout from the window watchdog
• Software reset request initiated by the Cortex M4 or Cortex M0+ core
• A low-power-mode security reset (which is generated when Stop, Standby or Shutdown mode is entered but is prohibited by the option byte configuration).

Note as well that the system reset (except when generated by a standby reset) asserts the NRST pad, allowing the reset of external components when a system reset occurs. The reset source flag can be found in the Reset and Clock Controller status register.
The power-on reset is the reset having the largest coverage. The power-on reset, resets all the logic located in the VDD and VFBSMPS domains except those in the Backup domain powered by VBAT 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 power-on reset.

The system reset resets most of the logic located in VDD domain except some resources located into the RCC and the PWR blocks. The backup domain is not affected by this reset. The backup domain reset, resets the backup domain powered by VBAT which contains the RTC and the external low-speed oscillator. In addition, most peripherals have individual local reset control bits.
The RCC offers a large choice of clock sources, which can be selected depending on low-power, accuracy, and performance requirements. STM32WB microcontrollers embed 5 internal RC oscillators: a high-speed internal RC oscillator (HSI) which can work at 16 MHz, a low-power internal RC oscillator (MSI), working at 100 kHz to 48 MHz, an accurate RC oscillator (HSI48), working at 48 MHz, a low-speed low power internal 32 kHz RC oscillator (LSI1), a low-speed low-drift internal ~32 kHz RC oscillator (LSI2). STM32WB microcontrollers also embed 2 oscillators for use with an external crystal or resonator: a high-speed external 32 MHz oscillator (HSE) with a clock security system and a low-speed external 32.768 kHz oscillator (LSE) also with a clock security system. STM32WB microcontrollers also embed 2 phase-locked
loops, each with three independent outputs for clocking the CPUs and different peripherals at different frequencies.
The high-speed internal oscillator (HSI) is a 16 MHz RC oscillator which provides 1% accuracy and fast wakeup times. The HSI is trimmed during production testing, and can also be user-trimmed. The HSI can be selected as clock at wakeup from system stop, and as the backup clock if an HSE failure is detected by the Clock Security System. The HSI is selected as system clock at wakeup from Standby mode.

The HSI can remain powered when the system goes to Stop mode in order to speed up the wakeup time, and use as kernel clock by peripherals in Stop mode. Some peripherals such as the I2Cs, USART/LPUART and LPTIMs can select the HSI as kernel clock. The HSI frequency can be trimmed versus HSE by using the MCO and TIM17 bits in Capture mode.
The low-power internal oscillator (MSI) is a multi-frequency RC oscillator in the 100 kHz to 48 MHz range which provides 3% accuracy and fast wakeup times. The MSI is trimmed during production testing, and can also be user-trimmed. The MSI can be selected as clock at wakeup from system stop. The MSI is selected as system clock after reset. Some peripherals such as the USB can use the MSI as kernel clock. The MSI frequency can be trimmed versus HSE by using the TIM17 bits in Capture mode.
The 48 MHz internal oscillator (HSI48) is a 48 MHz RC oscillator which provides 3.5% accuracy and fast wakeup times. The HSI48 is trimmed during production testing, and can also be user-trimmed by using the Clock Recovery System.

The HSI48 can be selected as clock for the USB and True Random Number Generator kernel clock.

The HSI 48 frequency can be trimmed versus HSE by using the MCO and TIM17 bits in Capture mode.
The high-speed external oscillator (HSE) provides a safe crystal system clock. The HSE supports a 32 MHz external crystal resonator. The frequency can be tuned to the required few 1/10 of ppm using on-chip capacitor trimming. Bias and current control tuning is also possible.

A clock security system allows an automatic detection of HSE failure. In this case 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 or MSI, so the application software does not stop in case of crystal failure.

The use of the high-speed external oscillator clock is mandatory when the radio is active. The high-speed external oscillator will automatically be managed in line with the radio activity.
STM32WB microcontrollers embed two LSI oscillators. An ultra-low-power 32 kHz RC oscillator named LSI1, and a low-drift 33 kHz RC oscillator named LSI2. Both are available in all modes except Shutdown and VBAT. One of the two low frequency RC oscillators can be selected as internal LSI clock.

The LSI can be used to clock the RTC, LCD, low-power timers, and the independent watchdog. Only the LSI2 can be selected for use by the radio system.

The accuracy of the LSI1 is plus or minus 1.6%, plus 1.5% over temperature and plus 0.2% over voltage. The LSI1 consumption is typically 110 nA.

The initial frequency of the LSI2 is between 22 and 44 kHz, with a stability of 125 parts per million per Celsius degree over temperature and voltage. The LSI2 typically consumes 500 nA.

The LSI frequency can be trimmed versus HSE by using the TIM16 bits in Capture mode.
The 32.768 kHz low-speed external oscillator (LSE) can be used with an 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 250 nA, to High-driving mode.

A clock security system monitors failure of the LSE oscillator. In case of failure, the application can switch from the RTC clock to the selected LSI clock.

The clock security system is functional in all modes except VBAT. It is also functional under reset.

The LSE can be used to clock the radio system, the RTC, the LCD, the low-power timers, the USART, and low-power UART peripherals.
The system clock can be derived from the HSI, MSI, HSE or the PLLRCLK output of the PLL system. The switch used to select the system clock is dynamic, meaning that it is possible to change the frequency on-the-fly according to application performance needs. The Cortex-M4 core, Cortex-M0+ radio system and the Flash memory have their independent clock dividers allowing to run each of them on different frequencies. It is recommended to run the Flash memory on the HCLK shared at least at the same speed as the highest frequency selected for the Cortex-M4 and Cortex-M0+ cores. 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.
To optimize power consumption at lower frequencies the operating range can be changed or Low-power Run mode can be selected.

In Range 1, the clocks of the Cortex-M4 (HCLK) and Shared bus (HCLKS) must not exceed 64 MHz, and the Cortex-M0+ clock must not exceed 32 MHz.

In Range 2, the clocks of the Cortex-M4 (HCLK), Shared bus (HCLKS), and the Cortex-M0+ must not exceed 16 MHz.

In Low-power Run mode, the clocks of the Cortex-M4 (HCLK), Shared bus (HCLKS), and the Cortex-M0+ must not exceed 2 MHz.

BLE operation requires a HCLK2 clock frequency of at least 16 MHz and is not allowed in Low-power Run mode.

---

**Clock frequency and voltage scaling**

**Power optimization for lower frequencies**

- When running at lower frequencies, additional power can be saved by changing the operating range.
  - **Range 1**
    - Max. Cortex-M4 frequency = 64 MHz
    - Max. Cortex-M0+ frequency = 32 MHz
    - Max. PLL VCO frequency = 344 MHz
    - Max. MSI frequency = 48 MHz
  - **LPRun**
    - Max. Cortex-M4 frequency = 2 MHz
    - Max. Cortex-M0+ frequency = 2 MHz
    - PLL disabled
    - Max. MSI frequency = 2 MHz
  - **Range 2**
    - Max. Cortex-M4 frequency = 16 MHz
    - Max. Cortex-M0+ frequency = 16 MHz
    - Max. PLL VCO frequency = 128 MHz
    - Max. MSI frequency = 24 MHz

Not compatible with RF operation

**BLE operation requires at least 16 MHz on HCLK2.**
The PLLs embedded into the STM32WB microcontroller provides a flexible way to generate the required frequency for the system or peripheral clocks. They offer a wide input frequency range from 4 to 16 MHz. The PLLs share the same clock source: HSE, HSI or MSI which can be pre-divided. The PLL VCO has a wide frequency range from 64 to a maximum of 344 MHz in Range 1 and a maximum of 128 MHz in Range 2. Both PLLs provide 3 different output clocks which are all derived from the PLL VCO frequency via post-dividers (/P, /Q and /R). The PLLSYS is used to generate the system clock and SAI and USB kernel clocks. The PLLSAI is dedicated to generating kernel clocks for the SAI, ADC and USB peripherals.
The Multi-Clock Output is available on GPIO pin PA8 in Run and Stop modes and can select various high- and low-speed clocks.

A Low-Speed Clock Output is available on GPIO pin PA2 in Run, Stop and Standby modes and can select various low-speed clocks.
Some peripherals have a separate clock for the processor bus interface and the specific peripheral interface function. The bus clock is used to access the peripheral registers, whereas the kernel clock is used for the specific peripheral interface function.

Having a separate bus clock and kernel clock allows the application to change the interconnect and processor working frequency without affecting the peripheral operation. For example, the USART kernel clock is used to generate the baud rate for the serial interface communication, and the bus clock for the register interface.

The enabling of both the peripheral bus clock and kernel clock is controlled by the Reset and Clock Controller’s peripheral enable and sleep mode enable bits. When both bits are set to one, the peripheral is able operate and transfer data in Sleep mode. When HSI, LSI, or LSE is selected as the kernel clock, the peripheral is able to operate and wake up the system from Stop mode. In Stop mode, the

<table>
<thead>
<tr>
<th>xxxEN</th>
<th>xxxSMEN</th>
<th>bus clock</th>
<th>kernel clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Stopped</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Active in Run mode, stopped in Sleep and Stop modes</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Active in Run and Sleep modes, stopped in Stop mode</td>
<td>Active in Run, Sleep and Stop modes</td>
</tr>
</tbody>
</table>

Some peripherals are able to operate on its kernel clock in Stop mode

- I2C, LCD, USART, LPUSART and LPTIM

When the kernel clock selects HSI LSI, or LSE.
peripheral is not able to transfer data on the bus matrix, for example to memory. Refer to the specific peripheral training slides for more information. The USART and I2C1 are only available in Stop 0 and Stop 1 modes.
CPU, bus matrix and peripheral clocks are gated according to the CPU operating mode, the peripheral allocation and the peripheral sleep mode enable bit. A peripheral is allocated to a CPU when the Reset and Clock Controller peripheral enable bit belonging to the CPU is set. The peripheral and the associated bus matrix is clocked whenever the CPU is in CRUn mode. Before accessing a peripheral, it must be enabled by the CPU. When both CPUs need to access a peripheral, they must both enable the peripheral by the Reset and Clock Controller peripheral enable bits belonging to the CPUs.
A CPU is only clocked when in CRun mode. Only allocated peripherals are clocked when the CPU is in CRun, or when the CPU is in CSleep when the peripheral sleep mode operation is enabled. A bus matrix will be clocked when a CPU or peripheral on the bus matrix is clocked.
It is important to notice that the Reset and Clock Controller offers two register sets, allowing each processor to allocate (enable) peripherals. A peripheral and the associated bus matrix will only be clocked when allocated by a CPU and the CPU is in CRun mode, or in CSleep mode when the CPU peripheral sleep mode enable bit for this peripheral is also set.

Depending on the peripheral function, peripherals have a different behavior.

Peripherals needed for the system to operate don't have enable bits, and are all time-allocated to both CPUs. The SRAM1 and Quad SPI are dedicated to the CPU1 and can't be allocated by the CPU2. The radio peripherals can only be allocated by CPU2. All other peripherals can be allocated by both CPUs. Before accessing a peripheral the CPU must allocate it. If a peripheral is shared by both CPUs, it must be allocated by both processors; it is up to the application to avoid peripheral
access conflicts. A Hardware Semaphore IP exists to help manage accessing shared peripherals.
The peripheral allocation allows to dynamically configure a CPU sub-system, and have only the peripherals used by the CPU being clocked. The CPU, plus the peripherals allocated by this CPU, and the associated bus matrixes are considered by the Reset and Clock Controller as a CPU sub-system. To give a simple example, when the CPU1 is active in Run mode, the system peripherals, the CPU1 peripherals and any allocated peripheral will run as well, including the shared bus matrix and the CPU1 bus matrix.
The Radio system has its own sub-system which consists of the BLE and IEEE 802.15.4 RF modules, the system peripherals and both the shared bus matrix and radio bus matrix. This allows the radio system to transfer data to SRAM2.
The following table gives a simplified view of the system states versus sub-system states.

- Sub-system states
  - In CRun or CSleep mode, its bus matrix and peripherals are clocked.
  - In CStop mode, its bus matrix and peripherals bus clock are stopped.

- System states
  - Is in Run mode when at least one sub-system is in CRun or CSleep mode.
  - Is in Stop, Standby or Shutdown mode when all sub-systems are in CStop mode.

<table>
<thead>
<tr>
<th>System States</th>
<th>Cortex-M4 sub-system (CPU1)</th>
<th>Cortex-M0+ sub-system (CPU2)</th>
<th>Radio system sub-system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run*</td>
<td>CRun/CSleep/CStop</td>
<td>CRun/CSleep/CStop</td>
<td>CRun/CStop</td>
</tr>
<tr>
<td>Stop</td>
<td></td>
<td>CStop</td>
<td>CStop</td>
</tr>
<tr>
<td>Standby</td>
<td>CStop</td>
<td>CStop</td>
<td>CStop</td>
</tr>
<tr>
<td>Shutdown</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* At least one sub-system shall be in CRun or CSleep

For more details on system states, please refer to the power controller (PWR) training slides.
The radio system is capable of waking up autonomously from Stop and Standby modes, and transferring data with the SRAM2. The shared bus matrix clock either uses the currently running sysclk, when the system is already in Run mode due to a CPU being active, or uses the HSI clock. The clock for the radio system is automatically enabled by the radio system itself; the HSI is used during radio startup and the HSE during radio TX and RX communication. The Reset and Clock Controller register bit STOPWakeUpClock must select the HSI as the wakeup clock source.
The SMPS needs a clock to operate in SMPS mode. In Bypass mode, no clock is needed. The clock to use and the frequency in SMPS mode can be selected between HSI, MSI, and HSE via the Reset and Clock Controller register bits SMPSSEL and SMPSDIV. The SMPS clock must be selected between 8 MHz and 2 MHz, where a higher clock frequency allows a minimum supply noise and a lower frequency allows the lowest power consumption. When the radio system is active, the HSE is forced as SMPS clock, in order for the SMPS artefacts to be synchronous with the carrier.
Wake up and startup

- At CPU and System power-on and system reset startup:
  - The MSI is selected as system clock. Other clocks and PLLs are disabled.

- When CPU and System wake up from Stop mode:
  - The HSI or MSI can be selected as system clock (STOPWUCK). Other clocks and PLLs are disabled.
    - HSI must be selected when SMPS mode or the Radio system is enabled
    - The HSI may be kept active during Stop to allow use as peripheral kernel clock. (HSIKERON)
      - In Stop 0 mode with SMPS mode enabled, the HSI must be kept active and selected as SMPS clock.

- When CPU and System wake up from Standby mode:
  - The HSI is selected as system clock. Other clocks and PLLs are disabled.

- When CPU wakes up from CStop mode with system in Run mode:
  - The CPU clocks will be the same as when entering CStop mode.

When the system restart the clock system is reset, all high-speed clocks and PLLs are disabled, except for the high-speed clock used to start up the system.
LSI and LSE are still working, if they were previously enabled.

After power on and a system reset, the MSI clock is enabled as system clock.

When waking up from Stop mode, the system clock can be selected between MSI or HSI with Reset and Clock Controller register bit STOPWUCK. When the SMPS mode or Radio system are enabled, the HSI must be selected. In Stop mode, the HSI clock may be kept active to allow its use as peripheral kernel clock. When using SMPS mode in Stop0 mode, the HSI clock needs to be kept active and selected as SMPS clock.

When waking up from Standby mode, the HSI clock is enabled as system clock.
When a CPU wakes up from CStop mode, while the system
remained in Run mode, the clock setting are maintained and the CPU will wake up with the same clock as when it entered CStop mode.
The system wakeup mode can be read from the Power Controller Stop and Standby flags. Refer to Power Controller training slides for more information.
The CPU CSleep mode does not affect the clock settings, but only plays on the CPU sub-system clock gating.
This slide lists the Reset and Clock Controller interrupts. The HSE and LSE clock security systems, the PLLs, and all 7 oscillator ready signals can generate an interrupt.

<table>
<thead>
<tr>
<th>Interrupt event</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSE clock security system</td>
<td>NMI</td>
<td>Set when a failure is detected in the HSE oscillator</td>
</tr>
<tr>
<td>LSE clock security system</td>
<td>IRQ</td>
<td>Set when a failure is detected in the LSE oscillator</td>
</tr>
<tr>
<td>PLL ready interrupt flag</td>
<td>IRQ</td>
<td>Clock ready caused by PLL lock</td>
</tr>
<tr>
<td>PLLSAI ready interrupt flag</td>
<td>IRQ</td>
<td>Clock ready caused by PLLSAI lock</td>
</tr>
<tr>
<td>HSE ready</td>
<td>IRQ</td>
<td>Clock ready caused by the HSE oscillator</td>
</tr>
<tr>
<td>HSI ready</td>
<td>IRQ</td>
<td>Clock ready caused by the HSI oscillator</td>
</tr>
<tr>
<td>MSI ready</td>
<td>IRQ</td>
<td>Clock ready caused by the MSI oscillator</td>
</tr>
<tr>
<td>HSI48 ready</td>
<td>IRQ</td>
<td>Clock ready caused by the HSI48 oscillator</td>
</tr>
<tr>
<td>LSE ready</td>
<td>IRQ</td>
<td>Clock ready caused by the LSE oscillator</td>
</tr>
<tr>
<td>LS1 ready</td>
<td>IRQ</td>
<td>Clock ready caused by the LS1 oscillator</td>
</tr>
<tr>
<td>LS2 ready</td>
<td>IRQ</td>
<td>Clock ready caused by the LS2 oscillator</td>
</tr>
</tbody>
</table>
In addition to this training, you may find the Power Control and Interrupt Controller trainings useful.