AN5042
Application note

How to calibrate the HSE clock for RF applications on STM32 wireless MCUs

Introduction

This document describes how to tune the HSE for RF applications using STM32WL, STM32WB, and STM32WBA series microcontrollers (hereinafter referred to as STM32 wireless MCUs). These products offer a cost-effective and efficient solution to control the oscillator accuracy by using their internal load capacitances, thus saving the cost of external capacitances, and lowering the crystal constraints.

STM32 wireless MCUs use an external oscillator high-speed clock source as the base for RF clock generation. HSE accuracy is essential for RF system performance, and the external oscillator is therefore fine-tuned to achieve the highest clock accuracy.

The first part of the document introduces the crystal oscillator solutions. The second part introduces and then compares three HSE frequency tuning methods, namely a manual one, an automatic one, and another one based on STM32CubeMonitor-RF (for the STM32WB series only). The application of these methods to Nucleo boards is described in the following sections, and provided as firmware and script examples in X-CUBE-CLKTRIM, an STM32Cube Expansion Package.

A specific section is dedicated to describe how to select the crystal for STM32WB series microcontrollers.

This document must be read in conjunction with the reference manuals and datasheets available at *www.st.com*.

Contents

List of tables

List of figures

1 HSE oscillator

RF systems require high frequency accuracy to achieve the best performance. Any clock deviation can cause system malfunctions and/or degrade performance.

Table 1 shows the accuracy requirements for two RF protocols supported by STM32WB series microcontrollers. For other protocols and standards, refer to the corresponding specifications.

Table 1. Carrier accuracy requirement for RF protocols

RF standard	Carrier accuracy
Bluetooth® Low Energy	± 50 ppm
IEEE 802.15.4 / Thread	± 40 ppm

In STM32 wireless MCUs, based on Arm®(a) Cortex® cores, the RF clock is provided by a high frequency VCO, which takes as reference a signal created by an embedded oscillator that uses an external crystal.

This crystal is the HSE (high-speed external) clock source of the RF synthesizer and of the microcontroller. Its nominal frequency can vary, depending on factors such as process variations, used crystal, and PCB design. The HSE inaccuracy is directly transferred to the RF clock, hence it must be fine-tuned by adjusting load capacitance at crystal terminals.

STM32 wireless MCUs offer an efficient architecture with internal load capacitances, which allows the users to fine tune the crystal frequency, without extra cost for additional external capacitances.

Note: AN2867 (Oscillator design guide for ST microcontrollers), which generally describes HSE for STM32 products, does not apply to STM32 wireless MCUs because of RF constraints. This document is the correct reference for these products.

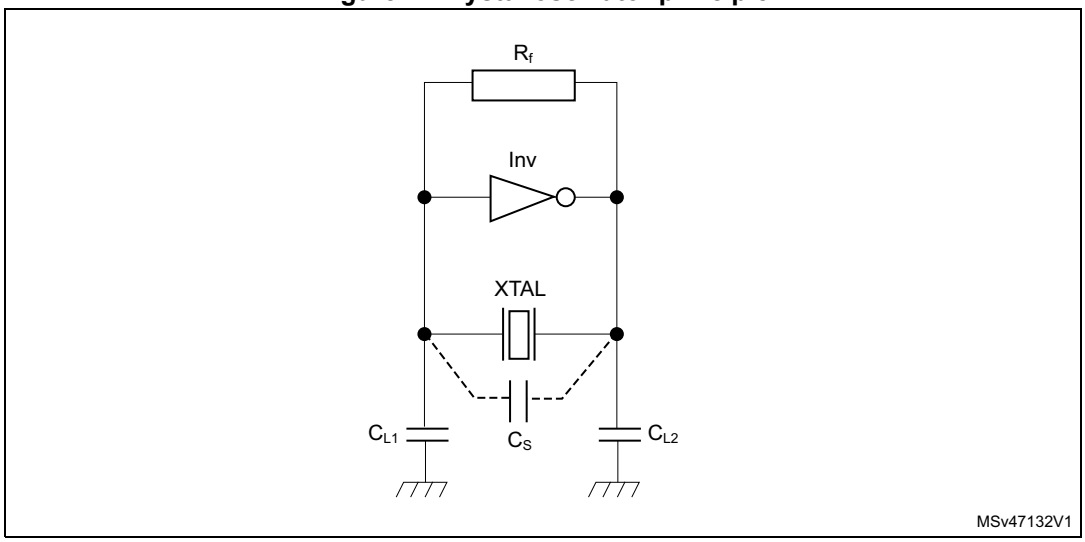
arm

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

1.1 Crystal oscillator

Figure 1 shows the principle of the crystal oscillator system. An oscillator consists of an inverting amplifier, a feedback resistor (R_f), a crystal (XTAL), and two load capacitors (C_{L1} and C_{L2}). C_s is the stray (parasitic) capacitance, resulting from the sum of the MCU pin capacitances (between OSC_IN and OSC_OUT). This value is negligible, it is not necessary to know it exactly, because trimming and startup optimization procedures do not rely on it.

Figure 1. Crystal oscillator principle



C_L load capacitance

The load capacitance is the terminal capacitance of the circuit connected to the crystal oscillator. This value is determined by the external capacitors C_{L1} and C_{L2}, and the stray capacitance of the printed circuit board and connections (C_s). The C_L value is specified by the crystal manufacturer. For the frequency to be accurate, the oscillator circuit must show to the crystal the same load capacitance as the one the crystal was adjusted for.

Frequency stability requires that the load capacitance be constant. The external trimming capacitors C_{L1} and C_{L2} are used to tune the desired value of C_L to reach the value specified by the crystal manufacturer.

Equation 1: Load capacitance

$$C_L = \frac{C_{L1} \times C_{L2}}{C_{L1} + C_{L2}} + C_s$$

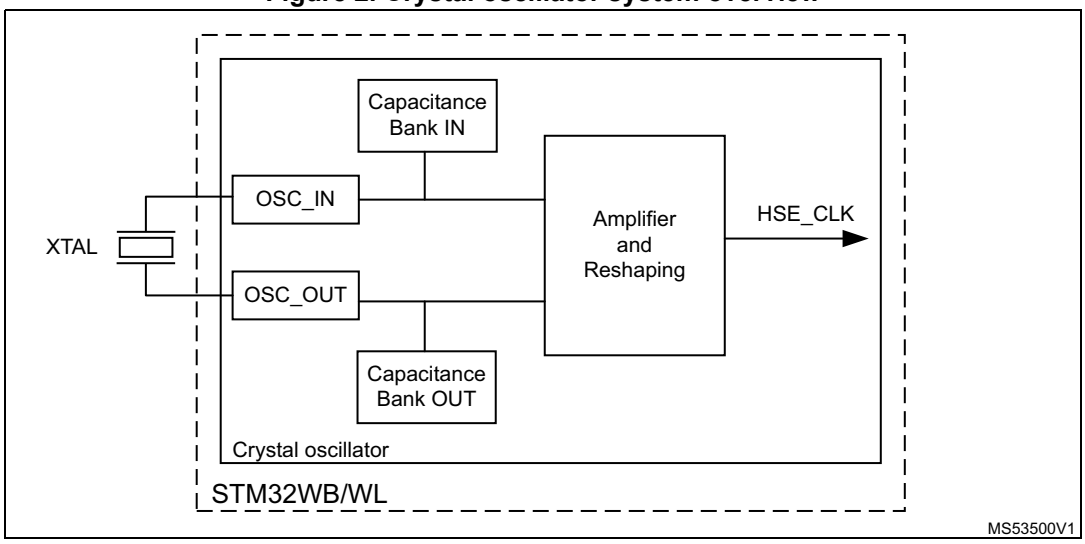
1.2 STM32 wireless MCUs architecture

These MCUs embed an efficient and cost-effective crystal oscillator system with internal capacitances for trimming. The advantages of the internal mechanism for load capacitance tuning are twofold:

- it reduces the accuracy constraints on the external crystal
- it reduces the global BOM (and the footprint) of the PCB.

Figure 2 shows the crystal oscillator system embedded in the STM32 wireless MCUs. The crystal is the only external component. No extra load capacitances are needed.

Figure 2. Crystal oscillator system overview



The crystal oscillator system consists of two pads (OSC_IN and OSC_OUT) with their respective capacitance banks, and the amplifier stage.

For the STM32WB and STM32WBA series, the capacitance value is the same for both the IN and the OUT banks. This value, alongside with the oscillator gain and sense (the parameters used to optimize the startup phase), is driven by a register, and controls the system behavior. These parameters are explained in *Section 1.3* and *Section 1.4*.

For the STM32WL series, the capacitance values for the IN and the OUT banks are independent. These two values are driven by two sub-GHz radio registers and control the system behavior. These parameters are explained in *Section 1.5*.

1.3 HSE configuration parameters - STM32WB series

Three parameters can be set to control the oscillator module. They are accessible in the RCC_HSECR register described below.

RCC_HSECR

Address 0x09C

Reset 0x0000 0030

Access This register is protected to avoid on-the-fly modification.
A key (0xCAFECAFE) must be written at the register address to unlock it before any single write access, it is then locked again.
The HSE clock must be switched off during register access procedure to avoid unpredictable behavior. Note that HSE must not be used as CPU clock source during this step.
In this document the default MSI clock is used as system clock source after startup from Reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	HSETUNE[5:0]						Res.	HSEGMC[2:0]			HSES	Res.	Res.	UNLOCKED
		rw	rw	rw	rw	rw	rw		rw	rw	rw	rw			rw

Load capacitance: HSETUNE[5:0]

This is the parameter responsible for the clock accuracy. It selects the capacitance value added on both input and output pads. The adjustable range is set to have a global load capacitance from 12 to 16 pf. The minimum (0x00) and maximum (0x3F) values correspond, respectively, to the smallest and to the largest load capacitance.

Default value is 0x00 (minimum load capacitance).

Current control: HSEGMC[2:0]

This parameter (referred to as Gm or Gm_crit_max in the documentation of other STM32 products for which it has a fixed value) is the maximum critical crystal transconductance of the oscillator. It controls the startup performance of the system, and its value must be greater than g_{mcrit} (see *Section 11*). A low value decreases the power consumption, while a high value improves the startup time.

The minimum value (0b000) corresponds to Gm = 0.18 mA/V, the maximum value (0b111) to Gm = 2.84 mA/V. Default value is 0x3 (Gm = 1.13 mA/V).

HSES sense amplifier threshold

This parameter controls an internal comparison threshold for the oscillator startup. When this bit is set (1), the startup time is reduced (from around 15 µs), but the current consumption is higher, because the HSE starts earlier. The default value is 0x0 (1/2 ratio).

1.4 HSE configuration parameters - STM32WBA series

Three parameters can be set to control the oscillator module. They are accessible in the RCC external clock sources calibration register 1 (RCC_ECSCR1) described below.

RCC_ECSCR1

Address offset: 0x210

Power-on reset value: 0x0020 0000

Reset value not effected by exit Standby mode, nor reset from system reset and BORx (x = 1 to 4).

Access: no wait state; word, half-word and byte access

Access to this register can be protected by RCC HSESEC and RCC SPRIV or RCC NSPRIV.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HSETRIM[5:0]					
										rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

Bits 31:22 Reserved, must be kept at reset value.

Bits 21:16 **HSETRIM[5:0]**: HSE32 clock trimming

These bits provide user-programmable capacitor trimming value. It can be programmed to adjust the HSE32 oscillator frequency.

Bits 15:0 Reserved, must be kept at reset value.

1.5 HSE configuration parameters - STM32WL series

For the STM32WL series two parameters can be set to control the oscillator module. They are accessible in the SUBGHZ_HSEINTRIMR and the SUBGHZ_HSEOUTTRIMR sub-GHz radio registers, which contain the capacitance value of the IN and OUT banks, respectively. The associated value is represented by their six lower bits (the remaining bits of these registers must be kept at their reset values).

For both registers:

- 0x00 corresponds to the minimum capacitance (~11.3 pF)
- 0x2F corresponds to the maximum capacitance (~33.4 pF)
- the values must not exceed 0x2F, and the trimming step is ~0.47 pF
- the reset value is 0x12, corresponding to ~20.3 pF.

As mentioned earlier, SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR are part of the sub-GHz radio, not of the system CPU. To modify their values, the user code must communicate with the sub-GHz radio via its SPI interface. The addresses of these registers, considering this SPI interface, are 0x911 for SUBGHZ_HSEINTRIMR and 0x912 for SUBGHZ_HSEOUTTRIMR.

1.6 Board implementation (STM32WB series)

Oscillator pads are available on different pins (named OSC_IN and OSC_OUT), depending upon the package. *Table 2* shows the pin numbers for different packages used in Nucleo and USB dongle boards for the STM32WB series.

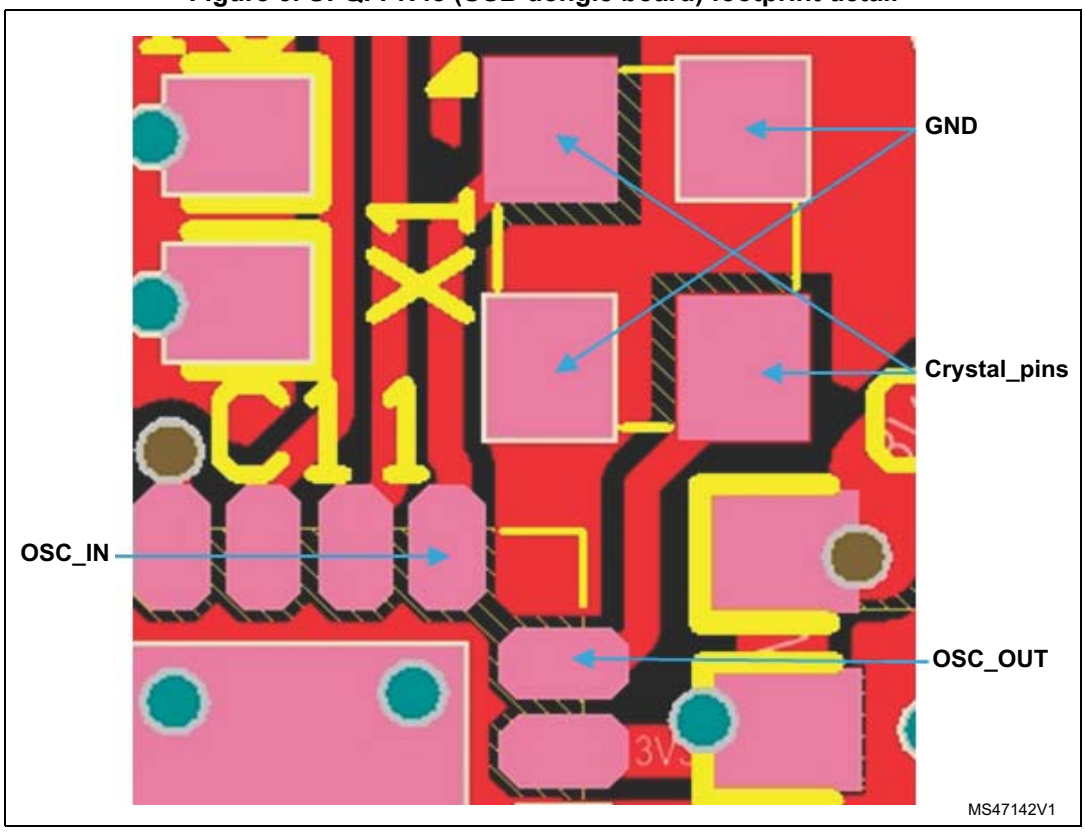
Table 2. Oscillator pin numbers for the STM32WB series

Package	OSC_IN	OSC_OUT
UFQFPN48	25	24
WLCSP49	F1	F2
VFQFPN68	35	34
WLCSP100	J1	J2
BGA129	M13	N13

The crystal is plugged directly onto the pads, with no extra capacitance, close to the device, to minimize parasitic capacitances.

Figure 3 shows a typical UFQFPN48 footprint for the STM32WB series.

Figure 3. UFQFPN48 (USB dongle board) footprint detail



The PCB layout for the STM32WB and STM32WL series reference designs is detailed, respectively, in AN5165 "*Development of RF hardware using STM32WB microcontrollers*", and AN5407 "*Optimized RF board layout for STM32WL Series*", both available on *www.st.com*.

1.7 Board implementation (STM32WBA series)

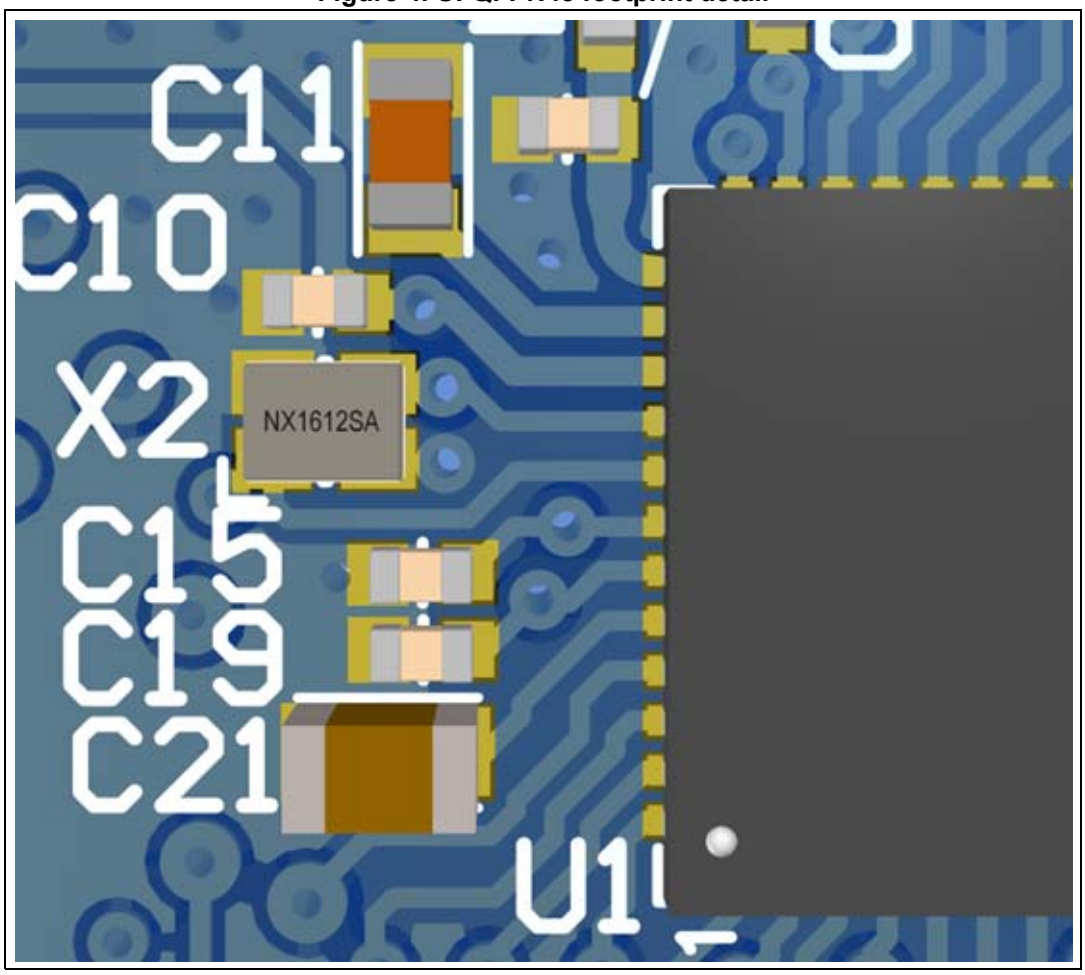
Oscillator pads are available on different pins (named OSC_IN and OSC_OUT), depending upon the package. *Table 3* shows the pin numbers used in Nucleo and USB dongle boards.

Table 3. Oscillator pin numbers for the STM32WBA series

Package	OSC_IN	OSC_OUT
UFQFPN48	41	40

The HSE crystal is connected directly to the pads, with no extra capacitance, close to the device, to minimize parasitic capacitances. *Figure 4* shows a typical footprint.

Figure 4. UFQFPN48 footprint detail



The PCB layout for the reference design is detailed in AN5948 "*Development of RF hardware using STM32WBA microcontrollers*", available on *www.st.com*.

1.8 Crystal references

Table 4 shows the specification of the crystals used to validate the reference designs.

Table 4. Crystal specifications

Series	NDK crystal	Parameter	Value
STM32WB	NX2016SA 32 MHz EXS00A-CS06654	Load capacitance	8 pF
		Frequency tolerance	± 10 ppm at 25 ± 3 °C
		Frequency vs. temperature (with reference to +25 °C)	± 25 × 10^{-6}
STM32WBA	NX612SA 32MHz	Load capacitance	8 pF
		Frequency tolerance	± 10 ppm at 25 ± 3 °C
		Frequency vs. temperature (with reference to +25 °C)	± 15 × 10^{-6}
STM32WL	NX2016SA 32 MHz EXS00A-CS06465	Load capacitance	10 pF
		Frequency tolerance	± 10 ppm at 25 °C

It is possible to use other crystals, as long as they respect the carrier accuracy requirements (see *Table 1* and the HSE crystal requirements in the datasheets). More precisely, the sum of the maximum values for the following parameters of the chosen crystal must be below the carrier accuracy requirements:

- frequency tolerance
- frequency versus temperature characteristics (for the range in which the crystal is used)
- aging (for the duration during which the crystal must be used)
- crystal pullability (the C_L deviation can be estimated to 10% maximum, refer to AN2867)

Note: Such data can be named differently or absent in the crystal documentation.

The previous verification is reliable if the PCB based on the chosen crystal is designed with the requirements mentioned in *Section 1.6*, and fine-tuned with one of the HSE trimming methods described in the next sections of this document.

For STM32WB devices, in the case of crystals with low load capacitance (such as 6 pF), if the parasitic layout capacitance is not negligible, frequency fine tuning to 32 MHz with 0 ppm can be difficult to achieve (the trimming range is centered on crystals with a load capacitance of 8 pF).

The accuracy of the capacitance bank used for the HSE trimming can vary from a product to another. If the carrier accuracy verification is needed for the whole production chain, refer to *Section 1.9*.

1.9 Tuning in production

It is important to know if the trimming parameters defined for a test PCB have the same efficiency when applied to all the PCBs to be produced. To have a fast and efficient manufacturing, it is not mandatory to operate the trimming process on each PCB.

Data gathered during the production of the STM32WB Nucleo boards have shown some variations of the HSE frequency for a fixed HSETUNE value (HSETUNE accuracy). Besides, the typical HSETUNE granularity is 1 ppm (see XOTUNE granularity in STM32WB

datasheets). To compensate for these variations, the following trimming method is suggested for production:

1. Tune some PCBs (not more than a few dozens), to get a significant span of HSETUNE values.
2. Calculate the median of the found values, and use the result to trim all other PCBs.
3. The interval between the median and the maximum/minimum of the found values corresponds to the HSETUNE accuracy for the tuned design.

After this flow, a potential frequency deviation due to the HSETUNE accuracy (point *3*) must be taken into account, and added to the carrier accuracy verification calculation presented in *Section 1.8*.

A similar method can be used for the other series. The number of test boards must be larger, because of the two trimming parameters for these products.

2 Trimming methods comparison

The three trimming methods described in this document are detailed in *Table 5*, and compared in *Table 6*.

Table 5. Trimming methods

Method	Description
Manual	A precision frequency meter is used to measure the HSE frequency output on one of the STM32 pins. Then, the user tunes the HSE frequency with the buttons of a Nucleo board. A button is dedicated to the saving of the tuning parameters in the STM32 nonvolatile memory.
Automatic	One STM32 timer is clocked with a precision external clock source provided by the user via one of the STM32 pins. This reference clock allows the user to measure the internal STM32 system HSE frequency. Then, the STM32 can compare the frequency measured with the one expected, to test and determine the best tuning parameters. Finally, the STM32 saves these parameters in its nonvolatile memory.
STM32CubeMonitor-RF(1)	A precision frequency meter is used to measure the HSE frequency output on one of the STM32 pins. Then, the user tunes the HSE frequency with a script to run in STM32CubeMonitor-RF. The user has to change the tuning parameter values in the script to test them. When the correct values are found, another script saves them in the STM32 nonvolatile memory.

1. Compatible with STM32WB series only.

Table 6. Comparison of trimming methods

Method	Advantages	Disadvantages
Manual	Runs in SRAM (user program in flash memory is not affected).	Needs the use of buttons and a frequency meter for each product.
Automatic	Runs in SRAM (user program in flash memory is not affected). The user must set up a reference clock only once, to trim as many devices as wanted.	Needs two timers with adequate accuracy.
STM32CubeMonitor-RF(1)	Convenient for users familiar with STM32CubeMonitor-RF, willing to achieve the maximum functionality.	BLE stack and transparent mode FW must be flashed in the device. Requires several actions from the user for each product (modifying and running the script, use of a frequency meter).

1. Compatible only with the STM32WB series.

Note: The proposed methods require the use of a pin (MCO) to output the HSE signal for the frequency measurement. If no MCO pin is available for this purpose, AN5378 (available on www.st.com) describes another technique: instead of measuring the frequency on the MCO pin, it proposes to perform it on a tone frequency emitted by the STM32WB radio. However,

while this tone is emitted, HSE cannot be trimmed, because it is used by the radio (actually this is more a verification than a real trimming method, like those exposed in this document).

3 Manual frequency trimming procedure example for the STM32WB series

The firmware and scripts associated to this document are available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\P-NUCLEO-WB55.Nucleo\RCC_HSE_Calib).

3.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. This clock is output on pin PA8 and is measured by a precision frequency meter.

Note: An external reference is mandatory since no such accurate one is integrated in the device.

A step-by-step tuning of the load capacitance is performed to reach the best accuracy of the HSE clock. The load capacitance value is then stored inside a nonvolatile location of the device, either a dedicated area of the user flash memory or in the One-Time-Programming (OTP) area.

Flash memory or OTP programming is done with a double-word granularity (64 bits). To save OTP bytes (1 K in the STM32WB series), the load capacitance value on six bits can be appended to a 64-bit wide structure with other personalization data (such as Bluetooth device address, MAC short address, product specific code, key).

This procedure can be done several times, only the latest setup is active.

Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function), and the HSE configuration register set accordingly.

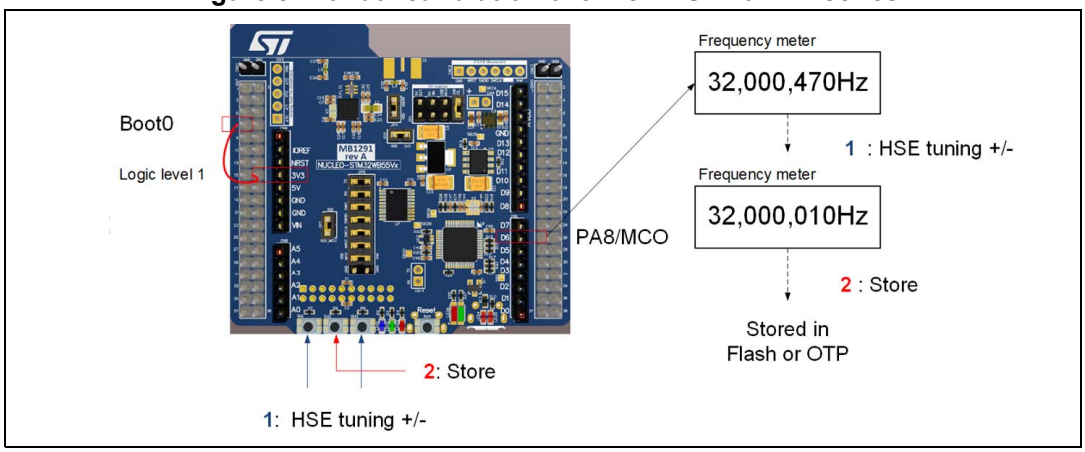
3.2 Implementation

The procedure is executed in SRAM, so it can be run on an already programmed device, without modifying the flash memory content.

3.2.1 Hardware setup

Figure 5 shows the manual calibration procedure for an STM32WB series Nucleo-68 board.

Figure 5. Manual calibration overview - STM32WB series



A precision frequency meter (better than 0.1 ppm) must be connected to pin PA8/MCO, and set to detect a 32 MHz 3.3 V square wave peak to peak signal.

Note: Standard oscilloscopes are not sufficiently accurate for this kind of measurement.

Boot from SRAM

Boot selection can be done through the BOOT0 pin and nBOOT1 bit in the User options (FLASH_OPTR).

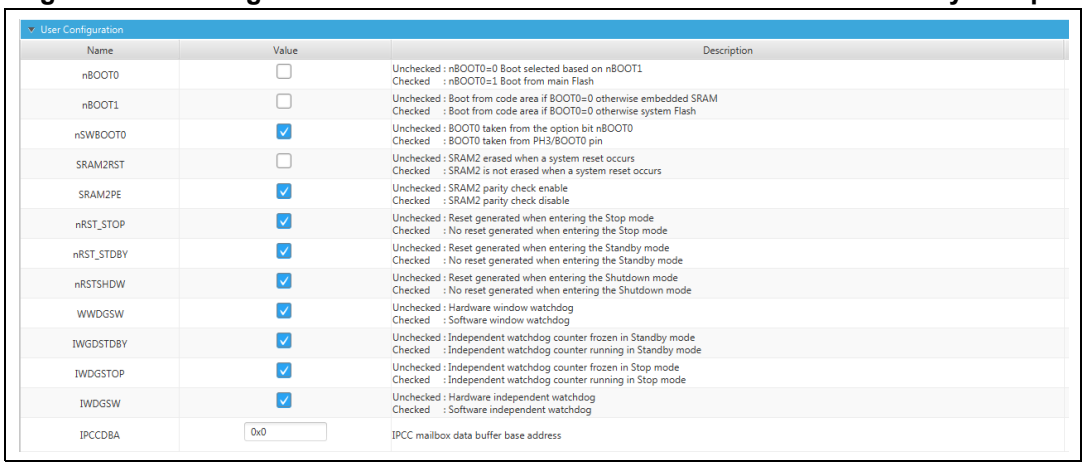
The boot from SRAM configuration is set by both Boot0 = 1 and nBoot1 = 0 conditions, nBoot1 is set only by option bit FLASH_OPTR[23].

Boot0 can be selected:

- through value of pin PH3 at startup if option bit nSWBOOT0 = 1 (FLASH_OPTR[26] = 1), see the option byte panel in *Figure 6*
- through option bit value nBOOT0 if option bit nSWBOOT0 = 0 (FLASH_OPTR[26] = 0), see the option byte panel in *Figure 7*

Option bits can be selected through STM32CubeProgrammer.

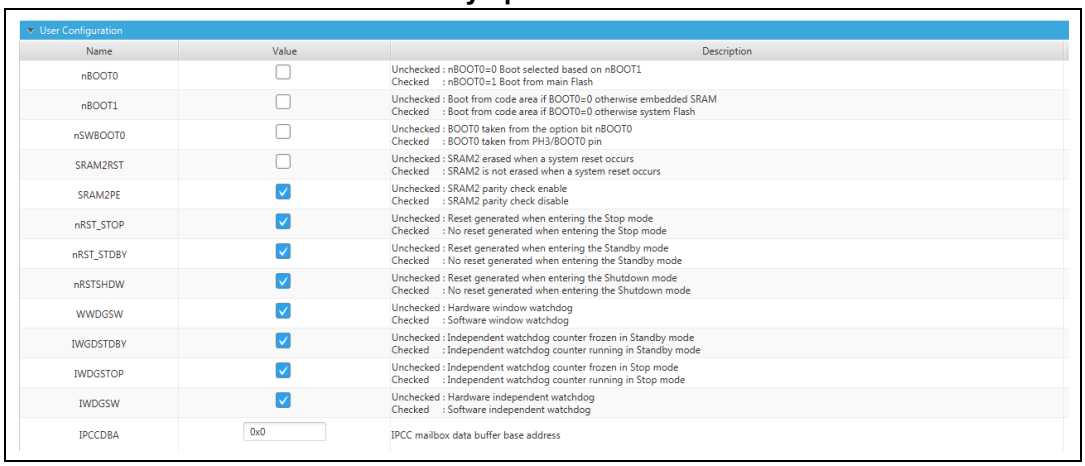
Figure 6. OB configuration to boot from SRAM with BOOT0 value driven by PH3 pin



User Configuration

Name	Value	Description
nBOOT0	☐	Unchecked : nBOOT0=0 Boot selected based on nBOOT1 Checked : nBOOT0=1 Boot from main Flash
nBOOT1	☐	Unchecked : Boot from code area if BOOT0=0 otherwise embedded SRAM Checked : Boot from code area if BOOT0=0 otherwise system Flash
nSWBOOT0	☑	Unchecked : BOOT0 taken from the option bit nBOOT0 Checked : BOOT0 taken from PH3/BOOT0 pin
SRAM2RST	☐	Unchecked : SRAM2 erased when a system reset occurs Checked : SRAM2 is not erased when a system reset occurs
SRAM2PE	☑	Unchecked : SRAM2 parity check enable Checked : SRAM2 parity check disable
nRST_STOP	☑	Unchecked : Reset generated when entering the Stop mode Checked : No reset generated when entering the Stop mode
nRST_STDBY	☑	Unchecked : Reset generated when entering the Standby mode Checked : No reset generated when entering the Standby mode
nRSTSHDW	☑	Unchecked : Reset generated when entering the Shutdown mode Checked : No reset generated when entering the Shutdown mode
WWDGSW	☑	Unchecked : Hardware window watchdog Checked : Software window watchdog
IWGDSTDBY	☑	Unchecked : Independent watchdog counter frozen in Standby mode Checked : Independent watchdog counter running in Standby mode
IWDGSTOP	☑	Unchecked : Independent watchdog counter frozen in Stop mode Checked : Independent watchdog counter running in Stop mode
IWDGSW	☑	Unchecked : Hardware independent watchdog Checked : Software independent watchdog
IPCCDBA	0x0	IPCC mailbox data buffer base address

Figure 7. OB configuration to boot from SRAM with BOOT0 value driven by option bit nBOOT0



User Configuration

Name	Value	Description
nBOOT0	☐	Unchecked : nBOOT0=0 Boot selected based on nBOOT1 Checked : nBOOT0=1 Boot from main Flash
nBOOT1	☐	Unchecked : Boot from code area if BOOT0=0 otherwise embedded SRAM Checked : Boot from code area if BOOT0=0 otherwise system Flash
nSWBOOT0	☐	Unchecked : BOOT0 taken from the option bit nBOOT0 Checked : BOOT0 taken from PH3/BOOT0 pin
SRAM2RST	☐	Unchecked : SRAM2 erased when a system reset occurs Checked : SRAM2 is not erased when a system reset occurs
SRAM2PE	☑	Unchecked : SRAM2 parity check enable Checked : SRAM2 parity check disable
nRST_STOP	☑	Unchecked : Reset generated when entering the Stop mode Checked : No reset generated when entering the Stop mode
nRST_STDBY	☑	Unchecked : Reset generated when entering the Standby mode Checked : No reset generated when entering the Standby mode
nRSTSHDW	☑	Unchecked : Reset generated when entering the Shutdown mode Checked : No reset generated when entering the Shutdown mode
WWDGSW	☑	Unchecked : Hardware window watchdog Checked : Software window watchdog
IWGDSTDBY	☑	Unchecked : Independent watchdog counter frozen in Standby mode Checked : Independent watchdog counter running in Standby mode
IWDGSTOP	☑	Unchecked : Independent watchdog counter frozen in Stop mode Checked : Independent watchdog counter running in Stop mode
IWDGSW	☑	Unchecked : Hardware independent watchdog Checked : Software independent watchdog
IPCCDBA	0x0	IPCC mailbox data buffer base address

Clock output

The HSE clock is output on the pin PA8 (MCO), available on connectors CN9/D6 and CN10/25. PA8 configuration is performed by the firmware. A frequency meter probe is connected to one of these connectors, and ground can be taken from connector CN11 or CN12. According to the type of frequency meter used, AC coupling instead of DC may be needed.

The next sequence is required to output the HSE on the MCO pin.

1. Turn on the HSE oscillator:
   a) Set clock control register RCC_CR[16] = 1
2. Configure PA8 pin to Clock output function:
   a) Select GPIO alternate function to MCO (= 0x0)
   b) Select GPIO speed to Very high frequency (= 0x3)
3. Select HSE as output clock with no division factor:
   a) Clock configuration register RCC_CFGR[30:24] = 0b000_0100

Load capacitance setting

The RCC_HSECR[13:8] register drives the load capacitance.

The proposed procedure uses the three push-button available on the Nucleo68 board to modify the register value:

- Pushing SW1 button increases this value by 1.
- Pushing SW2 button saves it in the nonvolatile memory.
- Pushing SW3 button decreases it by 1.
  – Initial value is set to 0 after reset, it cannot be increased above 0x3F (maximum load capacitance), and cannot be decreased below 0x0 (minimum load capacitance). After each action, measure the frequency.

The following sequence is required for each tested value:

1. Disable the HSE clock:
   – Clock control register RCC_CR[16] = 0
2. Unlock RCC_HSECR register:
   – WRITE_REG(RCC_HSECR, 0xCAFECAFE);
3. Write the six bits of load capacitance in RCC_HSECR[13:8]
4. Turn on the HSE oscillator:
   – Clock control register RCC_CR[16] = 1

Other fields of the register remain unchanged, and keep their initial value:

- HSE current control (HSEGMC = RCC_HSECR[6:4]) is set to 0x3 → current max limit 1.13 mA / V.
- HSE sense amplifier threshold (HSES = RCC_HSECR[3]) is set to 0 → HSE bias current factor 1/2

3.2.2 Software implementation

Project configurations

Two project configurations are available in the package:

1. STM32WBxx_Nucleo_Set_Calibration: calibration procedure
2. STM32WBxx_Nucleo_Test_Calibration: to test the stored value

This last configuration is given as an implementation example of the HSE clock initialization in RF applications.

The firmware is built on the STM32WB HAL drivers.

1. STM32WBxx_Nucleo_Set_Calibration: the device is programmed to:
   - Send the HSE clock on the PA8 pin.
   - Modify and set the load capacitance value when push-buttons SW1 and SW3 are actioned.
   - Store the load capacitance value together with the 48 bits of additional data (Bluetooth device address) in the OTP or the selected flash memory area when SW2 button is pushed.
2. STM32WBxx_Nucleo_Test_Calibration: configuration to test the actual HSE settings:
   - Configure the HSE clock and output it on PA8 pin.
   - Fetch the load capacitance value from the OTP/flash memory area.
   - Program RCC_HSECR register accordingly.

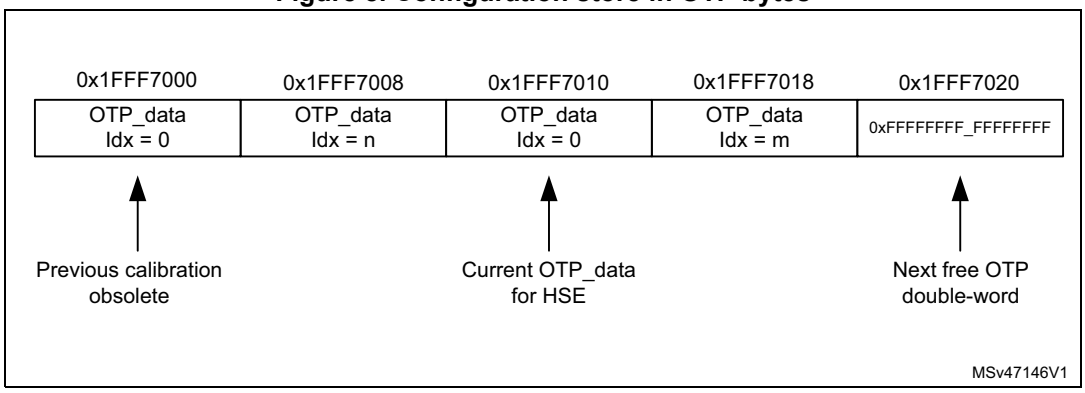
Storage in OTP (one-time-programmable) bytes

The load capacitance value is included in a 64-bit structure. Each OTP structure type is indicated by its index (one byte). The index used for the structure in this document is 0. Six bytes remain to store additional data like MAC, Bluetooth device address or crypto key.

```
typedef __packed struct
  {
    uint8_t   additional _data[6]; /* 48 bits */
    uint8_t   hse_tuning; /* Load capacitance value */
    uint8_t   index; /* structure index ==0x00*/
  } OTP_BT_t;
```

The configuration phase usually is not repeated, but there are cases when it must be overwritten. When using OTP bytes, the current configuration cannot be removed, hence the new one is placed at the next free double-word slot (see *Figure 8*).

Figure 8. Configuration store in OTP bytes



When the calibration phase has been done, the application initialization phase must retrieve the load capacitance (and other additional data) from this OTP area. The value retained is then the last one with the right index.

3.2.3 Scripts

Two batch scripts are provided to run each firmware configuration:

1. STM32WBxx_Nucleo_Set_HSE_Calibration_OTP.bat
2. STM32WBxx_Nucleo_Test_HSE_Calibration_OTP.bat

These scripts call the STM32CubeProgrammer in command-line mode, and the path to the tool must be set accordingly: SET CLI="C:\Program Files (x86)\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI.exe.

The binary or hex file of the configuration must be set also (example for Set_Calibration)

1. For Keil®:
   SET HEXFILE="STM32WBxx_Nucleo\STM32WBxx_Nucleo_Set_Calibration.hex"
2. For IAR™:
   SET HEXFILE="STM32WBxx_Nucleo_Set_Calibration\Exe\
   STM32WBxx_Nucleo_Set_Calibration.hex"
3. For STM32CubeIDE:
   SET HEXFILE="STM32WBxx_Nucleo_Set_Calibration\Debug\
   STM32WBxx_Nucleo_Set_Calibration.hex"

The 48 bits of additional data (last parameter) are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

```
%CLI% -c port=swd -w32 0x2002FFF0  0x33445566
%CLI% -c port=swd -w32 0x2002FFF4  0x00001122
```

4 Manual frequency trimming procedure example for the STM32WBA series

The firmware and scripts associated to this document are available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\P-NUCLEO-WBA52CG\RCC_HSE_Calib).

4.1 Procedure description

The procedure consists in measuring the HSE clock generated from the external crystal ine the device. This clock is output on pin PA8, and is measured by a precision frequency meter.

A step-by-step tuning of the load capacitance is performed to reach the best accuracy of the HSE clock. The load capacitance value is then stored inside a nonvolatile location of the device, either a dedicated area of the user flash memory, or in the One-Time-Programming (OTP) area (512 bytes).

Flash memory or OTP programming is done with a quad-word granularity (128 bits). The load capacitance (1 byte) value is saved along a 112-bit wide structure that contains personalization data (such as Bluetooth device address, MAC short address, product specific code, key).

This procedure can be done several times, only the latest setup is active.

Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function), and the HSE configuration register set accordingly.

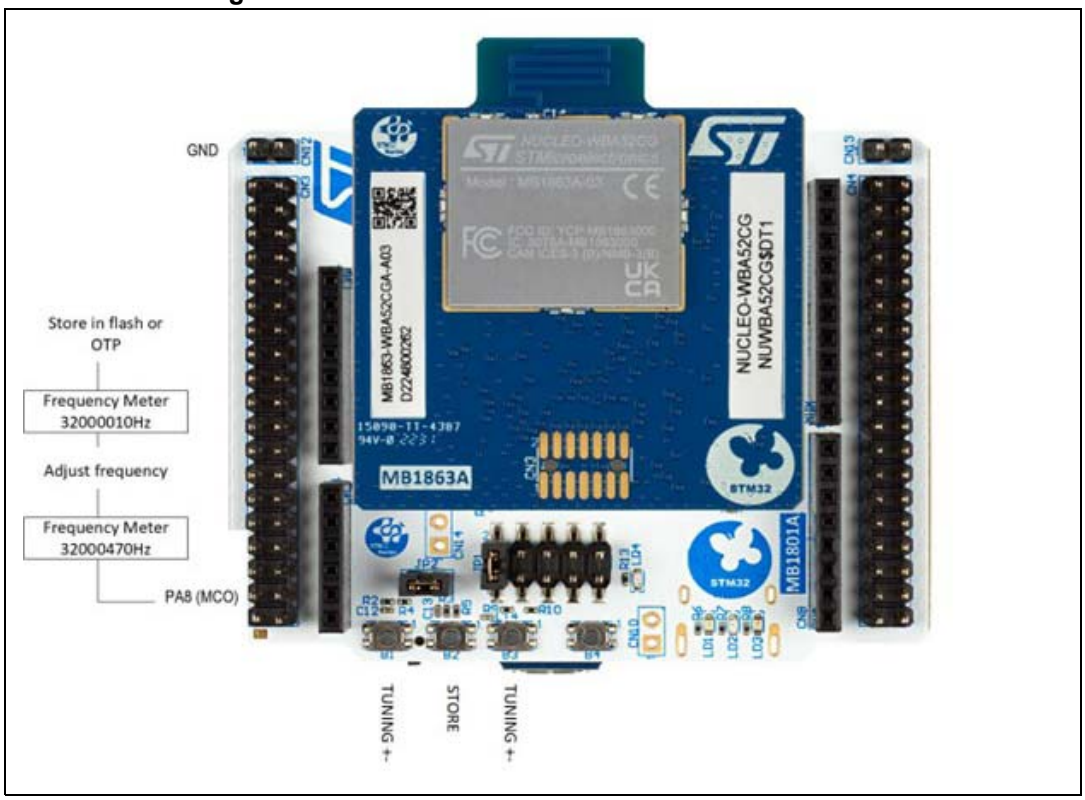
4.2 Implementation

The procedure is executed in SRAM, so it can be run on an already programmed device, without modifying the flash memory content.

4.2.1 Hardware setup

Figure 9 shows an example of hardware configuration for the manual calibration of an STM32WBA Series Nucleo-68 board.

Figure 9. Manual calibration overview - STM32WBA52



A precision frequency meter (better than 0.1 ppm) must be connected to pin PA8/MCO, and set to detect a 32 MHz 3.3 V square wave peak to peak signal.

Note: Standard oscilloscopes are not sufficiently accurate for this kind of measurement.

Clock output

HSE clock is output on the pin PA8 (MCO), which is available on pin 35 from connector CN3. A frequency meter probe should be connected to this connector and ground can be taken from connector CN12. According to the type of frequency meter used, AC coupling instead of DC may be needed.

The next sequence is required to output the HSE on the MCO pin.

1. Turn on the HSE oscillator:
   a) Set clock control register RCC_CR[16] = 1
2. Configure PA8 pin to Clock output function
   a) Select GPIO alternate function to MCO (= 0x0)
   b) Select GPIO speed to high frequency (= 0x2)
3. Select HSE as output clock with no division factor:
   a) Clock configuration register RCC_CFGR1[30:24] = 0b000_0100

Load capacitance setting

The HSETRIM[21:16] bits from RCC_ECSCR1 register drive the load capacitance.

The proposed procedure uses the three push-button available on the Nucleo68 board to modify the register value HSETRIM:

- Pushing SW1 button increases this value by 1.
- Pushing SW2 button saves it in the nonvolatile memory.
- Pushing SW3 button decreases it by 1.

Initial value is set to 0x20 after reset, it cannot be increased above 0x3F (maximum load capacitance) and cannot be decreased below 0x0 (minimum load capacitance).

4.2.2 Software implementation

Project configurations

Two project configurations are available in the package

1. RCC_HSE_Calib_Set_Calibration, for the calibration procedure
2. RCC_HSE_Calib_Test_Calibration, to test the stored value

This last configuration is given as an implementation example of the HSE clock initialization in RF applications. The firmware is built on the STM32WBA HAL drivers.

1. RCC_HSE_Calib_Set_Calibration: the device is programmed to:
   - Send the HSE clock on the PA8 pin
   - Modify and set the load capacitance value when push-buttons SW1 and SW3 are actioned
   - Store the load capacitance value together with the 112 bits of additional data when SW2 button is pushed
2. RCC_HSE_Calib_Test_Calibration: configuration to test the actual HSE settings:
   - Configure the HSE clock and output it on PA8 pin
   - Fetch the load capacitance value HSETRIM from the OTP
   - Program RCC_ECSCR1 register accordingly

Storage in OTP

Load capacitance value is included in a 128-bit structure. Each OTP structure type is indicated by its index (one byte). The index used for the structure in this document is 0, there are 14 bytes to store additional data like MAC, Bluetooth device address or crypto key.

```
typedef __packed struct
{
  uint8_t bytes[14];
  uint8_t hsetune;
  uint8_t index;
} OTP_Data_s;
```

Usually the configuration phase is not repeated, but there are cases when it must be overwritten. When using OTP bytes, the current configuration cannot be removed, the new one is placed at the next free double-word slot (see *Figure 8*).When the calibration phase is done, the application initialization phase must retrieve the load capacitance and other data from this OTP area. The retained value is the last one with the right index.

4.2.3 Scripts

Two batch scripts are provided to run each firmware configuration:

1. RCC_HSE_Calib_Set_Calibration.bat
2. RCC_HSE_Calib_Test_Calibration.bat

These scripts call the STM32CubeProgrammer in command-line mode, and the path to the tool must be set accordingly: SET CLI="C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI.exe".

To be executed in SRAM, the Reset_Handler and Main Stack Pointer must be set accordingly, using the map file generated after the compilation process. In the script example, this is done automatically by parsing the.map file generated by the IDE.

The 112 bits of additional data are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

5 Manual frequency trimming procedure example for the STM32WL series

The firmware associated to this application note is available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\Nucleo-WL55JC\RCC_HSE_Calib_SingleCore).

Note: The trimmed NUCLEO-WL55JC board needs to be configured to use the on-board crystal as HSE instead of the TCXO (which adjusts its frequency autonomously, according to the temperature). For details refer to UM2592, available on www.st.com.

5.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. This clock is output on pin PA8 and is measured by a precision frequency meter.

Note: An external reference is mandatory since no such accurate internal one exists in the device.

A step-by-step tuning of the load capacitance is performed to reach the best accuracy of the HSE clock. The IN and OUT banks load capacitance values are then stored in a nonvolatile location of the device, either a dedicated area of the user flash memory or in the OTP programming area.

Flash memory or OTP programming is done with a double-word granularity (64 bits). To save OTP bytes (1 K in the STM32WL series), the IN and OUT load capacitance values, each one on six bits, can be appended to a 64-bit wide structure with other personalization data (such as device address, product specific code, key).

This procedure can be done several times, only the last setup is active.

Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function), and the HSE configuration register set accordingly.

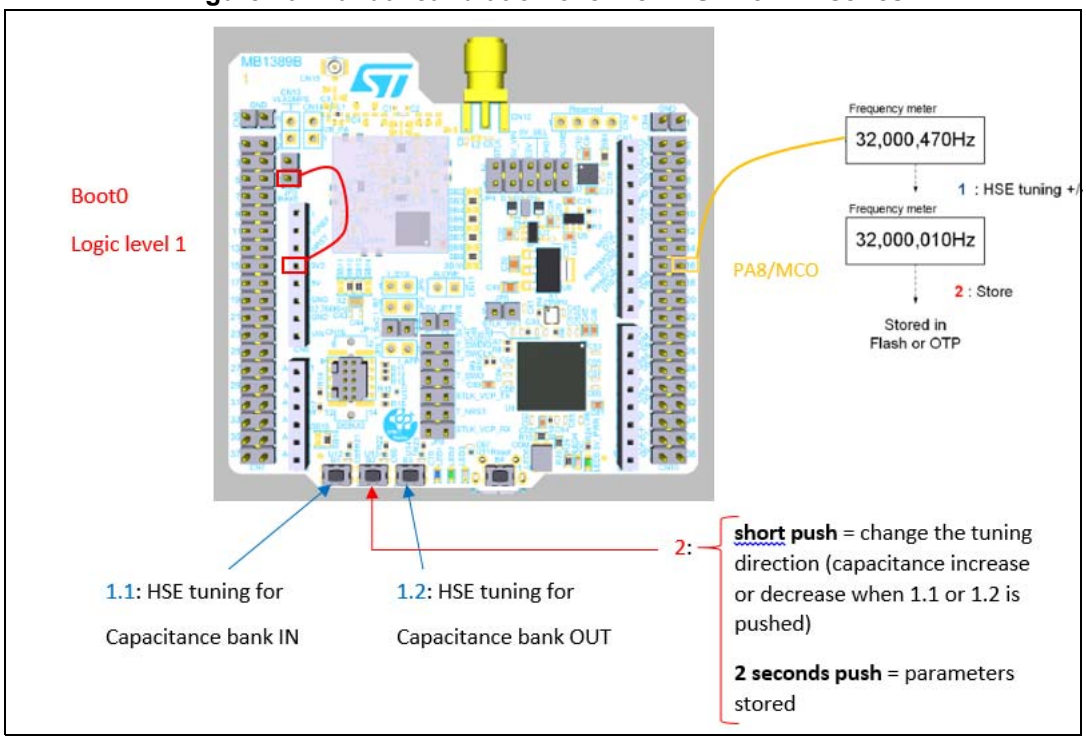
5.2 Implementation

The procedure runs in SRAM, so it can be executed on an already programmed device without modifying the flash memory content.

5.2.1 Hardware setup

Figure 10 shows the manual calibration procedure for an STM32WL series Nucleo board.

Figure 10. Manual calibration overview - STM32WL series



A precision frequency meter (better than 0.1 ppm) must be connected to pin PA8/MCO and set to detect a 32 MHz square wave 3.3 V peak to peak signal.

Note: Standard oscilloscopes are not enough accurate for this kind of measurement.

Boot from SRAM

Boot selection can be done through the BOOT0 pin and nBOOT1 bit in the user options (FLASH_OPTR).

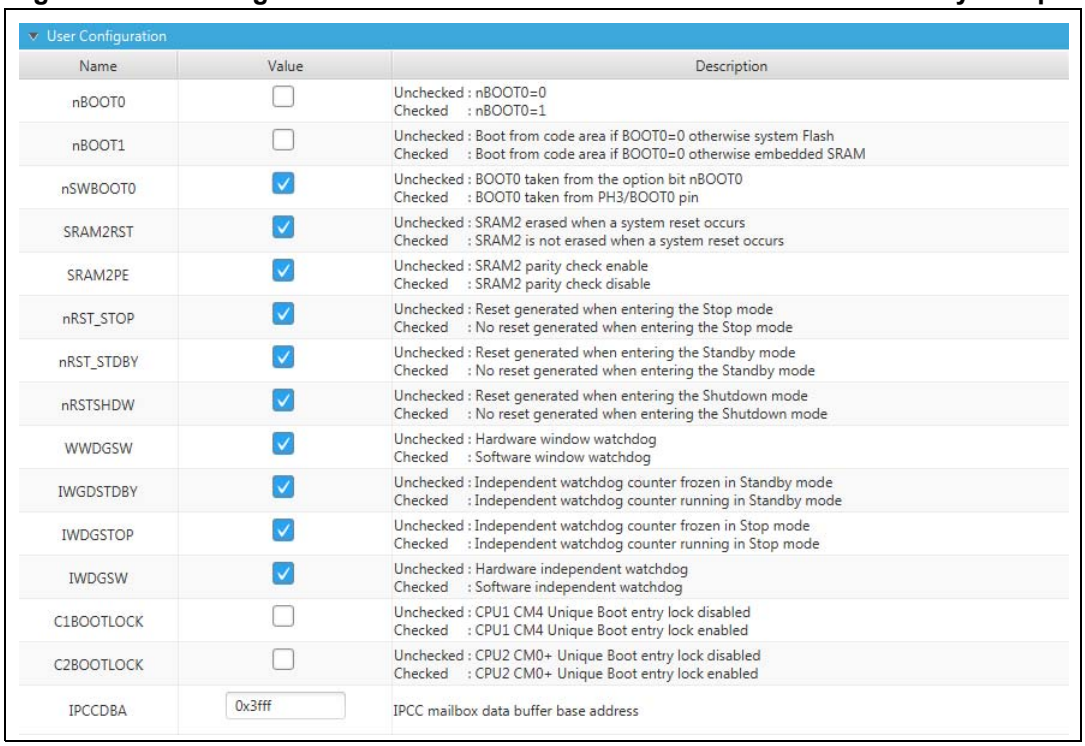
The boot from SRAM configuration is set by both BOOT0 = 1 and nBOOT1 = 0 conditions, nBOOT1 is set only by option bit FLASH_OPTR[23].

BOOT0 can be selected

- through value of pin PH3 at startup if option bit nSWBOOT0 = 1 (FLASH_OPTR[26] = 1), see the option byte panel in *Figure 11*
- through option bit value nBOOT0 if option bit nSWBOOT0 = 0 (FLASH_OPTR[26] = 0), see the option byte panel in *Figure 12*

Option bits can be selected through STM32CubeProgrammer.

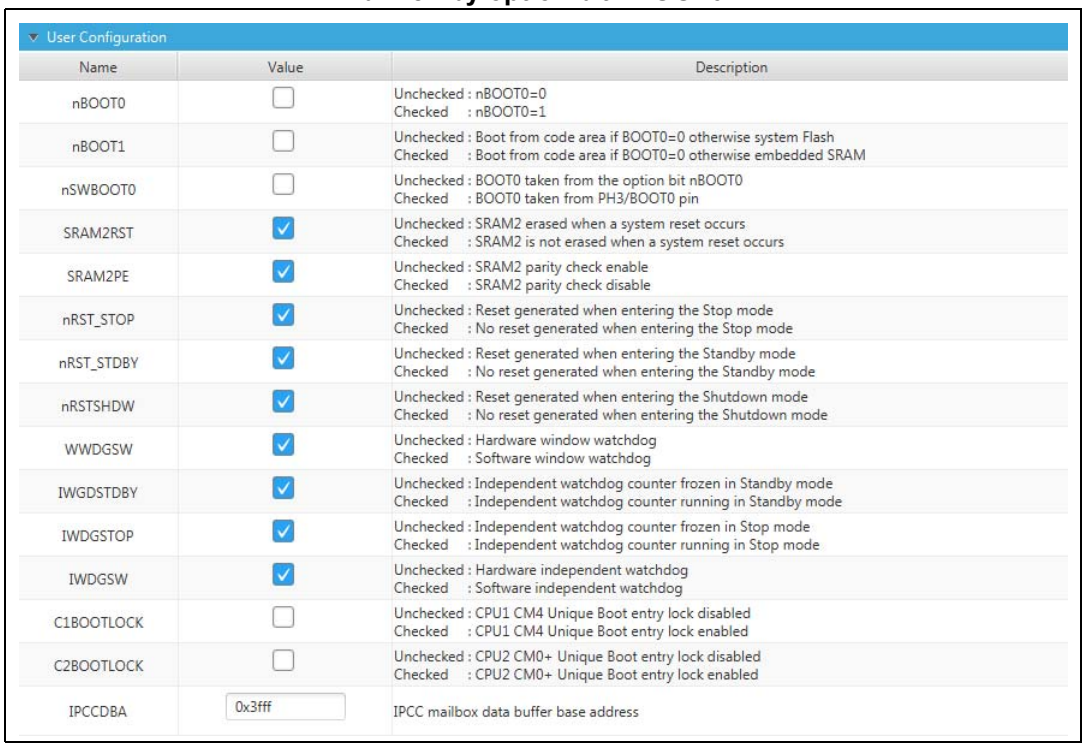
Figure 11. OB configuration to boot from SRAM with BOOT0 value driven by PH3 pin



User Configuration

Name	Value	Description
nBOOT0	☐	Unchecked : nBOOT0=0 Checked : nBOOT0=1
nBOOT1	☐	Unchecked : Boot from code area if BOOT0=0 otherwise system Flash Checked : Boot from code area if BOOT0=0 otherwise embedded SRAM
nSWBOOT0	☑	Unchecked : BOOT0 taken from the option bit nBOOT0 Checked : BOOT0 taken from PH3/BOOT0 pin
SRAM2RST	☑	Unchecked : SRAM2 erased when a system reset occurs Checked : SRAM2 is not erased when a system reset occurs
SRAM2PE	☑	Unchecked : SRAM2 parity check enable Checked : SRAM2 parity check disable
nRST_STOP	☑	Unchecked : Reset generated when entering the Stop mode Checked : No reset generated when entering the Stop mode
nRST_STDBY	☑	Unchecked : Reset generated when entering the Standby mode Checked : No reset generated when entering the Standby mode
nRSTSHDW	☑	Unchecked : Reset generated when entering the Shutdown mode Checked : No reset generated when entering the Shutdown mode
WWDGSW	☑	Unchecked : Hardware window watchdog Checked : Software window watchdog
IWGDSTDBY	☑	Unchecked : Independent watchdog counter frozen in Standby mode Checked : Independent watchdog counter running in Standby mode
IWDGSTOP	☑	Unchecked : Independent watchdog counter frozen in Stop mode Checked : Independent watchdog counter running in Stop mode
IWDGSW	☑	Unchecked : Hardware independent watchdog Checked : Software independent watchdog
C1BOOTLOCK	☐	Unchecked : CPU1 CM4 Unique Boot entry lock disabled Checked : CPU1 CM4 Unique Boot entry lock enabled
C2BOOTLOCK	☐	Unchecked : CPU2 CM0+ Unique Boot entry lock disabled Checked : CPU2 CM0+ Unique Boot entry lock enabled
IPCCDBA	0x3fff	IPCC mailbox data buffer base address

Figure 12. OB configuration to boot from SRAM with BOOT0 value driven by option bit nBOOT0



User Configuration

Name	Value	Description
nBOOT0	☐	Unchecked : nBOOT0=0 Checked : nBOOT0=1
nBOOT1	☐	Unchecked : Boot from code area if BOOT0=0 otherwise system Flash Checked : Boot from code area if BOOT0=0 otherwise embedded SRAM
nSWBOOT0	☐	Unchecked : BOOT0 taken from the option bit nBOOT0 Checked : BOOT0 taken from PH3/BOOT0 pin
SRAM2RST	☑	Unchecked : SRAM2 erased when a system reset occurs Checked : SRAM2 is not erased when a system reset occurs
SRAM2PE	☑	Unchecked : SRAM2 parity check enable Checked : SRAM2 parity check disable
nRST_STOP	☑	Unchecked : Reset generated when entering the Stop mode Checked : No reset generated when entering the Stop mode
nRST_STDBY	☑	Unchecked : Reset generated when entering the Standby mode Checked : No reset generated when entering the Standby mode
nRSTSHDW	☑	Unchecked : Reset generated when entering the Shutdown mode Checked : No reset generated when entering the Shutdown mode
WWDGSW	☑	Unchecked : Hardware window watchdog Checked : Software window watchdog
IWGDSTDBY	☑	Unchecked : Independent watchdog counter frozen in Standby mode Checked : Independent watchdog counter running in Standby mode
IWDGSTOP	☑	Unchecked : Independent watchdog counter frozen in Stop mode Checked : Independent watchdog counter running in Stop mode
IWDGSW	☑	Unchecked : Hardware independent watchdog Checked : Software independent watchdog
C1BOOTLOCK	☐	Unchecked : CPU1 CM4 Unique Boot entry lock disabled Checked : CPU1 CM4 Unique Boot entry lock enabled
C2BOOTLOCK	☐	Unchecked : CPU2 CM0+ Unique Boot entry lock disabled Checked : CPU2 CM0+ Unique Boot entry lock enabled
IPCCDBA	0x3fff	IPCC mailbox data buffer base address

Clock output

HSE clock is output on pin PA8 (MCO), available on connectors CN10/16. PA8 configuration is performed by the firmware. Frequency meter probe is connected to this connector and ground can be taken from connector CN3 or CN4. According to the type of frequency meter used, it can be needed to use AC coupling instead of DC.

The next sequence is required to output the HSE on the MCO pin.

1. Turn on the HSE oscillator:
   a) Set clock control register RCC_CR[16]=1
2. Configure PA8 pin to clock output function:
   a) Select GPIO alternate function to MCO (= 0x0)
   b) Select GPIO speed to Very high frequency (= 0x3)
3. Select HSE as output clock with no division factor:
   a) Clock configuration register RCC_CFGR[30:24] = 0b000_0100

Load capacitance setting

The SUBGHZ_HSEINTRIMR[5:0] bits set the capacitance for bank IN.

The SUBGHZ_HSEOUTTRIMR[5:0] bits set the capacitance for bank OUT.

The proposed procedure uses the three push-buttons available on the Nucleo board to modify the registers value:

- After startup, each push of SW1 or SW3 respectively increases the value of SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR by 1.
- One short push on SW2 changes the variations direction. For example if we push one time SW2 after startup, then each SW1 / SW3 push respectively decreases the value of SUBGHZ_HSEINTRIMR / SUBGHZ_HSEOUTTRIMR by 1. Another SW2 push sets back an increasing variations direction, another one sets back a decreasing variations direction, and so on.
- Pushing SW2 for two seconds (the three Nucleo LEDs blink at the same time when the push has been taken into account) stores the parameters in memory (OTP or flash memory according to constants defined by the user).

For these two registers, the initial value is set to 0x12 (20.3 pF) after reset. It cannot be increased above 0x2F (maximum load capacitance 33.4 pF), and cannot be decreased below 0x0 (minimum load capacitance 11.3 pF). After each action, the frequency must be measured.

The following sequence is required for each tested value:

1. Enter the sub-GHz radio in standby with HSE32 mode:
   – sends the command code 0x80 to the sub-GHz radio SPI interface with no parameters (set the parameter to 0 by default).
2. Write the six bits of load capacitance in SUBGHZ_HSEINTRIMR[5:0] or SUBGHZ_HSEOUTTRIMR[5:0]:
   – sends the command code 0x0D (write register command) to the sub-GHz radio SPI interface followed by the register address (0x0911 for SUBGHZ_HSEINTRIMR and 0x0912 for SUBGHZ_HSEOUTTRIMR) and the value they have to take.
3. Enter the sub-GHz radio in FS (frequency synthesis) mode (to exit standby mode):
   – sends the command code 0xC1 to the sub-GHz radio SPI interface with no parameters (set the parameter to 0 by default).

5.2.2 Software implementation

Project configurations

Two project configurations are available in the package, one for the calibration procedure (determination called RCC_HSE_Calib_SingleCore) and another to test the stored value (called RCC_HSE_Calib_SingleCore_Test). The latter configuration is given as an implementation example of the HSE clock initialization in RF applications.

The firmware is built on the STM32WL HAL drivers.

1. RCC_HSE_Calib_SingleCore: the device is programmed to:
   a) Send the HSE clock on the PA8 pin.
   b) Modify and set the IN and OUT load capacitance values when push-buttons SW1 and SW3 are actioned (short push on SW2 changes the variations direction).
   c) Store the load capacitance value together with the 40 bits of additional data (Bluetooth device address) in the OTP or the selected flash memory area when SW2 is pushed for two seconds.
2. RCC_HSE_Calib_SingleCore_Test, a configuration to test the saved HSE settings:
   a) Configure the HSE clock and output it on PA8 pin.
   b) Fetch the IN and OUT load capacitance values from the OTP/flash memory area.
   c) Program SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR registers accordingly.

Storage in OTP bytes

Load capacitance values are included in a 64-bit structure. Each OTP structure type is indicated by its index (one byte). The index used for the structure in this application note is 0. Five bytes remain to store additional data like device address or crypto key.

```
typedef __packed struct
{
uint8_t additional _data[5]; /* 48 bits */
uint8_t hse_tuning_in; /* IN bank load capacitance value */
uint8_t hse_tuning_out; /* OUT bank load capacitance value */
uint8_t index; /* structure index ==0x00*/
} OTP_BT_t
```

Even if the configuration phase is not supposed to be repeated, there may be some case where it should be overwritten. When using OTP bytes, the current configuration cannot be removed, so the new one is placed at the next free double-word slot (see *Figure 8*).

When the calibration phase has been done, the application initialization phase must retrieve the IN and OUT load capacitances (and other additional data) from this OTP area. The values retained are the last ones with the right index.

5.2.3 Scripts

Two batch scripts are provided to run each firmware configuration:

- RCC_HSE_Calib_SingleCore_OTP.bat
- RCC_HSE_Calib_SingleCore_Test_OTP.bat

These scripts call the STM32CubeProgrammer command-line interface, and the path to the tool must be set accordingly:

```
SET CLI="C:\Program Files (x86)
\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI
.exe"
```

The binary or hex file of the configuration must be set also

For IAR™:

```
SET BINFILE="RCC_HSE_Calib_SingleCore\Exe\RCC_HSE_Calib_SingleCore.bin"
```

For Keil®:

```
SET HEXFILE="RCC_HSE_Calib_SingleCore\Exe\RCC_HSE_Calib_SingleCore.hex"
```

The 40 bits of additional data (last parameter) are transmitted to the device through these scripts. They are stored inside the SRAM and read by the firmware.

```
STM32_Programmer_CLI.exe -c port=swd -w32 0x20006FF0  0x22334455
STM32_Programmer_CLI.exe -c port=swd -w32 0x20006FF4  0x00000011
```

6 Automatic frequency trimming procedure example for the STM32WB series

The firmware and scripts associated to this document are available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\P-NUCLEO-WB55.Nucleo\RCC_HSE_AutoCalib).

6.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. The measure is done internally to the STM32 system and is based on an accurate external 16 MHz clock reference provided by the user on PA9.

Note: An external reference is mandatory since no such accurate one is integrated in the device.

An automatic tuning of the load capacitance is performed by a specific STM32 application to reach the best accuracy of the HSE clock. The load capacitance value found is then stored in a nonvolatile location of the device, in a dedicated area of the user flash memory or of the OTP.

Flash memory or OTP programming is done with a double-word granularity (64 bits). To save OTP bytes (1 K in the STM32WB series), the load capacitance value on six bits can be appended to a 64-bit wide structure with other personalization data (such as the Bluetooth device address, the MAC short address, the product specific code, the key).

This procedure can be done several times, only the latest setup is active.

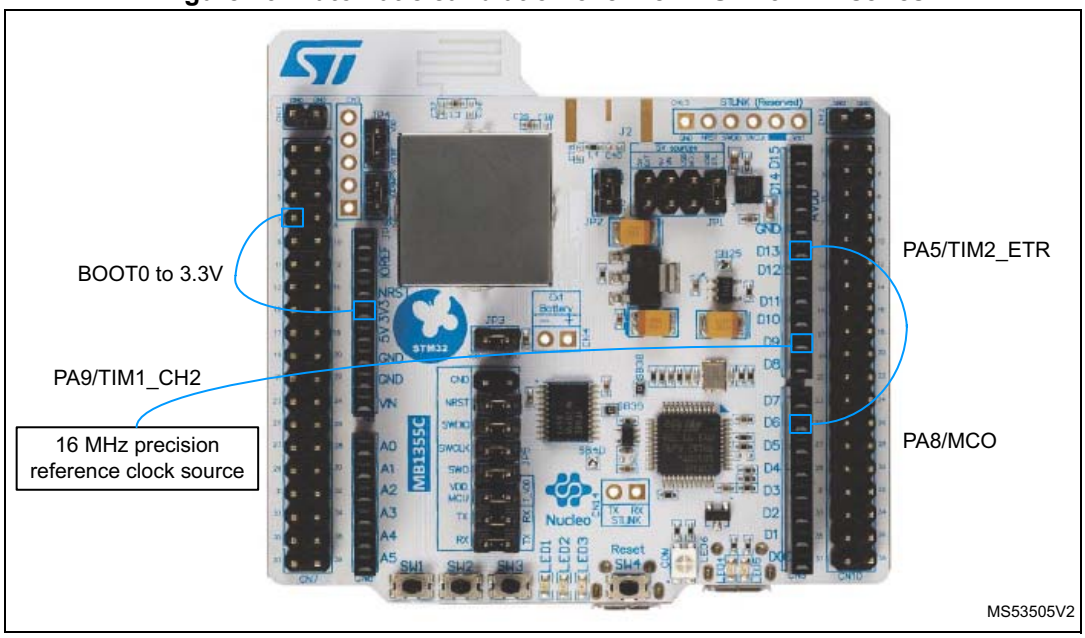
Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function) and the HSE configuration register set accordingly.

6.2 Implementation

6.2.1 Hardware setup

Figure 13 shows the automatic calibration procedure for an STM32WB series Nucleo-68 board.

Figure 13. Automatic calibration overview - STM32WB series



A precision clock generator (better than 0.1 ppm) must be connected to pin PA9/TIM1_CH2, and set to generate a 16 MHz, 3.3 V_{PP} square wave. To see how to configure the board so as to boot from SRAM refer to *Section 3.2.1: Hardware setup*.

Note: A standard signal generator does not give enough accuracy for this application.

Hardware connections principle

Two timers (TIM1 and TIM2) are used by the MCU to exploit the external reference clock and thus to measure and tune the HSE frequency.

TIM1 is clocked by the 16 MHz precision external reference clock, itvgenerates a PWM signal low during an OFF time (fixed to 16 ms) and high during a REF time (100 or 1000 ms, depending on the stage of the calibration procedure).

This PWM signal is used to enable TIM2 counter when high.

TIM2 is clocked by HSE/2 output on MCO pin.

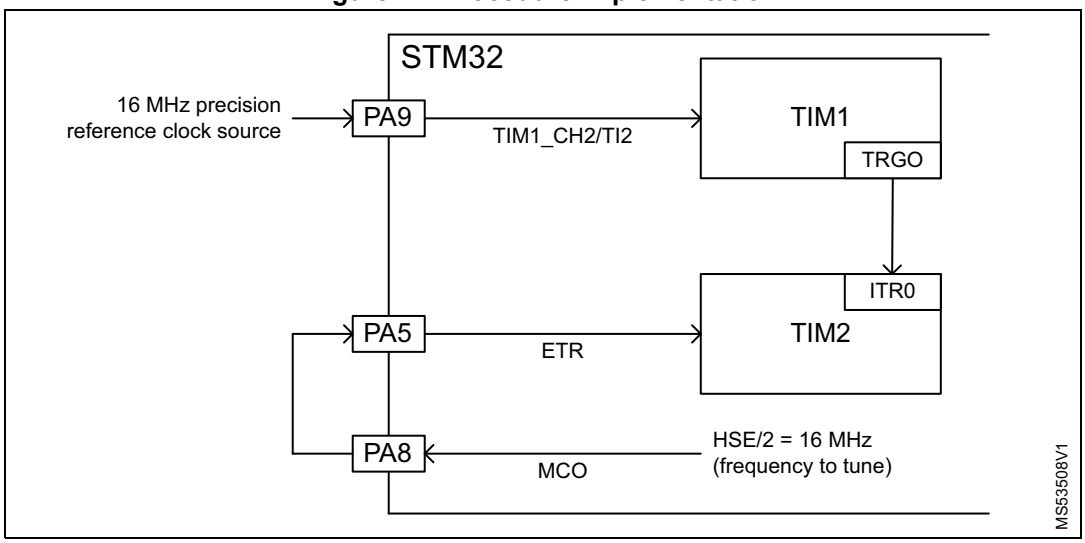
Then, after one PWM signal period, TIM2 counter corresponds to the number of HSE/2 periods elapsed during REF time.

This value can be compared to the expected value to adjust the load capacitance conditioning HSE frequency.

Note: OFF time is needed to temporize and wait for HSE stabilization when several HSE measurements are done in sequence.

Figure 14 illustrates the connections between the elements involved in the procedure.

Figure 14. Procedure implementation



TIM1 configuration

TIM1 is configured to be clocked by a 16 MHz external signal provided on PA9. PA9 is associated to the TIM1 CH2, which provides the TI2 (trigger input 2) input. TI2 must be chosen as TIM1 clock source.

- SMS bits of TIM1_SMCR register = b0111 (external clock mode 1)
- TS bits of TIM1_SMCR register = b00110 (trigger input = TI2, filter, and polarity for the input must be set).

TIM1 must generate a PWM signal low during OFF time, and high during REF time.

For this purpose, TIM1 CH3 is configured in PWM mode 2 (low, then high cycles) with a prescaler of 16000 to get a 1 ms granularity (APB2 frequency is configured to be 16 MHz and 16 MHz / 16000 = 1 kHz). The period is set to REF time + OFF time - 1, and the pulse duration to OFF time to get the desired signal (REF and OFF times in milliseconds).

- TIM1_ARR register = 0x3E7F (prescaler = 16000)
- TIM1_PSC register = REF time + OFF time - 1 (period)
- OC3M bits of TIM1_CCMR2 register = b111 (PWM mode 2)
- TIM1_CCR3 register = OFF time (pulse for channel 3)

Finally, TIM1 must be set in master mode OC3REF so that its TRGO (trigger output) relays the PWM signal generated by its CHANNEL 3.

- MMS bits of TIM1_CR2 register = b110 (OC3REF master mode)

TIM2 configuration

TIM2 must be clocked by HSE/2.

For this purpose HSE/2 is output on MCO pin PA8:

- RCC_CFGR[30:24] = b001_0100 (HSE as MCO output with a division factor of 2).

PA8 is linked externally to PA5, corresponding to TIM2 ETR (external trigger) input.

Then, external clock mode 2 (counter clocked by any active edge on the ETRF signal) must be chosen as TIM2 clock source.

- ECE bits in TIM2_SMCR = b1

TIM2 CH2 is configured as a simple time base to count the HSE/2 periods. The time base period is set to its maximum value 0xFFFFFFFF, to be sure that HSE/2 measurement does not overflow.

- TIM2_ARR register = 0 (prescaler = 0)
- TIM2_PSC register = 0xFFFFFFFF (period)

Finally, TIM2 must be set in GATED slave mode with ITR0 as trigger input. Since TIM2 ITR0 and TIM1 TRGO are connected, the GATED mode enables the TIM2 CH2 counter only when the TIM1 CH3 PWM signal displayed on TRGO is high.

- SMS bits in TIM2_SMCR register = b0101 (GATED mode)
- TS bits in TIM2_SMCR register = b00000 (ITR0 as trigger input)

6.2.2 Software implementation

Firmware tuning algorithm principle

The timer links presented above allow to count the number of HSE/2 periods elapsed during REF time with the external reference clock precision.

The HSE_Measurement function of the firmware provided with this document makes it possible to get this number of periods by launching one TIM1 CH3 PWM cycle, and returning the value of the TIM2 CH2 counter.

The function HSE_Tuning exploits HSE_Measurement to tune HSE, using dichotomy logic to find the best load capacitance value for HSE, to be the closest to 32 MHz.

The tuning parameter that sets the load capacitance can take values from 0 to 63. The function starts by setting it to 32 (RCC_HSECR[13:8] = HSETUNE = 32), then it makes an HSE/2 measurement.

If the measured frequency (calculated thanks to the number of periods returned by HSE_Measurement) is above 16 MHz, the function increases HSETUNE to lower the frequency. If it is below, it decreases it, to increase the frequency.

The variations of HSETUNE are done with a step starting with a value of 16 when HSETUNE is 32. After each measurement, this step is divided by two to calculate the new HSETUNE value to test.

So far, the measurements duration is 100 ms. When the step value reaches a value of 1 and that the load capacitance approaches the best value, the duration is set to 1000 ms to be even more precise.

When the step reaches 1, if the measured HSE/2 frequency is below 16 MHz, the function decrements HSETUNE and makes new measurements and adjustments until the measured frequency exceeds 16 MHz. Then the function takes the HSETUNE value between the one above 16 MHz and the last one below with the best HSE precision.

The reasoning is the same if the last 100 ms measurement (the last one before the step has reached 1) is above 16 MHz.

Finally, the HSETUNE value found is saved in a nonvolatile memory.

Other fields of the RCC_HSECR register remain unchanged, and keep their initial value:

- HSE current control (HSEGMC = RCC_HSECR[6:4]) is set to 0x3 → current max limit 1.13 mA/V
- HSE sense amplifier threshold (HSES = RCC_HSECR[3]) is set to 0 → HSE bias current factor 1/2

Firmware configurations

The constant STORE_ADDRESS in hse_trimming.h allows the user to choose to save HSETUNE in OTP (STORE_ADDRESS = 0x1FFF7000U) or flash memory (STORE_ADDRESS = 0x080A0000 for example).

Two project configurations are available in the package, one (RCC_HSE_AutoCalib) for the calibration procedure, another one (RCC_HSE_AutoCalib_Test) to test the stored value. This last configuration is given as an implementation example of the HSE clock initialization in RF applications.

Note: Under STM32CubeIDE, the two projects are called, respectively, STM32WBxx_Nucleo_Set_AutoCalibration, STM32WBxx_Nucleo_Test_AutoCalibration.

The firmware is built on the STM32WB HAL drivers.

- RCC_HSE_AutoCalib: the device is programmed to:
  - send the HSE clock on the PA8 pin, configure the timers and exploit the external clock source
  - modify and set the load capacitance value following the trimming algorithm presented before
  - store the load capacitance value together with the 48 bits of additional data (Bluetooth device address) in the OTP or in the flash memory.
- RCC_HSE_AutoCalib_Test: the device is programmed to test the actual HSE setting:
  - configure the HSE clock and output it on PA8 pin
  - fetch the load capacitance value from the OTP/flash memory area
  - program RCC_HSECR register accordingly.

Storage in OTP bytes

The way to store tuning parameters in OTP memory is the same used for the manual trimming procedure for STM32WB series (see *Section 3.2.2: Software implementation*).

6.2.3 Scripts

Two batch scripts are provided to run each FW configuration:

1. STM32WBxx_Nucleo_Set_HSE_AutoCalibration_OTP.bat
2. STM32WBxx_Nucleo_Test_HSE_AutoCalibration_OTP.bat

These scripts call the STM32CubeProgrammer in command-line mode, and the path to the tool must be set accordingly: SET CLI="C:\Program Files (x86)\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI.exe"

The binary or hex file of the configuration must be set also (example for the Set Calibration application)

- for IAR™: SET HEXFILE="RCC_HSE_AutoCalib \Exe\RCC_HSE_AutoCalib.hex"
- for Keil®: SET HEXFILE="RCC_HSE_AutoCalib\ RCC_HSE_AutoCalib.hex"
- For STM32CubeIDE:
  SET HEXFILE="STM32WBxx_Nucleo_Set_AutoCalibration\Debug
  \STM32WBxx_Nucleo_Set_AutoCalibration.hex"

The 48 bits of additional data (last parameter) are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

```
%CLI% -c port=swd -w32 0x2002FFF0  0x33445566
%CLI% -c port=swd -w32 0x2002FFF4  0x00001122
```

7 Legacy automatic frequency trimming procedure example for the STM32WBA series

The firmware and scripts associated to this document are available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\ NUCLEO-WBA52CG\RCC_HSE_AutoCalibLegacy).

7.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. The measure is done internally to the STM32 system, and is based on an accurate external 16 MHz clock reference provided by the user on PA15.

Note: An external reference is mandatory, as the device does not integrate an accurate one.

An automatic tuning of the load capacitance is performed by a specific STM32 application to reach the best accuracy of the HSE clock. The found value is stored in a dedicated area of the user flash memory or of the OTP.

Flash memory or OTP programming is done with a quad-word granularity (128 bits). The load capacitance value (1 byte) is saved along a 112-bit wide structure that contains personalization data (such as Bluetooth® device address, MAC short address, product specific code, key).

This procedure can be done several times, only the latest setup is active.

Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function), and the HSE configuration register set accordingly.

7.2 Implementation

7.2.1 Hardware setup

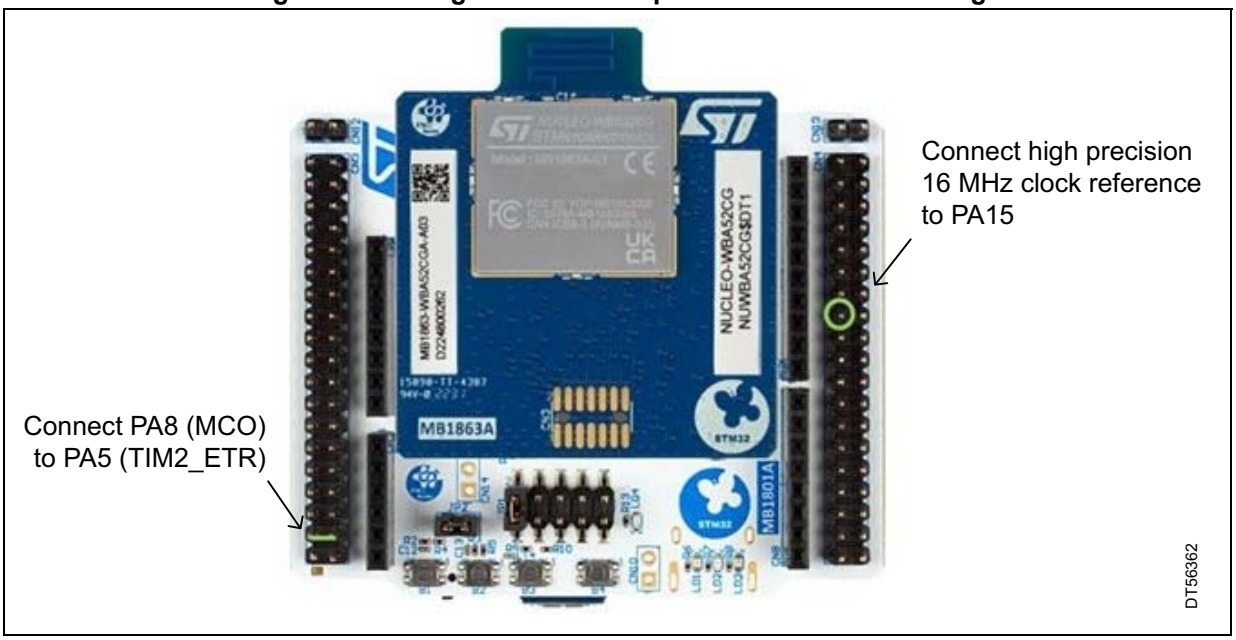
Figure 15 shows the automatic calibration procedure for a Nucleo-68 board.

A precision clock generator (better than 0.1 ppm) must be connected to pin PA15/TIM1_ETR, and set to generate a 16 MHz, 3.3 V_{PP} square wave.

Note: A standard signal generator does not give enough accuracy for this application.

Two timers (TIM1 and TIM2) are used by the MCU to exploit the external reference clock and to measure and tune the HSE frequency.

Figure 15. Configuration for simplified automatic trimming

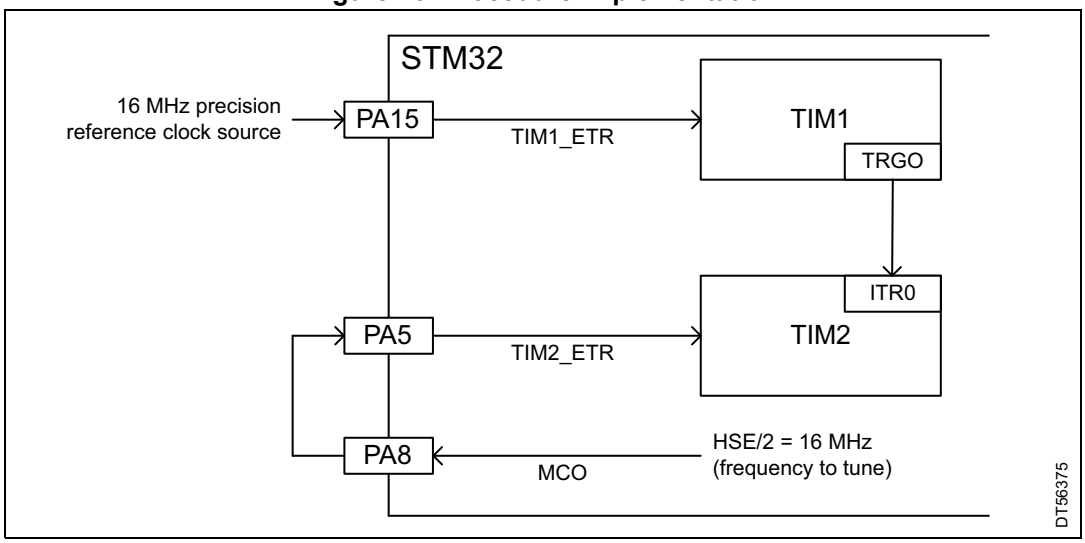


Hardware connections principle

Figure 16 illustrates the connections between the elements involved in the procedure.

Two timers (TIM1 and TIM2) are used by the MCU to exploit the external reference clock and thus to measure and tune the HSE frequency.

Figure 16. Procedure implementation



TIMER1 is clocked by the 16 MHz precision external reference clock. It generates a PWM signal low during an OFF time (fixed to 16 ms) and high during an HSETRIM_TICKS time (by default 1000 ms).

This PWM signal is used to enable TIM2 counter when high.

TIM2 is clocked by HSE/2 output on MCO pin.

Then, after one PWM signal period, TIM2 counter corresponds to the number of HSE periods divided by two elapsed during HSETRIM_TICKS.

This value can be compared with the expected value to adjust the load capacitance conditioning HSE frequency.

Note: OFF time is needed to temporize and wait for HSE stabilization when several HSE measurements are done in sequence.

TIM1 configuration

TIM1 is configured to be clocked by a 16 MHz external signal provided on PA15. PA15 is associated to the TIM1_ETR.

- ECE bit of TIM1_SMCR register = 1 (external clock mode 2)

TIM1 must generate a PWM signal low during OFF, and high during HSETRIM_TICKS.

For this purpose, TIM1 CH3 is configured in PWM mode 2 (low, then high cycles) with a prescaler of 16000, to get a 1 ms granularity (APB2 frequency is configured to 16 MHz).

The period is set to HSETRIM_TICKS + OFF - 1 (in ms), and the pulse duration to OFF, to get the desired signal.

- TIM1_ARR register = 0x3E7F (prescaler = 16000)
- TIM1_PSC register = HSETRIM_TICKS + OFF - 1 (period)
- OC3M bits of TIM1_CCMR2 register = b111 (PWM mode 2)
- TIM1_CCR3 register = OFF (pulse for channel 3)

Finally, TIM1 must be set in master mode OC3REF, so that its TRGO (trigger output) relays the PWM signal generated by its CHANNEL 3.

- MMS bits of TIM1_CR2 register = b110 (OC3REF master mode)

TIM2 configuration

TIM2 must be clocked by HSE/2.

For this purpose, HSE/2 is output on MCO pin PA8:

- RCC_CFGR[30:24] = b001_0100 (HSE as MCO output with a division factor of 2). PA8 is linked externally to PA5, corresponding to TIM2 ETR (external trigger) input.

Choose external clock mode 2 (counter clocked by any active edge on the ETRF signal) as TIM2 clock source.

- ECE bit in TIM2_SMCR = 1

TIM2 CH2 is configured as a simple time base to count the HSE/2 periods. The time base period is set to its maximum value 0xFFFFFFFF, to be sure that HSE/2 measurement does not overflow.

- TIM2_ARR register = 0 (prescaler = 0)
- TIM2_PSC register = 0xFFFFFFFF (period)

Finally, TIM2 must be set in GATED slave mode with ITR0 as trigger input. Since TIM2_ITR0 and TIM1_TRGO are connected, the GATED mode enables the TIM2 CH2 counter only when the TIM1 CH3 PWM signal displayed on TRGO is high.

- SMS bits in TIM2_SMCR register = b0101 (GATED mode)
- TS bits in TIM2_SMCR register = b00000 (ITR0 as trigger input)

7.2.2 Software implementation

Firmware tuning algorithm principle

The timer links described above make it possible to count the number of HSE/2 periods elapsed during REF time with the external reference clock precision.

The _hse_measure_frequency function of the firmware allows the user to get this number by launching one TIM1 CH3 PWM cycle, and returning the value of the TIM2 counter.

The HSETRIM_Process function is used to tune HSE oscillator, using dichotomy logic to find the best load capacitance value HSETRIM, to be the closest possible to 32 MHz.

The tuning parameter that sets the load capacitance can take values from 0 to 63. The function starts by setting it to 32 (RCC_ECSCR1[13:8] = HSETRIM = 32), then, it makes an HSE/2 measurement. It is possible to adjust the measurement duration with the macro HSETRIM_TICKS, set by default at 1000 ms. At most, the binary search algorithm finds the best HSE load capacity in $\log_2 64$ * HSETRIM_TICKS, so, by default, in 6 seconds. The default HSETRIM_TICKS is conservative, may need to be adapted.

Finally, the load capacitance HSETRIM value found is saved in the OTP or in the flash memory.

Firmware configurations

Two project configurations are available in the package, one (RCC_HSE_AutoCalib_Set_Calibration) for the calibration procedure, another one (RCC_HSE_AutoCalib_Test_Calibration) to test the stored value. This last configuration is given as an implementation example of the HSE clock initialization in RF applications.

The firmware is built on the STM32WBA HAL & LL drivers.

- RCC_HSE_AutoCalib_Set_Calibration: the device is programmed to:
  - send the HSE clock on the PA8 pin, configure the timers and exploit the external clock source
  - modify and set the load capacitance HSETRIM value following the trimming algorithm presented before
  - store the load capacitance value together with the 112 bits of additional data (Bluetooth device address) in the OTP memory.
- RCC_HSE_AutoCalib_Test_Calibration the device is programmed to test the actual HSE setting:
  - configure the HSE clock and output it on PA8 pin
  - fetch the load capacitance value from the OTP memory area and program HSETRIM from RCC_ECSCR1 register accordingly.

Storage in OTP bytes

The way to store tuning parameters in OTP memory is the same used for the manual trimming procedure for STM32WBA Series (see *Section 3.2.2*).

7.2.3 Scripts

Two batch scripts are provided to run each firmware configuration:

1. RCC_HSE_AutoCalib_Set_Calibration.bat
2. RCC_HSE_AutoCalib_Test_Calibration.bat

These scripts call the STM32CubeProgrammer in command-line mode, and the path to the tool must be set accordingly: SET CLI="C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI.exe”.

To be executed in SRAM, the Reset_Handler and Main Stack Pointer must be set accordingly, using the map file generated after the compilation process. In the script example, this is done automatically by parsing the.map file generated by the IDE.

The 112 bits of additional data are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

8 Simplified automatic frequency trimming procedure example for the STM32WBA Series

The firmware and scripts associated to this document are available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\ NUCLEO-WBA52CG\RCC_HSE_AutoCalib).

8.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. The measure is done internally to the STM32 system and is based on an accurate external 32 MHz clock reference provided by the user on PA15.

Note: An external reference is mandatory, as the device does not integrate an accurate one.

An automatic tuning of the load capacitance is performed by a specific STM32 application to reach the best accuracy of the HSE clock. The found value is stored in a dedicated area of the user flash memory or of the OTP.

Flash memory or OTP programming is done with a quad-word granularity (128 bits). The load capacitance (1 byte) value is saved along a 112-bit wide structure that contains personalization data (such as Bluetooth device address, MAC short address, product specific code, key).

This procedure can be done several times, only the latest setup is active.

Once the procedure is completed, the active load capacitance value can be retrieved at startup (in the clock configuration function), and the HSE configuration register set accordingly.

8.2 Implementation

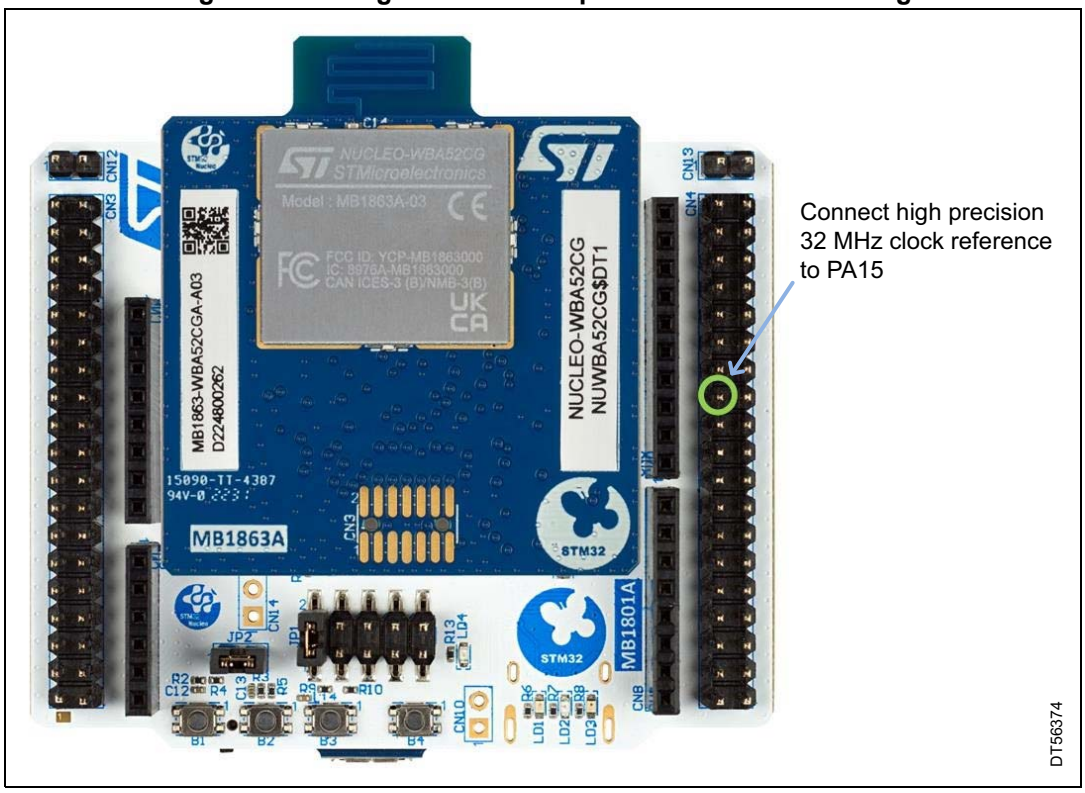
8.2.1 Hardware setup

Figure 17 shows the automatic calibration procedure for a STM32WBA Series Nucleo-68 board.

A precision clock generator (better than 0.1 ppm) must be connected to pin PA15/TIM1_ETR, and set to generate a 32 MHz, 3.3 V_{PP} square wave.

Note: A standard signal generator does not provide enough accuracy for this application.

Figure 17. Configuration for simplified automatic trimming



Hardware connections principle

Two timers (TIM1 and TIM2) are used by the MCU to exploit the external reference clock, and thus to measure and tune the HSE frequency.

TIM1 is clocked by the 32 MHz precision external reference clock divided by 4 (8 MHz). It generates a PWM signal low during OFF time (fixed to 16 ms), and high during HSETRIM_TICKS time (by default 1000 ms).

This PWM signal is used to enable TIM2 counter when high.

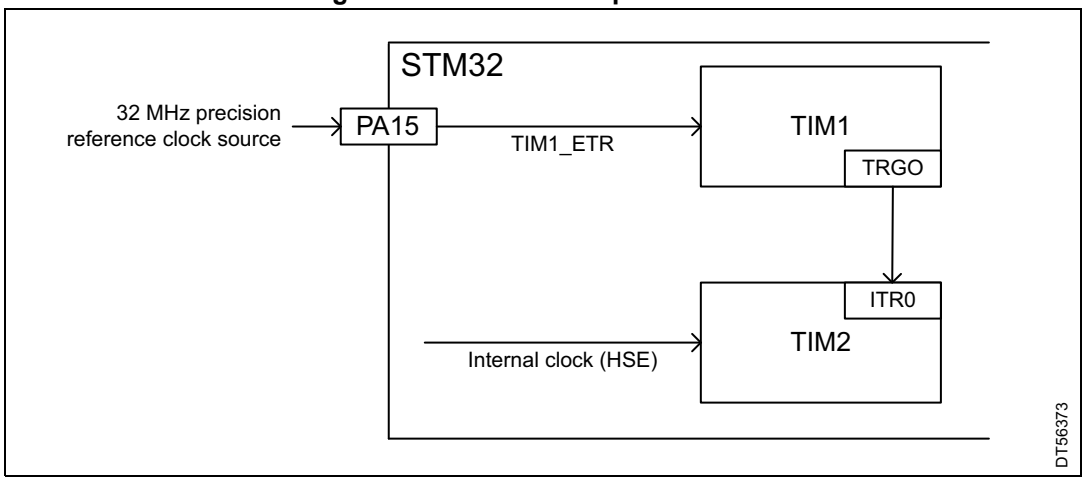
TIM2 is clocked by the internal system clock (HSE).

After one PWM signal period, TIM2 counter corresponds to the number of HSE periods elapsed during HSETRIM_TICKS. This value can be compared with the expected value, to adjust the load capacitance conditioning HSE frequency.

Note: OFF time is needed to temporize and wait for HSE stabilization when several HSE measurements are done in sequence.

Figure 18 illustrates the connections between the elements involved in the procedure.

Figure 18. Procedure implementation



TIM1 configuration

TIM1 is configured to be clocked by a 32 MHz external signal provided on PA15. PA15 is associated to the TIM1_ETR, configured to divide by 4 the 32 MHz external clock.

- ECE bit of TIM1_SMCR register = 1 (external clock mode 2)
- ETPS bits of TIM1_SMCR register = b10 (divided by 4)

TIM1 must generate a PWM signal low during OFF time, and high during REF time.

For this purpose, TIM1 CH3 is configured in PWM mode 2 (low, then high cycles) with a prescaler of 8000 to get a 1 ms granularity (APB2 frequency is configured to be 8 MHz and 8 MHz / 8000 = 1000 Hz). The period is set to HSETRIM_TICKS + OFF - 1 (in ms), and the pulse duration to OFF, to get the desired signal.

- TIM1_ARR register = 0x1F3F (prescaler = 8000 - 1)
- TIM1_PSC register = HSETRIM_TICKS time + OFF time - 1 (period)
- OC3M bits of TIM1_CCMR2 register = b111 (PWM mode 2)
- TIM1_CCR3 register = OFF time (pulse for channel 3)

Finally, set TIM1 in master mode OC3REF, so that its TRGO (trigger output) relays the PWM signal generated by its CHANNEL 3.

- MMS bits of TIM1_CR2 register = b110 (OC3REF master mode)

TIM2 configuration

TIM2 is clocked by the internal clock provided from RCC (HSE).

TIM2 CH2 is configured as a simple time base to count the HSE periods. The time base period is set to its maximum value 0xFFFFFFFF, to be sure that HSE measurement does not overflow.

- TIM2_ARR register = 0 (prescaler = 0)
- TIM2_PSC register = 0xFFFFFFFF (period)

TIM2 must be set in GATED slave mode with ITR0 as trigger input. Since TIM2 ITR0 and TIM1 TRGO are connected, the GATED mode enables the TIM2 CH2 counter only when the TIM1 CH3 PWM signal displayed on TRGO is high.

- SMS bits in TIM2_SMCR register = b0101 (GATED mode)
- TS bits in TIM2_SMCR register = b00000 (ITR0 as trigger input)

8.2.2 Software implementation

Firmware tuning algorithm principle

The timer links described above make it possible to count the number of HSE/2 periods elapsed during HSETRIM_TICKS with the external reference clock precision.

The _hse_measure_frequency function of the firmware provided with this document makes it possible to get this number of periods by launching one TIM1 CH3 PWM cycle, and returning the value of the TIM2 counter.

The function HSETRIM_Process is used to tune HSE oscillator, using binary search to find the best load capacitance value HSETRIM, to be close to 32 MHz.

The tuning parameter that sets the load capacitance can take values from 0 to 63. The function starts by setting it to 32 (RCC_ECSCR1[13:8] = HSETRIM = 32), then it makes a measurement of the current HSE frequency. The measurement duration can be adjusted with the macro HSETRIM_TICKS, set by default to 1000 ms. At most, the binary search algorithm finds the best HSE load capacity in $\log_2 64$ * HSETRIM_TICKS, so, by default, in 6 seconds. The default HSETRIM_TICKS is conservative, adapt it if needed.

The load capacitance HSETRIM value found is then saved in the OTP or in the flash memory.

Firmware configurations

Two project configurations are available in the package:

- RCC_HSE_AutoCalib_Set_Calibration for the calibration procedure
- RCC_HSE_AutoCalib_Test_Calibration to test the stored value

This last configuration is given as an implementation example of the HSE clock initialization in RF applications.

The firmware is built on the STM32WBA HAL & LL drivers.

- RCC_HSE_AutoCalib_Set_Calibration: the device is programmed to:
  - modify and set the load capacitance HSETRIM value following the trimming algorithm presented before
  - store the load capacitance value together with the 112 bits of additional data (Bluetooth device address) in the OTP memory.
- RCC_HSE_AutoCalib_Test_Calibration: the device is programmed to test the actual HSE setting:
  - configure the HSE clock and output it on PA8 pin
  - fetch the load capacitance value from the OTP memory, and program HSETRIM from RCC_ECSCR1 register accordingly.

Storage in OTP bytes

The way to store tuning parameters in OTP memory is the same used for the manual trimming procedure for STM32WBA series (see *Section 3.2.2*).

8.2.3 Scripts

Two batch scripts are provided to run each firmware configuration:

1. RCC_HSE_AutoCalib_Set_Calibration.bat
2. RCC_HSE_AutoCalib_Test_Calibration.bat

These scripts call the STM32CubeProgrammer in command-line mode, and the path to the tool must be set accordingly: SET CLI="C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI.exe".

To be executed in SRAM, the Reset_Handler and Main Stack Pointer must be set accordingly, using the map file generated after the compilation process. In the script example, this is done automatically by parsing the .map file generated by the IDE.

The 112 bits of additional data are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

9 Automatic frequency trimming procedure example for the STM32WL series

The firmware associated to this document is available as an STM32Cube Expansion Package (X-CUBE-CLKTRIM_vx.y\Projects\NUCLEO-WL55JC\RCC_HSE_AutoCalib_SingleCore).

Note: The trimmed NUCLEO-WL55JC board needs to be configured to use the on-board crystal as HSE instead of the TCXO (which adjusts its frequency autonomously, according to the temperature). For details refer to UM2592, available on www.st.com.

9.1 Procedure description

The procedure consists in measuring the HSE clock generated inside the device from the external crystal. The measure is done internally to the STM32 system and is based on an accurate external 16 MHz clock reference provided by the user on PA9.

Note: An external reference is mandatory since no such accurate internal one exists in the device.

An automatic tuning of the load capacitance is performed by a specific STM32 application to reach the best accuracy of the HSE clock. The load capacitance values are then stored inside a nonvolatile location of the device (a dedicated area of the user flash memory, or in the OTP programming area).

Flash or OTP memory programming is done with a double-word granularity (64 bits). In order to save OTP bytes (1 K in the STM32WL series), the IN and OUT load capacitance values, each one on 6 bits, can be appended to a 64-bit wide structure with other personalization data (such as device address, product specific code, key).

This procedure can be repeated several times, only the last setup is active.

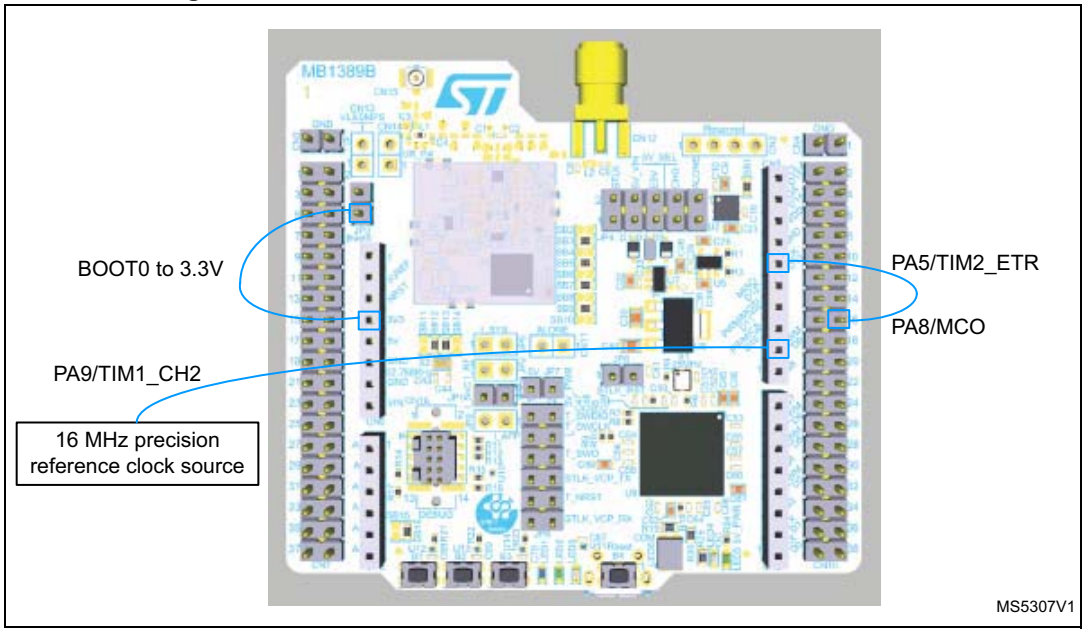
Once the procedure is completed, the active load capacitance values can be retrieved at startup (in the clock configuration function) and the HSE configuration register set accordingly.

9.2 Implementation

9.2.1 Hardware setup

Figure 19 shows the automatic HSE calibration procedure for an STM32WL series Nucleo board.

Figure 19. Automatic calibration overview - STM32WL series



A precision clock generator (better than 0.1 ppm) must be connected to pin PA9/TIM1_CH2, and set to generate a 16 MHz, 3.3 V_{PP} square wave. To see how to configure the board to boot from SRAM refer to *Section 5.2.1: Hardware setup*.

Note: A standard signal generator does not give enough accuracy for this application.

Hardware connections principle

The ways to establish connections and to implement TIM1 and TIM2 are the same than for the automatic trimming procedure for STM32WB series (see *Section 6.2.1*).

9.2.2 Software implementation

Firmware tuning algorithm principle

The timer links described in *Section 6.2.1* make it possible to count the number of HSE/2 periods elapsed during REF time with the external reference clock precision.

The HSE_Measurement function of the firmware provided with this document can be used to get this number by launching one TIM1 CH3 PWM cycle and returning the value of the TIM2 CH2 counter.

The function HSE_Tuning exploits HSE_Measurement to tune HSE. It follows a dichotomy logic to find the best load capacitance value for HSE, to be the closest to 32 MHz.

The tuning parameters that set the IN and OUT load capacitances can take values from 0 to 47. The function starts by setting it both to 24 (SUBGHZ_HSEINTRIMR[5:0] = 24 and SUBGHZ_HSEOUTTRIMR[5:0] = 24), then it makes an HSE/2 measurement.

If the frequency measured (calculated thanks to the number of periods returned by HSE_Measurement) is above 16 MHz, the function increases SUBGHZ_HSEINTRIMR to lower the frequency. If it is below, it decreases it to increase the frequency.

The variations of SUBGHZ_HSEINTRIMR are done with a step starting with a value of 12 when SUBGHZ_HSEINTRIMR is 24. After each measurement, this step is divided by two to calculate the new SUBGHZ_HSEINTRIMR value to test.

For each measurement, the parameter tuned is changed. That is to say, a first measurement is done and SUBGHZ_HSEINTRIMR is adjusted, another measurement is done and SUBGHZ_HSEOUTTRIMR is adjusted (and the adjustment principle explained above for SUBGHZ_HSEINTRIMR is followed), then another measurement is done and SUBGHZ_HSEINTRIMR is adjusted again, and so on.

So far, the measurements duration is 100 ms. When the step value reaches a value of 1 for one of the two parameters (the load capacitance approaches its best value), the duration is set to 1000 ms to be even more precise.

When the step reaches 1, if the HSE/2 frequency measured is below 16 MHz, the function decrements successively SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR, and makes new measurements and adjustments until the HSE/2 frequency measured exceeds 16 MHz. Once above 16 MHz, the function keeps the SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR pair between the one above and the last one below that got the best HSE precision.

The reasoning is the same if the last 100 ms measurement (the last one before the step has reached 1) is above 16 MHz.

Finally, the SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR pair found is saved in a nonvolatile memory.

Firmware configurations

The constant STORE_ADDRESS in hse_trimming.h allows the user to choose to save the SUBGHZ_HSEINTRIMR and SUBGHZ_HSEOUTTRIMR pair in OTP (STORE_ADDRESS = 0x1FFF7000U) or flash memory (STORE_ADDRESS = 0x0803F800 for example).

Also two project configurations are available in the package, one for the calibration procedure (RCC_HSE_AutoCalib_SingleCore), and another one to test the stored value (RCC_HSE_AutoCalib_SingleCore_Test). This last configuration is given as an implementation example of the HSE clock initialization in RF applications.

The firmware is built on the STM32WL HAL drivers.

- RCC_HSE_AutoCalib_SingleCore: the device is programmed to
  - Send the HSE clock on the PA8 pin, configure the timers and exploit the external clock source
  - Modify and set the IN and OUT load capacitance values following the trimming algorithm presented before
  - Store the IN and OUT load capacitance values together with the 40 bits of additional data in the OTP or in the selected flash memory area.
- RCC_HSE_AutoCalib_SingleCore_Test, to test the actual HSE settings saved:
  - Configure the HSE clock and output it on PA8 pin
  - Fetch the load capacitance value from the OTP/flash memory area
  - Program SUBGHZ_HSEINTRIMR and SUBGHZ_ HSEOUTTRIMR registers accordingly

Storage in OTP

The way to store tuning parameters in the OTP memory is the same as the one for the manual trimming procedure for STM32WL series (see *Section 5.2.2*).

9.2.3 Scripts

Two batch scripts are provided to run each FW configuration:

- RCC_HSE_AutoCalib_SingleCore_OTP.bat
- RCC_HSE_AutoCalib_SingleCore_Test_OTP.bat

These scripts call the STM32CubeProgrammer command-line interface, and the path to the tool must be set accordingly:

```
SET CLI="C:\Program Files (x86)
\STMicroelectronics\STM32Cube\STM32CubeProgrammer\bin\STM32_Programmer_CLI
.exe"
```

The binary or hex file of the configuration must be set also

- For IAR: `SET BINFILE="RCC_HSE_AutoCalib_SingleCore\Exe\RCC_HSE_AutoCalib_SingleCore.bin"`
- For Keil: `SET HEXFILE="RCC_HSE_AutoCalib_SingleCore\Exe\RCC_HSE_AutoCalib_SingleCore.hex"`

The 40 bits of additional data are transmitted to the device through these scripts. They are stored inside the SRAM and read by the FW.

```
STM32_Programmer_CLI.exe -c port=swd -w32 0x20006FF0  0x22334455
STM32_Programmer_CLI.exe -c port=swd -w32 0x20006FF4  0x00000011
```

10 STM32CubeMonitor-RF frequency trimming procedure example for the STM32WB series

The firmware and scripts associated to this document are available as an STM32Cube software expansion (X-CUBE-CLKTRIM_vx.y\Projects\P-NUCLEO-WB55.Nucleo\RCC_HSE_MonitorRFCalib).

Note: STM32CubeMonitor-RF procedure has been developed only for the STM32WB series, as the software does not support the STM32WL series.

10.1 Procedure description

The procedure is the same as the one detailed in *Section 3.1: Procedure description*.

10.2 Procedure steps

Details for these steps are given in *Section 10.3.2*.

This procedure tunes the STM32WB HSE clock using STM32CubeMonitor-RF. Thus, a connection between the two entities must be established.

To allow this connection, two softwares have to be flashed in the STM32WB cores:

- the Bluetooth Low Energy transparent mode firmware in the Cortex® M4 core
- the Bluetooth Low Energy stack in the Cortex® M0+ core.

Once the flashing is done, connect the STM32WB to the PC running STM32CubeMonitor-RF.

Go to the Script section in STM32CubeMonitor-RF.

Begin by running mco_output_config.txt to output HSE on pin PA8.

Modify hse_tunning.txt with the HSETUNE tuning parameter to test, and run it.

Measure the frequency of HSE on PA8 and adapt the tuning parameter in hse_tunning.txt if needed, before running it again.

When a satisfying parameter has been found, insert it in trim_param_flash.txt or trim_param_otp.txt, depending on the memory (flash or OTP) where it is saved. Then run the chosen script.

To save in the flash memory the selected page may have to be erased (erase operations are only possible at the page level), using the erase_flash_page.txt script.

Finally, the tuning parameter is saved in a nonvolatile memory.

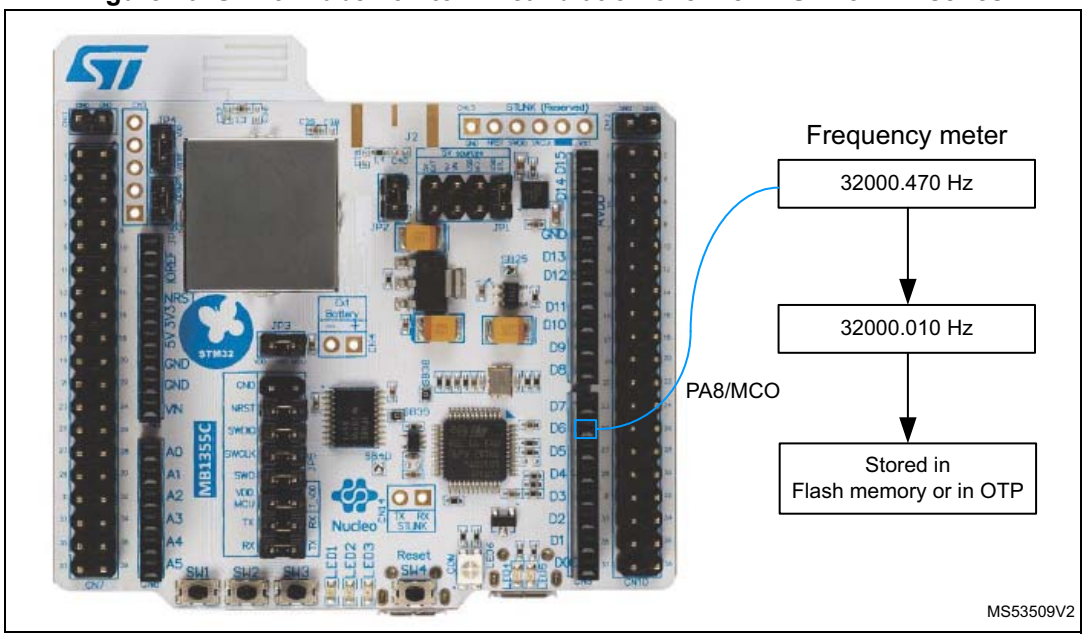
The C code retrieve_trimming_values gives an example for retrieving the parameter saved in memory.

10.3 Implementation

10.3.1 Hardware setup

Figure 20 shows the STM32CubeMonitor-RF HSE calibration procedure for an STM32WB series Nucleo-68 board.

Figure 20. STM32CubeMonitor-RFcalibration overview - STM32WB series



A precision frequency meter (better than 0.1 ppm) must be connected to pin PA8/MCO and set to detect a 32 MHz 3.3 V square wave peak to peak signal.

Note: Standard oscilloscopes are not sufficiently accurate for this kind of measurement.

10.3.2 Software and scripts setup

Flashing transparent firmware in the Cortex® M4 core

The Bluetooth Low Energy transparent mode firmware is available in the STM32Cube_FW_WB package:

- STM32Cube_FW_WB_Vx.y.z\Projects\P-NUCLEO-WB55.Nucleo\Applications\BLE\BLE_TransparentMode

This project can be opened in the user favorite IDE and must run on the Cortex® M4 core.

Flashing Bluetooth Low Energy stack in the Cortex® M0+ core

The Bluetooth Low Energy stack is available in the STM32Cube_FW_WB package:

- STM32Cube_FW_WB_Vx.y.z\Projects\STM32WB_Copro_Wireless_Binaries\stm32wb5x_BLE_Stack_fw.bin

The instructions to flash this stack into the Cortex® M0+ core are given in the release notes file: STM32Cube_FW_WB_Vx.y.z\Projects\STM32WB_Copro_Wireless_Binaries\Release_Notes.html

Note: For more information on transparent mode, Bluetooth Low Energy stack and STM32CubeMonitor-RF relations, refer to UM2288 "STM32CubeMonitor-RF software tool for wireless performance measurements", available on www.st.com.

Connecting the product to STM32CubeMonitor-RF

If a Nucleo board is used, simply link it to the PC via USB. If not, the user can communicate with the transparent mode firmware through the USART interface of the MCU, available on pins PB6/RX and PB7/TX.

Note: PB6 and PB7 are used by the ST-Link VCP (Virtual COM Port) on Nucleo boards.

10.3.3 Scripts

Note: The scripts presented below are available in the STM32Cube software expansion package (X-CUBE-CLKTRIM_vx.y\Projects\P-NUCLEO-WB55.Nucleo\RCC_HSE_MonitorRFCalib).

mco_output_config.txt

No need to modify, run it in STM32CubeMonitor-RF to output HSE on PA8.

hse_tuning.txt

This script includes the HSETUNE value to test. This value is the last parameter in the script line below (line 11) and is set to 0x18 for this example.

```
Send(VS_HCI_C1_WRITE_REGISTER;0x04;0x0000FF00;0x5800009C;0x00001800)
```

This line updates the HSETUNE bits of the RCC_HSECR register.

To test an HSETUNE tuning parameter, change this value as wanted and run the script in STM32CubeMonitor-RF.

Then, use the frequency meter (HSE measurement on PA8) to evaluate the quality of the HSETUNE value fixed: the closer the HSE frequency to 32 MHz, the better the HSETUNE quality.

Note: In BLE_TransparentMode the system clock is HSE. In this script the HSE is off, hence switch the system clock to another one before setting HSE to off.

erase_flash_page.txt

This script can be used to partially clean the flash memory to save new tuning parameters in an already full page.

Note: The running of the script is not mandatory for the procedure.

The number of the page to erase is the last parameter in the script line below (line 20), it is set to 0xA0 for this example (since the trim_param_flash.txt script delivered saves the parameter at this page).

```
Send(VS_HCI_C1_WRITE_REGISTER;0x04;0x000007F8;0x58004014;0x00000500)
```

This line updates the PNB bits of the FLASH_CR register.

Note: To traduce the page number (0x0500 → 0xA0) take into account the mask 0x7F8.

trim_param_flash.txt

This script saves an HSETUNE value in flash memory at a given address.

It must be modified with the HSETUNE value found with HSE_tuning.txt to get the best HSE frequency, with other information to complete a 64-bit data and with the address where these data must be saved.

The script lines below (lines 19 to 25) include the HSETUNE value to modify, and other information that can be modified.

As an example, the HSETUNE value saved is 0x1B.

```
#Write lower BD Address : Company Id = 0xE105 + Board Id = 0x7777 (for example)
Send(VS_HCI_C1_WRITE_REGISTER;0x04;0xFFFFFFFF;0x080A0000;0xE1057777)
Wait(100)
#Write upper BD Address : Id = 0x0 + HSE_Tun = 0x1B(for example) + Upper BD = 0x0080
Send(VS_HCI_C1_WRITE_REGISTER;0x04;0xFFFFFFFF;0x080A0004;0x001B0080)
Wait(100)
```

These lines save in memory a 64-bit data equal to 0x 001B 0080 E105 7777 (64 bits is the size of a flash memory word line).

The other 58 bits apart from the HSETUNE value can be chosen as wanted. For example, they can be the Bluetooth device address.

Moreover, 0x080A0000 corresponds to the address where the 64-bit data is saved (the 32 lower bits at 0x080A0000 and the 32 upper bits at 0x080A0004).

trim_param_otp.txt

trim_param_otp.txt is equivalent to trim_param_flash.txt: the user must modify the same elements for the script to get the requested behavior.

However, the saving address is not the same, as OPT and flash memory are mapped at different addresses in the STM32WB microcontrollers. In the delivered script, the address is close to the beginning of the OTP memory: 0x1FFF7008 and 0x1FFF700C.

10.3.4 C code

retrieve_trimming_values.c

This C code is given as an example of implementation for the HSE clock initialization procedure for RF applications, when tuning parameters are saved in OTP memory.

It gathers:

- Two constants, CFG_OTP_BASE_ADDRESS and CFG_OTP_END_ADDRESS, localizing the beginning and the ending of the OTP memory.
- The structure OTP_ID0_t representing the 64-bit data saved in memory by trim_param_otp.txt.
- The OTP_Read function that seeks the most recent 64-bit data saved in OTP memory with a 0 ID (0 has been chosen in this application note as the index for HSETUNE data in OTP memory).
- The Config_HSE function that exploits OTP_Read to find the most recent HSETUNE value in OTP memory so as to set the HSETUNE bits in the RCC_HSECR register and to get a satisfying HSE frequency.

Note: The same code can be used for flash memory by changing the two constant values with flash memory addresses.

11 Selection of a compatible HSE crystal for the STM32WB series

The crystal used for the HSE is an essential element for the performance of the radio system, the requirements are described in the datasheets. The next step is the calculation of the minimum transconductance of the selected oscillator.

Note: As already specified, for a crystal with low load capacitance (6 pF) and with not negligible parasitic layout capacitance, frequency fine tuning to 32 MHz with 0 ppm can be difficult to achieve. The trimming range is centered on crystals with a load capacitance of 8 pF, the recommended load capacitance. Crystals with 7 pF load capacitance can be used, but need special hardware layouts.

Assuming $C_{L1} = C_{L2}$, and that the crystal sees on its pads the C_L value given by the manufacturer (see *Figure 1*), the minimum transconductance is $g_{mcrit} = 4 * ESR * (2 * \pi * F)^2 * (C_0 + C_L)^2$, where C_0 is the shunt capacitance, C_L the nominal load capacitance, and F the nominal oscillation frequency. As an example, with the NX2016SA 32 MHz - EXS00A-CS06654 crystal, F = 32 MHz, C_L = 8 pF, C_0 = 0.6 pF, and ESR = 60 Ω, so $g_{mcrit} = 4 * 60 * (2 * \pi * 32 * 10^6)^2 * (0.6 * 10^{-12} + 8 * 10^{-12})^2 = 0.717$ mA/V.

Because of the limitation described in the STM32WB errata sheet, $g_{mcrit} \leq 1.13$ mA/V. It is recommended to keep HSEGMC it at its default (maximum) value 011 (current maximum limit 1.13 mA/V). Do not select a crystal whose g_{mcrit} does not respect this requirement.

The g_{mcrit} for crystal NX2016SA 32 MHz - EXS00A-CS06654 is safely below 1.13 mA/V, making it compatible with STM32WB devices. Follow hardware layout recommendations for placing the crystal and for its routing, detailed in AN5165.

Note: Due to the design of the BLE radio, it is not recommended to modify by software the CFG_BLE_HSE_STARTUP_TIME parameter. The default value 0x148 (800 µs) ensures that HSE frequency is stable for BLE over all voltage and temperature conditions. As the CPU2 is woken up 1 ms prior to Bluetooth Low Energy scheduled transaction, decreasing CFG_BLE_HSE_STARTUP_TIME does not bring any significant power saving.

HSERDY bit does not signal that HSE crystal frequency is stable enough for Bluetooth Low Energy communication. After the HSERDY flag is raised, some time is needed for the oscillation frequency stabilization. To be sure that HSE is ready for Bluetooth Low Energy communication, monitor HCI_HARDWARE_ERROR_EVENT (see AN5270). If this event is not generated, HSE frequency is correct during the communication.

Setting HSES bit does not decrease HSE start-up time. HSE oscillations are detected sooner, and HSERDY is triggered a dozen of microseconds sooner. Setting HSES does not modify the consumption (the peak current is negligible because it is very short).

Troubleshooting:

- Verify that HSE crystal is able to start (HSERDY is set)
- Verify HSE crystal parameters with respect to g_{mcrit}
- Verify that HSE frequency is tuned
- Check HCI_HARDWARE_ERROR_EVENT occurrence

If the crystal is compatible according to calculations, but still does not operate correctly, most likely issues are caused by the PCB design.

12 Conclusion

To ensure best performance, RF applications require a very accurate clock.

The RF clock is derived from an external crystal, and the frequency fine tuning is obtained by setting the right load capacitance at crystal pins.

STM32 wireless MCUs introduce a very efficient architecture with internal capacitances setting, removing the need for extra components on PCB, and lightening the constraints on the crystal performance.

13 Revision history

Table 7. Document revision history

Date	Revision	Changes
27-Sep-2017	1	Initial release.
14-Nov-2017	2	Updated: – *Section 3.2.2: Software implementation* – *Section 3.2.3: Scripts*
21-Feb-2019	3	Changed document classification, from ST restricted to Public. Updated *Section 1: HSE oscillator*. Minor text edits across the whole document.
30-Jan-2020	4	Introduced STM32WL series, hence updated document title, *Introduction* and *Section 12: Conclusion*. Updated *Section 1: HSE oscillator*, *Section 1.2: STM32 wireless MCUs architecture*, *Section 1.3: HSE configuration parameters - STM32WB series*, *Section 1.8: Crystal references*, *Section 3: Manual frequency trimming procedure example for the STM32WB series*, *Load capacitance setting* and *Project configurations*. Added *Section 1.5: HSE configuration parameters - STM32WL series*, *Section 2: Trimming methods comparison*, *Section 6: Automatic frequency trimming procedure example for the STM32WB series* and *Section 10: STM32CubeMonitor-RF frequency trimming procedure example for the STM32WB series*. Updated *Table 4: Crystal specifications*. Minor text edits across the whole document.
04-Feb-2020	5	Updated *Introduction*.
27-Apr-2020	6	Updated *Section 1: HSE oscillator*, *Section 1.6: Board implementation (STM32WB series)*, *Section 1.8: Crystal references* and *Section 3: Manual frequency trimming procedure example for the STM32WB series*. Added *Section 1.9: Tuning in production*. Updated *Table 1: Carrier accuracy requirement for RF protocols*. Minor text edits across the whole document.
26-May-2020	7	Added *Note:* in *Section 2: Trimming methods comparison*.

Table 7. Document revision history (continued)

Date	Revision	Changes
29-Jun-2020	8	Updated *Introduction*, *Section 1.3: HSE configuration parameters - STM32WB series*, *Section 1.9: Tuning in production*, *Section 3: Manual frequency trimming procedure example for the STM32WB series*, *Section 3.2.1: Hardware setup*, *Section 3.2.3: Scripts*, *Section 6: Automatic frequency trimming procedure example for the STM32WB series*, *Section 6.2.1: Hardware setup*, *Firmware configurations*, *Section 6.2.3: Scripts*, *Section 10: STM32CubeMonitor-RF frequency trimming procedure example for the STM32WB series*, *Section 10.3.2: Software and scripts setup* and *Section 10.3.3: Scripts*. Added footnotes to *Table 5: Trimming methods* and *Table 6: Comparison of trimming methods*. Updated *Figure 6: OB configuration to boot from SRAM with BOOT0 value driven by PH3 pin*, *Figure 7: OB configuration to boot from SRAM with BOOT0 value driven by option bit nBOOT0*, *Figure 13: Automatic calibration overview - STM32WB series* and *Figure 20: STM32CubeMonitor-RFcalibration overview - STM32WB series*.
05-Nov-2020	9	Introduced STM32WL series, hence updated *Introduction* and added *Section 5: Manual frequency trimming procedure example for the STM32WL series*, *Section 9: Automatic frequency trimming procedure example for the STM32WL series* and their subsections.
26-Jan-2021	10	Updated document title, *Introduction*, *Section 1: HSE oscillator*, and *Section 12: Conclusion*. Added *Section 11: Selection of a compatible HSE crystal for the STM32WB series*. Minor text edits across the whole document.
03-Nov-2022	11	Updated *Section 1.1: Crystal oscillator*, *Current control: HSEGMC[2:0]* and *Section 11: Selection of a compatible HSE crystal for the STM32WB series*. Updated *Table 2: Oscillator pin numbers for the STM32WB series*. Minor text edits across the whole document.
17-Jan-2023	12	Added note in *Section 10.3.3: Scripts*. Minor text edits across the whole document.
21-Apr-2023	13	Updated document title, *Section 1.8: Crystal references*, *Section 11: Selection of a compatible HSE crystal for the STM32WB series*, and *Section 12: Conclusion*. Minor text edits across the whole document.
13-Jun-2023	14	Document scope extended to STM32WBA series, hence updated *Introduction* and *Section 1.2: STM32 wireless MCUs architecture*. Added *Section 1.4: HSE configuration parameters - STM32WBA series*, *Section 1.7: Board implementation (STM32WBA series)*, *Section 4: Manual frequency trimming procedure example for the STM32WBA series*, *Section 7: Legacy automatic frequency trimming procedure example for the STM32WBA series*, and their subsections. Updated *Table 4: Crystal specifications* and *Table 6: Comparison of trimming methods*. Minor text edits across the whole document.