4.13 Clock tree configuration view

STM32CubeMX **Clock configuration** window (see *Figure 88*) provides a schematic overview of the clock paths, clock sources, dividers, and multipliers. Drop-down menus and buttons allow modifying the actual clock tree configuration to meet user application requirements.

Actual clock speeds are displayed and active. The clock signals that are used are highlighted in blue.

Out-of-range configured values are highlighted in red to flag potential issues. A solver feature is proposed to automatically resolve such configuration issues (see *Figure 89*).

Reverse path is supported: just enter the required clock speed in the blue filed and STM32CubeMX will attempt to reconfigure multipliers and dividers to provide the requested value. The resulting clock value can then be locked by right clicking the field to prevent modifications.

STM32CubeMX generates the corresponding initialization code:

- main.c with relevant HAL_RCC structure initializations and function calls
- stm32xxxx_hal_conf.h for oscillator frequencies and V_{DD} values.

4.13.1 Clock tree configuration functions

External clock sources

When external clock sources are used, the user must previously enable them from the **Pinout** view available under the RCC peripheral.

Peripheral clock configuration options

Some other paths, corresponding to clock peripherals, are grayed out. To become active, the peripheral must be properly configured in the **Pinout** view (e.g. USB). This view allows to:

- **Enter a frequency value for the CPU Clock (HCLK), buses or peripheral clocks**

  STM32CubeMX tries to propose a clock tree configuration that reaches the desired frequency while adjusting prescalers and dividers and taking into account other peripheral constraints (such as USB clock minimum value). If no solution can be found,

STM32CubeMX proposes to switch to a different clock source or can even conclude that no solution matches the desired frequency.

- **Lock the frequency fields for which the current value should be preserved**

  Right click a frequency field and select **Lock** to preserve the value currently assigned when STM32CubeMX will search for a new clock configuration solution.

  The user can unlock the locked frequency fields when the preservation is no longer necessary.

- **Select the clock source that will drive the system clock (SYSCLK)**
  - External oscillator clock (HSE) for a user defined frequency.
  - Internal oscillator clock (HSI) for the defined fixed frequency.
  - Main PLL clock
- **Select secondary sources (as available for the product)**
  - Low-speed internal (LSI) or external (LSE) clock
  - I2S input clock
  - …
- **Select prescalers, dividers and multipliers values.**
- **Enable the Clock Security system (CSS) on HSE when it is supported by the MCU**

  This feature is available only when the HSE clock is used as the system clock source directly or indirectly through the PLL. It allows detecting HSE failure and inform the software about it, thus allowing the MCU to perform rescue operations.

- **Enable the CSS on LSE when it is supported by the MCU**

  This feature is available only when the LSE and LSI are enabled and after the RTC or LCD clock sources have been selected to be either LSE or LSI.

- **Reset the Clock tree default settings by using the toolbar Reset button (**  **):**

  This feature reloads STM32CubeMX default clock tree configuration.

- **Undo/Redo user configuration steps by using the toolbar Undo/Redo buttons (** **)**
- **Detect and resolve configuration issues**

  Erroneous clock tree configurations are detected prior to code generation. Errors are highlighted in red and the **Clock Configuration** view is marked with a red cross (see *Figure 89*).

  Issues can be resolved manually or automatically by clicking the **Resolve Clock Issue** button () which is enabled only if issues have been detected.

  The underlying resolution process follows a specific sequence:

  a) Setting HSE frequency to its maximum value (optional).

  b) Setting HCLK frequency then peripheral frequencies to a maximum or minimum value (optional).

  c) Changing multiplexers inputs (optional).

  d) Finally, iterating through multiplier/dividers values to fix the issue. The clock tree is cleared from red highlights if a solution is found. Otherwise an error message is displayed.

*Note: To be available from the clock tree, external clocks, I2S input clock, and master clocks shall be enabled in RCC configuration in the **Pinout** view. This information is also available as tooltips.*

The tool will automatically perform the following operations:

- Adjust bus frequencies, timers, peripherals and master output clocks according to user selection of clock sources, clock frequencies and prescalers/multipliers/dividers values.
- Check the validity of user settings.
- Highlight invalid settings in red and provide tooltips to guide the user to achieve a valid configuration.

The Clock tree view is adjusted according to the RCC settings (configured in RCC IP pinout and configuration views) and vice versa:

- If in RCC **Pinout** view, the external and output clocks are enabled, they become configurable in the clock tree view.
- If in RCC Configuration view, the Timer prescaler is enabled, the choice of Timer clocks multipliers will be adjusted.

Conversely, the clock tree configuration may affect some RCC parameters in the configuration view:

- Flash latency: number of wait states automatically derived from V_{DD} voltage, HCLK frequency, and power over-drive state.
- Power regulator voltage scale: automatically derived from HCLK frequency.
- Power over-drive is enabled automatically according to HCLK frequency. When the power drive is enabled, the maximum possible frequency values for AHB and APB domains are increased. They are displayed in the Clock tree view.

The default optimal system settings that is used at startup are defined in the *system_stm32f4xx.c* file. This file is copied by STM32CubeMX from the STM32CubeF4 firmware package. The switch to user defined clock settings is done afterwards in the main function.

Figure 88 gives an example of Clock tree configuration view for an STM32F429x MCU and *Table 12* describes the widgets that can be used to configure each clock.

Figure 88. STM32F429xx Clock Tree configuration view

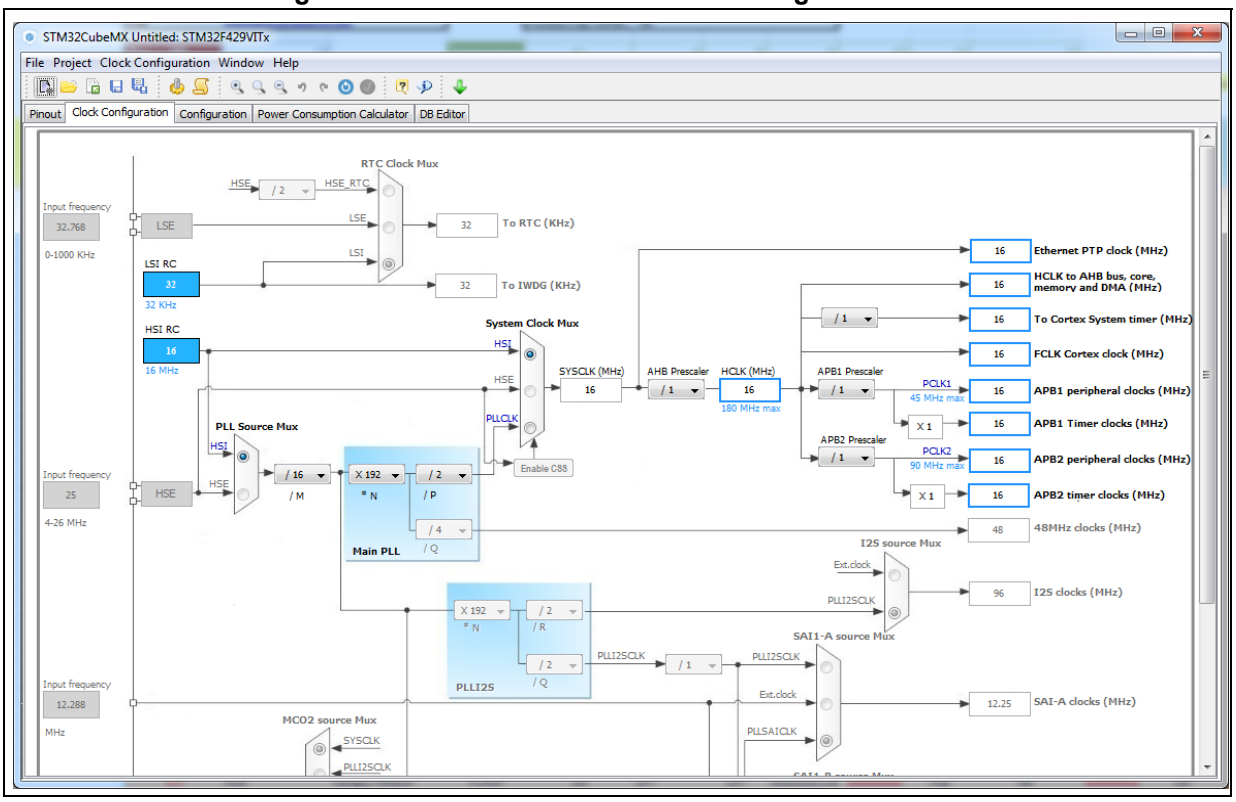


Figure 89. Clock Tree configuration view with errors

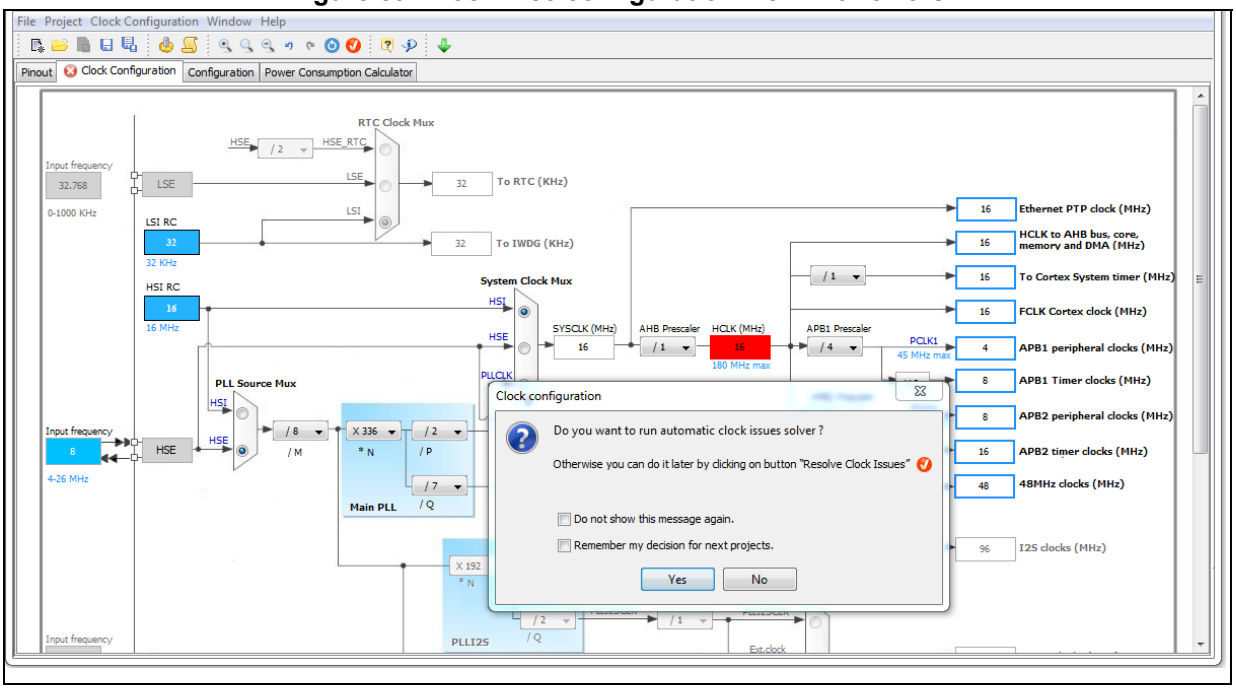
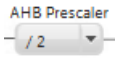
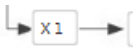
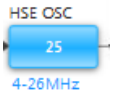
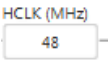
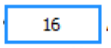
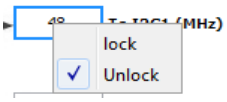


Table 12. Clock tree view widget

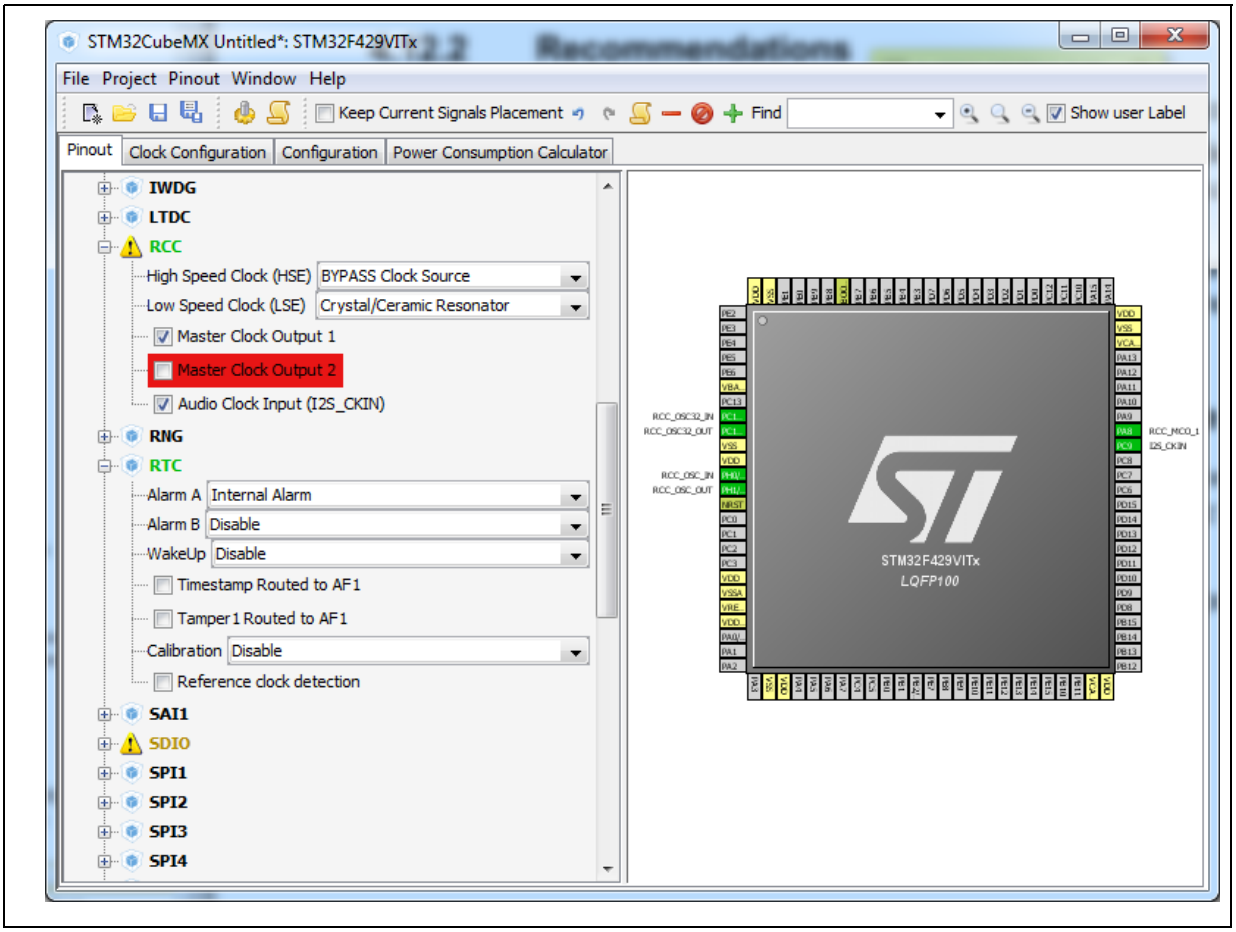
Format	Configuration status of the Peripheral Instance
	Active clock sources
	Unavailable settings are blurred or grayed out (clock sources, dividers,…)
	Gray drop down lists for prescalers, dividers, multipliers selection.
	Multiplier selection
	User defined frequency values
	Automatically derived frequency values
	User-modifiable frequency field
	Right click blue border rectangles, to lock/unlock a frequency field. Lock to preserve the frequency value during clock tree configuration updates.

4.13.2 Recommendations

The **Clock tree** view is not the only entry for clock configuration.

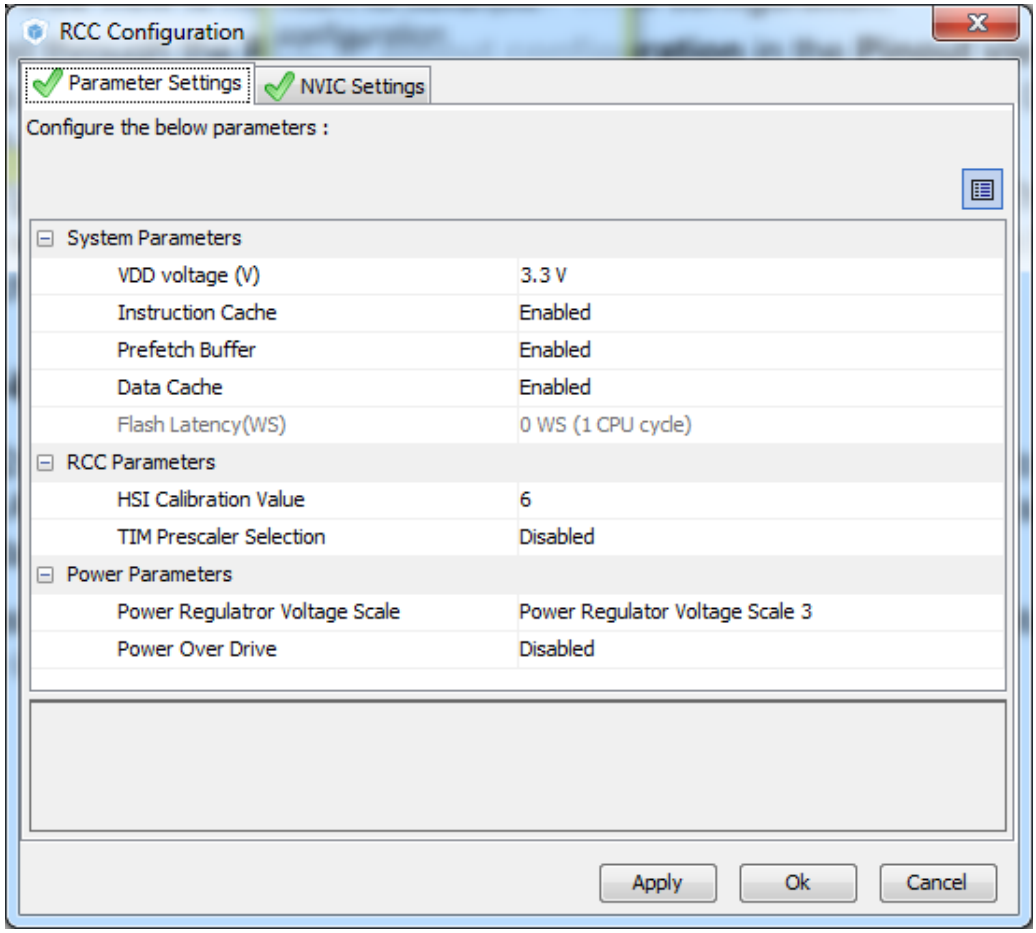
1. Go first through the **RCC IP pinout configuration** in the **Pinout** view to enable the clocks as needed: external clocks, master output clocks and Audio I2S input clock when available (see *Figure 90*).

Figure 90. Clock tree configuration: enabling RTC, RCC Clock source and outputs from Pinout view



2. Then go to the **RCC IP configuration** in the **Configuration view**. The settings defined there for advanced configurations will be reflected in the **clock tree view.** The settings defined in the clock tree view may change the settings in the RCC configuration (see *Figure 91*).

Figure 91. Clock tree configuration: RCC Peripheral Advanced parameters



4.13.3 STM32F43x/42x power-over drive feature

STM32F42x/43x MCUs implement a power over-drive feature allowing to work at the maximum AHB/APB bus frequencies (e.g., 180 MHz for HCLK) when a sufficient V_{DD} supply voltage is applied (e.g V_{DD} > 2.1 V).

Table 13 lists the different parameters linked to the power over-drive feature and their availability in STM32CubeMX user interface.

Table 13. Voltage scaling versus power over-drive and HCLK frequency

Parameter	STM32CubeMX panel	Value
V_{DD} voltage	Configuration (RCC)	User-defined within a predefined range. Impacts power over-drive.
Power Regulator Voltage scaling	Configuration (RCC)	Automatically derived from HCLK frequency and power over-drive (see *Table 14*).
Power Over Drive	Configuration (RCC)	This value is conditioned by HCLK and V_{DD} value (see *Table 14*). It can be enabled only if $V_{DD} \geq 2.2$ V When $V_{DD} \geq 2.2$ V, it is either automatically derived from HCLK or it can be configured by the user if multiple choices are possible (e.g., HCLK = 130 MHz)
HCLK/AHB clock maximum frequency value	Clock Configuration	Displayed in blue to indicate the maximum possible value. For example: maximum value is 168 MHz for HCLK when power over-drive cannot be activated (when $V_{DD} \leq 2.1$ V), otherwise it is 180 MHz.
APB1/APB2 clock maximum frequency value	Clock Configuration	Displayed in blue to indicate maximum possible value

Table 14 gives the relations between power-over drive mode and HCLK frequency.

Table 14. Relations between power over-drive and HCLK frequency

HCLK frequency range: V_{DD} > 2.1 V required to enable power over-drive (POD)	Corresponding voltage scaling and power over-drive (POD)
≤120 MHz	Scale 3 POD is disabled
120 to 14 MHz	Scale 2 POD can be either disabled or enabled
144 to 168 MHz	Scale 1 when POD is disabled Scale 2 when POD is enabled
168 to 180 MHz	POD must be enabled Scale 1 (otherwise frequency range not supported)

4.13.4 Clock tree glossary

Table 15. Glossary

Acronym	Definition
HSI	High Speed Internal oscillator: enabled after reset, lower accuracy than HSE.
HSE	High Speed External oscillator: requires an external clock circuit.
PLL	Phase Locked Loop: used to multiply above clock sources.
LSI	Low Speed Internal clock: low power clocks usually used for watchdog timers.
LSE	Low Speed External clock: powered by an external clock.
SYSCLK	System clock
HCLK	Internal AHB clock frequency
FCLK	Cortex free running clock
AHB	Advanced High Performance Bus
APB1	Low speed Advanced Peripheral Bus
APB2	High speed Advanced Peripheral Bus

4.14 Power Consumption Calculator (PCC) view

For an ever-growing number of embedded systems applications, power consumption is a major concern. To help minimizing it, STM32CubeMX offers the **Power Consumption Calculator** (PCC) tab (see *Figure 92*), which, given a microcontroller, a battery model and a user-defined power sequence, provides the following results:

- Average current consumption

  Power consumption values can either be taken from the datasheet or interpolated from a user specified bus or core frequency.
- Battery life
- Average DMIPs

  DMIPs values are directly taken from the MCU datasheet and are neither interpolated nor extrapolated.
- Maximum ambient temperature (T_{AMAX})

  According to the chip internal power consumption, the package type and a maximum junction temperature of 105 °C, the tool computes the maximum ambient temperature to ensure good operating conditions.

  Current T_{AMAX} implementation does not account for I/O consumption. For an accurate T_{AMAX} estimate, I/O consumption must be specified using the Additional Consumption field. The formula for I/O dynamic current consumption is specified in the microcontroller datasheet.

The **PCC** view allows developers to visualize an estimate of the embedded application consumption and lower it further at each PCC power sequence step:

- Make use of low power modes when any available
- Adjust clock sources and frequencies based on the step requirements.
- Enable the peripherals necessary for each phase.

For each step, the user can choose VBUS as possible power source instead of the battery. This will impact the battery life estimation. If power consumption measurements are available at different voltage levels, STM32CubeMX will also propose a choice of voltage values (see *Figure 96*).

An additional option, the transition checker, is available for STM32L0, STM32L1 and STM32L4 series. When it is enabled, the transition checker detects invalid transitions within the currently configured sequence. It ensures that only possible transitions are proposed to the user when a new step is added.

4.14.1 Building a power consumption sequence

The default starting view is shown in *Figure 92*.

Figure 92. Power Consumption Calculator default view