AN3101
Application note

STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx internal RC oscillator calibration

Introduction

The STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers offer the possibility of using internal RC oscillators HSI (High-speed internal factory trimmed oscillator of 16 MHz, typically) or LSI (Low-speed internal low-consumption oscillator of 38 kHz, typically) as clock sources.

The operating temperature and voltage have an impact on RC accuracy. At 3 V and 25 °C, the HSI has an accuracy of ±1% typically, but in the full temperature and voltage ranges, the accuracy of the HSI frequency decreases to -5% / +5% (STM8L05xxx), -4.5% / +3% (STM8L15xxx and STM8L162xx) and -4% / +4% (STM8AL31xx/3Lxx).

To compensate for the influence of temperature and voltage on internal RC oscillator accuracy, the STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers provide the capability of calibrating the HSI clock and of measuring the LSI clock.

This application note gives two methods of calibrating the HSI internal RC oscillator. The first method is based on finding the frequency with the minimum error and the second one consists in finding the maximum allowed frequency error. Both are implemented by providing an accurate reference source.

The measurement of the LSI clock is performed by connecting the LSI oscillator to a timer input capture. Depending on the measured value, peripheral registers are updated to meet user requirements.

This application note applies to the products listed in *Table 1*.

Table 1. Applicable products

Type	Product lines
Microcontrollers	– STM8L051/052 Value Line – STM8L151/152 – STM8L162 – STM8AL31 – STM8AL3L

Contents

List of tables

List of figures

1 System clock for STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx

The STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers offer different clock sources that can be used to drive the system clock:

- 16 MHz high-speed internal (HSI) factory-trimmed RC clock
- 1 to 16 MHz high-speed external (HSE) oscillator clock
- 32.768 kHz low-speed external (LSE) oscillator clock
- 38 kHz low-speed internal (LSI) low-consumption clock

The internal RC oscillators (HSI and LSI) have the advantage of providing a low-cost clock source (no external components required). It also has a faster startup time than the external oscillator. However, even with calibration, the internal RC oscillator frequency is less accurate than the frequency of an external crystal oscillator or ceramic resonator.

Note: The internal oscillator can also be used as backup source (auxiliary clock) if the external oscillator fails.

2 High speed internal oscillator calibration

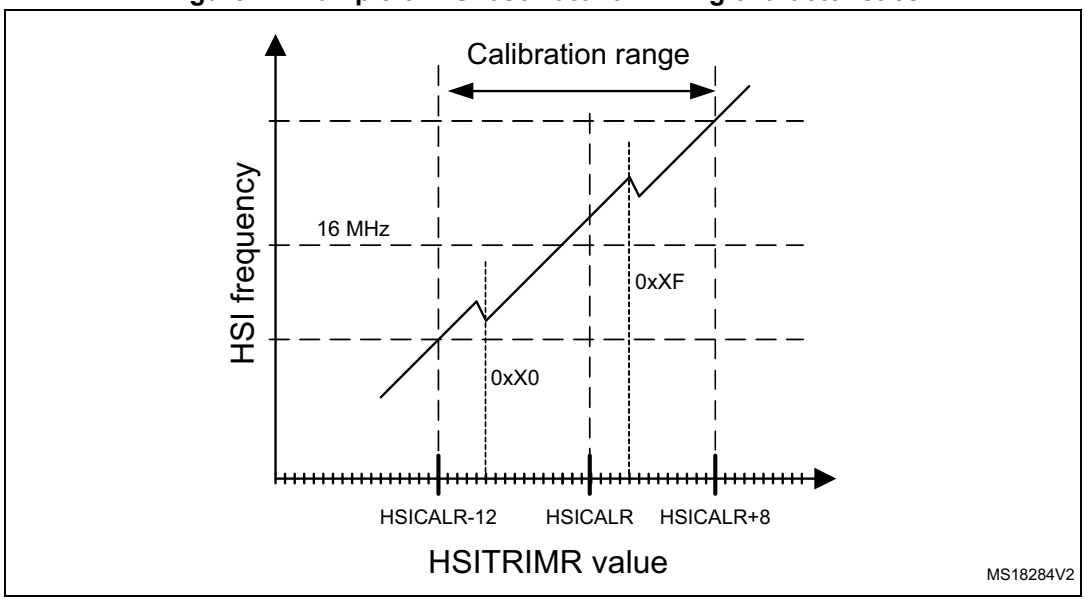
HSI oscillator frequencies may vary from one chip to another due to manufacturing process variations. For this reason, each device is factory-calibrated by ST to have a 1% accuracy at T_A = 25 °C. After reset, the factory calibration value is automatically loaded in an internal calibration HSICALR register.

The frequency of the HSI RC oscillator can be trimmed to achieve a better accuracy with wider temperature and supply voltage ranges, the HSITRIMR register is used for this purpose.

To modify the content of the HSITRMR register the access has to be granted by successfully unlocking the write protection mechanism which consists of two consecutive write accesses to the HSIUNLCKR register, the first one with the value 0xAC and the second one with the value 0x35. This must be immediately followed by a write access to the HSITRIM register. Once this register is configured, its value is used instead of the HSICALR register values.

The HSI oscillator frequency increases monotonically with the increasing values in the HSITRIMR register as long as the trimming value changes in the low nibble (4-bits). Any change to the HSITRIMR value in the high nibble can cause a non-monotonicity in the graph of the trimming function resulting in a sawtooth shape, as exemplified in *Figure 1*.

Figure 1. Example of HSI oscillator trimming characteristics



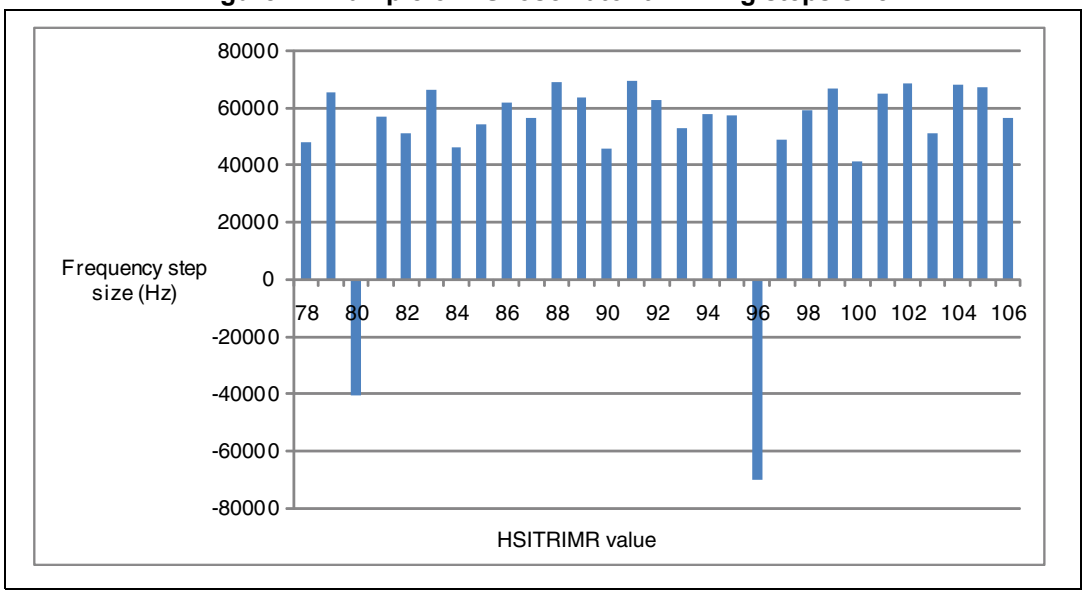
The maximum HSI frequency variation due to the temperature and/or voltage drift is 16 MHz + 3%, that is, 16 MHz + 480 kHz. Please, refer to the datasheet.

8 trimming steps (each of about 64 kHz) are required to compensate for this variation and the upper trimming threshold is therefore required to be HSICALR + 8.

The minimum HSI frequency variation due to the temperature and/or voltage drift is 16 MHz - 4.5%, that is 16 MHz - 720 kHz. To be able to compensate for this variation, 12 trimming steps (each of about 64 kHz) are required to compensate for this variation and the lower trimming threshold must be HSICALR -12.

The frequency step size for two consecutive HSITRIMR values is not constant. It is typically 0.4% (max. 0.7%) for the monotonous range (change in the low nibble only). The frequency step size for two consecutive HSITRIMR values which differ in the high nibble (i.e. from 0x3F to 0x40) can be up to +/- 1.5%. See the *Figure 2* and the datasheet for details.

Figure 2. Example of HSI oscillator trimming steps size



Note: This graph is given for guidance only. It is based on a device whose calibration value is equal to 90.

2.1 Calibration principle

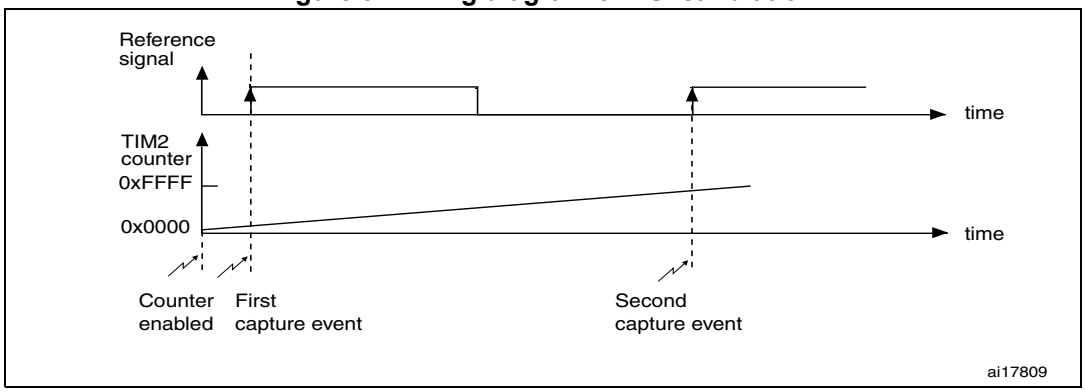
The calibration principle consists in

- first measuring the HSI frequency for many calibration values, then
- computing the frequency error for each calibration value, and
- finally setting the HSITRIMR register with the optimum value.

The HSI frequency is not measured directly but it is estimated from the number of counted HSI clock pulses using a timer compared with the typical value (16 MHz). To do so, a very accurate reference frequency must be available such as the LSE frequency provided by the external 32.768 kHz crystal or the 50 Hz/60 Hz of the mains (refer to *Section 2.2.2: Case where another source is used as the reference frequency*).

Figure 3 shows how the reference signal period is measured in number of timer counts.

Figure 3. Timing diagram of HSI calibration



After enabling the timer counter, when the first rising edge occurs, the timer counter value is registered in IC1ReadValue1. At the second rising edge, the timer counter is captured in IC1ReadValue2. The elapsed time between two consecutive rising edges (IC1ReadValue2 and IC1ReadValue1) represents an entire period of the reference signal.

Since the timer is clocked by the internal HSI oscillator, the microcontroller can compute the real frequency generated by the HSI versus the reference signal ($frequency_{REF}$)

$$frequency_{HSI} = (IC1ReadValue2 - IC1ReadValue1) \times frequency_{REF}$$

The error (in Hz) is computed as the absolute value of the difference between the measured HSI frequency ($Frequency_{HSI}$) and 16 MHz, hence the HSI frequency error is given by

$$Error(Hz) = |frequency_{HSI} - 16000000|$$

After calculating the error for each calibration value, the algorithm determines the optimum calibration value (the nearest value to 16 MHz) to be programmed in the HSITRIMR register (refer to *Section 2.3* for more details).

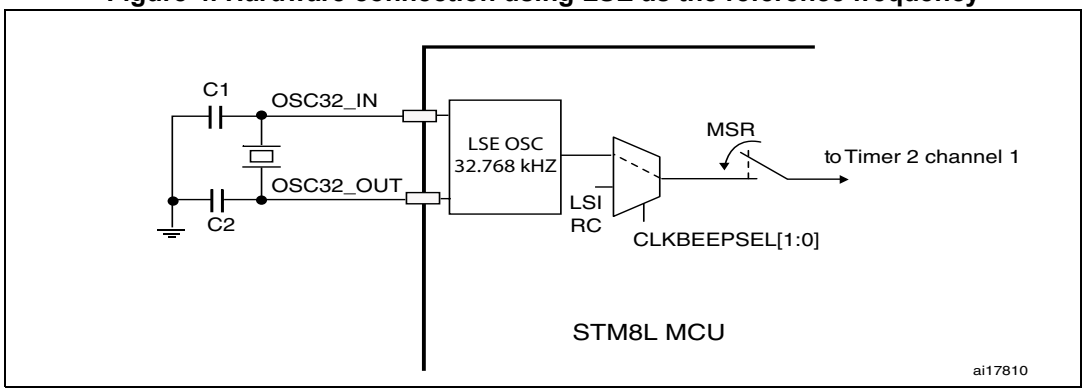
2.2 Hardware implementation

2.2.1 Case where LSE is used as the reference frequency

STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontroller devices offer a very useful feature, i.e. the ability to connect low speed oscillators (LSI or LSE) to Timer 2 channel 1. Thus, the LSE clock can be used as the reference signal for HSI calibration and no additional hardware connections are required. Only the LSE oscillator should be connected to OSC32_IN and OSC32_OUT.

Figure 4 shows the hardware connections needed for HSI calibration using LSE as an accurate frequency source for calibration.

Figure 4. Hardware connection using LSE as the reference frequency



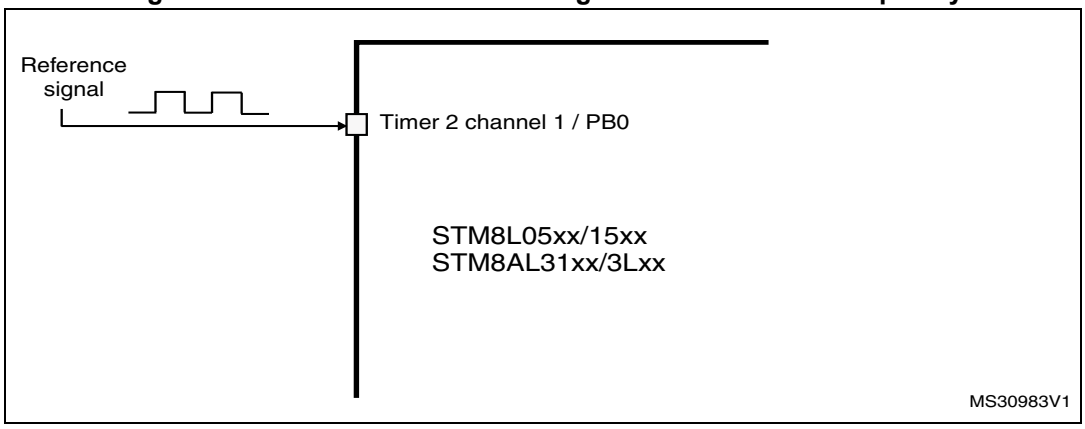
2.2.2 Case where another source is used as the reference frequency

Any signal with accurate frequency can be used for HSI calibration and the mains frequency is one of the possibilities.

*Note: Refer to the AN2868 application note ("*STM32F10xx *internal RC oscillator (HSI) calibration") for more details about using mains frequency for calibration.*

As indicated in *Figure 5*, the reference frequency must be connected to Timer 2 channel 1 (PB0 pin).

Figure 5. Hardware connection using external reference frequency



2.3 Description of the HSI clock calibration firmware

The HSI clock calibration firmware provided with this application note includes two major functions:

- `uint32_t HSI_CalibrateMinError(void)`
- `ErrorStatus HSI_CalibrateFixedError(uint32_t MaxAllowedError, uint32_t* Freq)`

2.3.1 HSI calibration with minimum error

The *HSI_CalibrateMinError()* function calibrates the HSI clock to have the nearest frequency to 16 MHz.

It measures 21 frequencies for different calibration values and provides the HSITRIM value that corresponds to the frequency with the minimum error. The resulting HSITRIM value is the calibration value that is written in the HSITRIMR register. The frequency measurement starts from HSITRIMR = HSICALR - 12 and ends with HSITRIMR = HSICALR + 8.

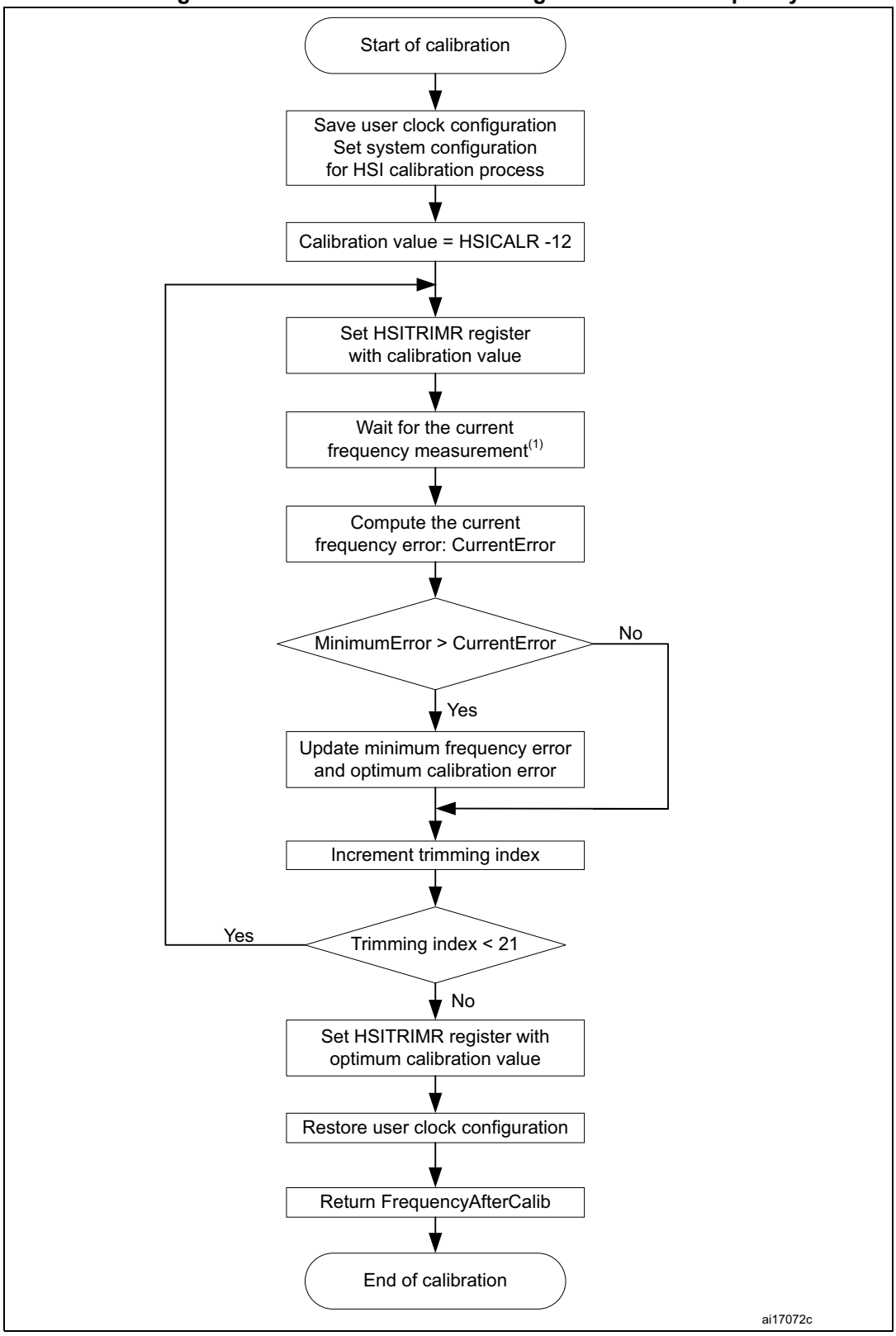
After calibration, the *HSI_CalibrateMinError()* function returns the HSI frequency value as an unsigned 32-bit integer (uint32_t). This value can be helpful to reconfigure prescalers, like the ones used for communication peripherals.

The flowchart in *Figure 6* gives the algorithm for this function.

Example

```
uint32_t HSIFrequencyAfterCalib = HSI_VALUE;
{
......
/* Calibrate HSI clock and return its value after calibration  */
HSIFrequencyAfterCalib = HSI_CalibrateMinError();
.........
}
```

Figure 6. HSI calibration flow: finding the minimum frequency error



1. Frequency measurement is described in *Section 2.3.3: HSI frequency measurement on page 15*.

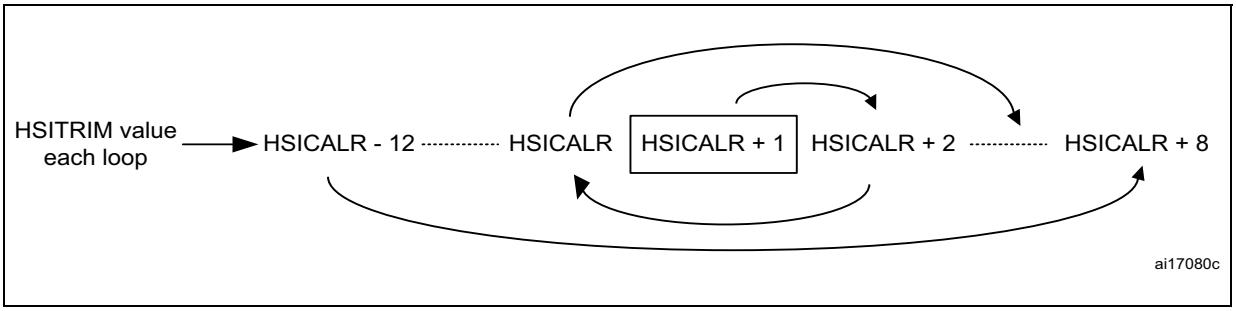
2.3.2 HSI calibration with fixed error

The *HSI_CalibrateFixedError()* function is provided to calibrate the HSI clock with a maximum allowed frequency error. It is configured by the user as an absolute value given in Hertz (the first parameter: *MaxAllowedError*). This function is the same as *HSI_CalibrateMinError()* (refer to *Section 2.3.1*.), but it searches for the frequency that has an error (in absolute value) less than or equal to *MaxAllowedError.*

- If it finds this frequency, it stops searching, configures the HSITRIMR register according to this frequency and returns SUCCESS, meaning that the calibration operation has been successfully performed.
- Otherwise, it continues searching for it until the HSITRIM bits = HSICALR -12 (21st frequency). It then sets the HSITRIMR register to the default calibration value and returns ERROR, meaning that the calibration has failed and did not find any frequency with an error less than or equal to *MaxAllowedError*.
  The frequency measurements start with HSTRIM = HSICALR + 1 (unlike in the *HSI_CalibrateMinError()* function where frequency measurements starts from HSICALR - 12 to end with HSICALR + 8). The HSITRIM value is computed in loops to find the next value: the HSITRIM value starts from HSICALR + 1, then goes to the next value to the right: HSICALR + 2, then to the next to the left: HSICALR, then to the second to the right HSICALR + 3 and so on until it reaches HSICALR + 8, forming a “spring loop” (as shown in *Figure 7*).

  This algorithm is based on the fact that the probability of finding the frequency that has the minimum error increases when the HSITRIM bit value tends to HSICALR. This algorithm is implemented to minimize the time consumed by the calibration process.

Figure 7. “Spring loop”



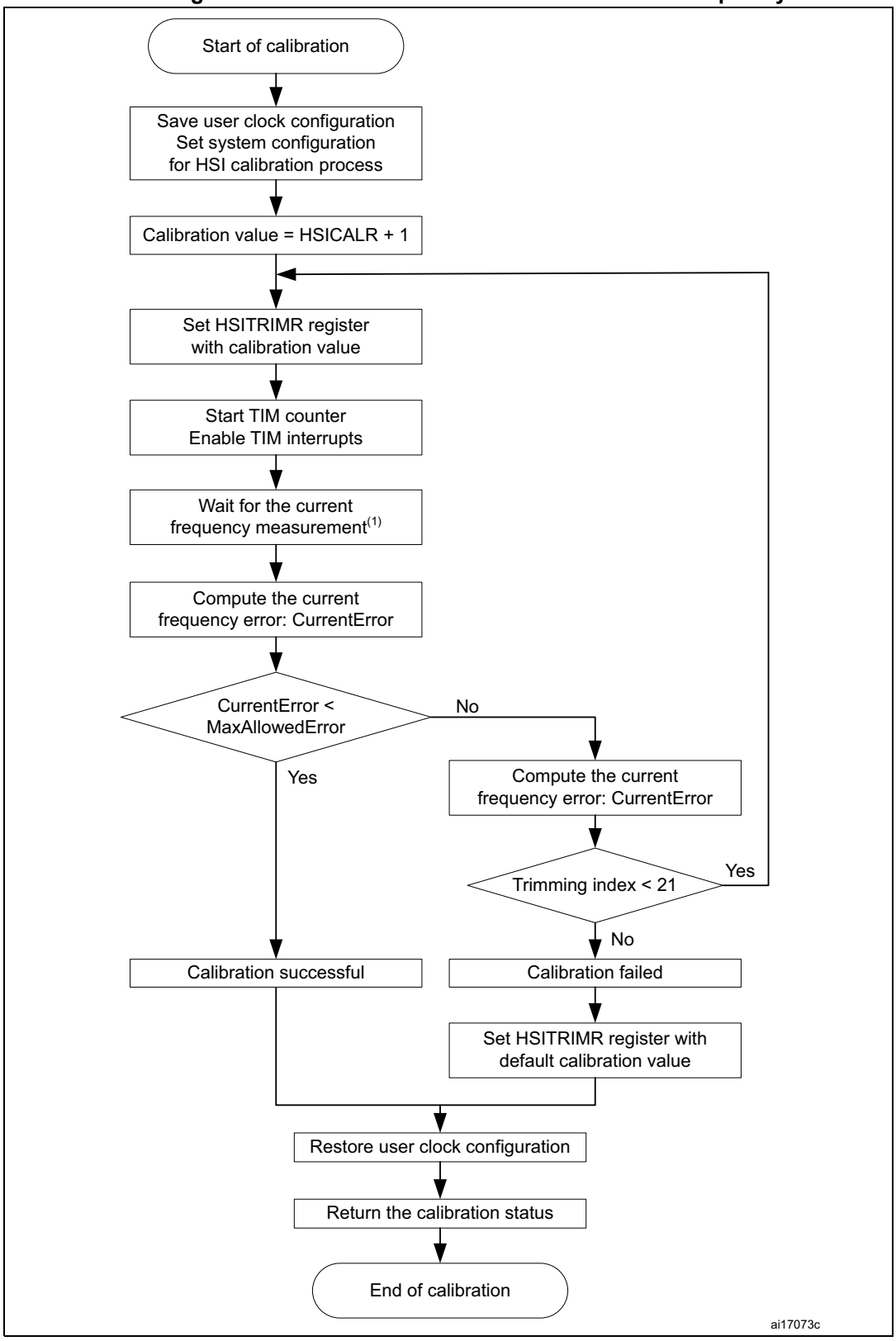
The second parameter is used to get the frequency (in Hertz) after calibration in the form of an unsigned 32-bit integer (unit32_t).

The flowchart in *Figure 8* gives the algorithm for this function.

Example

```
uint32_t CurrentHSIFrequency = 0;
ErrorStatus CalibStatus = ERROR;
......
{
......
/* Calibrate HSI with the maximum allowed error in Hz */
/* Set the maximum value of the error frequency at
     +/- 60 000 Hz -> 0.375 % */
CalibStatus = HSI_CalibrateFixedError(60000, &CurrentHSIFrequency);
if(CalibStatus != ERROR)
{
STM_EVAL_LEDOn(LED1);
LCD_SetCursorPos(LCD_LINE2, 0);
LCD_Print("    SUCCESS    ");
}
else
{
STM_EVAL_LEDOn(LED3);
LCD_SetCursorPos(LCD_LINE2, 0);
LCD_Print("      FAIL     ");
}
......
}
```

Figure 8. HSI calibration flow: maximum allowed frequency error



1. Frequency measurement is described in *Section 2.3.3: HSI frequency measurement on page 15*.

2.3.3 HSI frequency measurement

HSI frequency measurement is performed by Timer2 capture interrupt. In the timer capture event interrupt, an entire period of HSI frequency is computed. The number of periods to be measured for each trimming value is configurable by the user in the *hsi_calibration.h* file as follows:

```
#define HSI_PERIOD_NUMBERS 10 /* Number of periods to be measured = 10 */
```

The averaging method is used to minimize frequency error measurements. So, if the number of periods reaches `HSI_PERIOD_NUMBERS`, the average of all measured frequencies is computed.

You can easily configure the frequency of the reference source. It is defined in the *hsi_calibration.h* file as follows:

If the LSE clock is used as the reference frequency, uncomment the line below to make sure the LSE is configured and internally connected to Timer 2 channel 1:

```
#define USE_REFERENCE_LSE
```

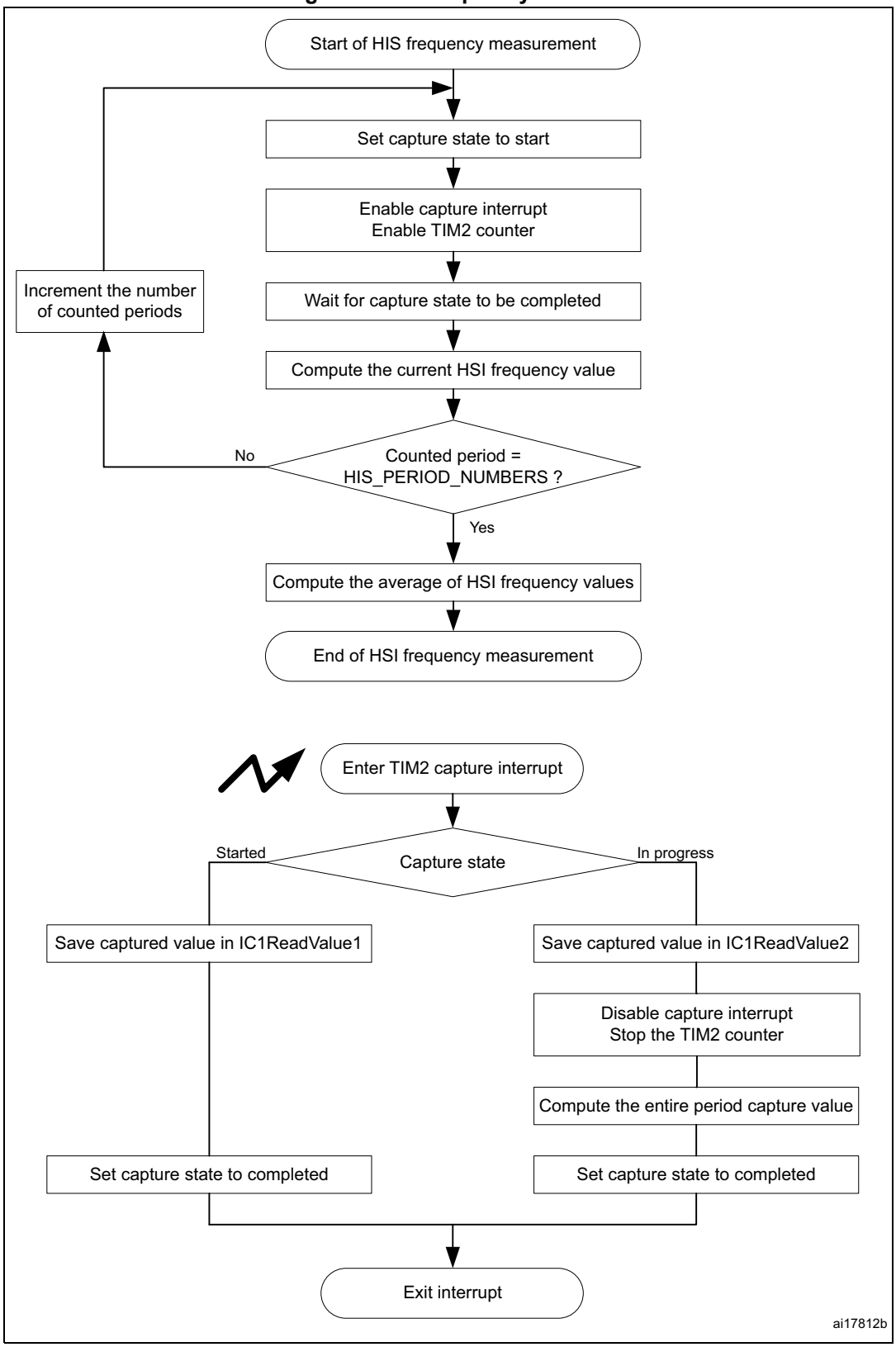
If the reference frequency is a mains source frequency equal to 50 Hz, then comment the line above and define the reference frequency as shown below:

```
#define REFERENCE_FREQUENCY 50 /* The reference frequency value in hertz */
```

The computation of the frequency measurements does not depend on the duty cycle of the source reference signal, but rather on its frequency, as the capture 1 interrupt is configured to occur on every rising edge of the reference signal (refer to *Figure 3*).

Figure 9 provides the frequency measurement algorithm.

Figure 9. HSI frequency measurement flow



2.3.4 HSI calibration demo description

The demo provided with this application note shows the ability of the firmware to calibrate the HSI oscillator of the STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers.

By default, the demo is configured to run the calibration of the HSI oscillator routine using the LSE source frequency.

To run the calibration process that provides the frequency with minimum error, you have to comment the following define line in the *main.c* file. Conversely, to run the calibration process that finds the frequency with a maximum allowed error, you have to uncomment the same line:

```
/* #define USE_HSI_FIXED_ERROR */
```

To display the HSI signal, connect an oscilloscope probe to the PC4 pin and uncomment the following line in the *main.c* file:

```
/* #define OUTPUT_HSI_ON_CCO_FOR_DEBUG */
```

2.4 Recommendations on the use of the HSI calibration library

1. If the reference frequency used for HSI calibration is lower than 250 Hz, the TIM2 counter prescaler can be used to support low reference frequencies.
2. The TIM2 input capture divider (1, 2, 4 and 8) can be used to support high reference frequencies.
3. Frequency measurement accuracy is not guaranteed when using a reference frequency that exceeds 240 kHz.
4. It is not recommended to call calibration functions in an interrupt routine since, in this case, the calibration process may be long (refer to the *Section 2.5.1*).
5. It is recommended to stop all application activities before the calibration process, and to restart them after calling the calibration functions. The application therefore has to stop communications, ADC measurements (see *Note: 1*) etc. since these processes are supposed to use clock configurations different from those used in the calibration process. Otherwise, errors might be introduced in the application: errors while reading/sending frames, ADC reading errors since the sampling time has changed, etc.

Note: 1 Except when using the ADC for the calibration process (refer to 7.).

6. The HSI calibration firmware uses the following peripherals: Clock (for trimming HSI), Timer 2 (for measuring HSI), Beeper (for connecting LSE to Timer 2 if LSE method is used). So, it is recommended to reconfigure these peripherals (if used in the application) after running HSI calibration routine
7. Real-time calibration vs. temperature can be used when the ambient temperature changes noticeably while the application is running. The internal temperature sensor can be used with the ADC watchdog with two thresholds. Each time an ADC watchdog interrupt occurs, a new HSI calibration process has to be performed and the two thresholds are updated according to the current temperature (this feature is not implemented in this application note):

   Threshold_High = CurrentTemperatureValue + TemperatureOffset

   Threshold_Low = CurrentTemperatureValue – TemperatureOffset

2.5 Calibration process performance

2.5.1 Duration of the calibration process

The duration of the calibration process depends on:

- the used reference frequency;
- the number of measured periods per frequency;
- the number of measured frequencies during the calibration process.

Once peripherals are configured and ready, the duration of the calibration process is approximated by:

$$\text{CalibDuration} = 2x\left[\frac{(\text{NPeriod}+1)\times \text{NFreq}}{f_{REF}}\right]$$

Where:

– *NPeriod* is the number of times the frequency is measured for each HSITRIM configuration (the same frequency)

– *NFreq* is the number of measured frequencies (number of HSITRIM values used for the frequency measurement)

– f_{REF} is the frequency value (in Hz) of the reference signal (after the input capture divider)

In the case of the calibration process with a minimum frequency error (*HSI_CalibrateMinError()*) the number of NFreq is equal to 21. If the LSE oscillator is used as the reference frequency (f_{REF} = LSE value / input capture divider = 32768/8 = 4096 Hz) and the selected number of measured periods configured by the user is 10, the calibration lasts around 113 ms.

$$\text{CalibDuration} = 2\times\left[\frac{11\times 30}{4096}\right] = 113\text{ms}$$

In the case of the calibration process with a maximum allowed error (*HSI_CalibrateFixedError()*), NFreq changes from chip to chip. NFreq also depends on the maximum allowed error that is selected. The higher the selected allowed error, the more NFreq will tend to 1. The lower the selected allowed error, the more NFreq will tend to 21.

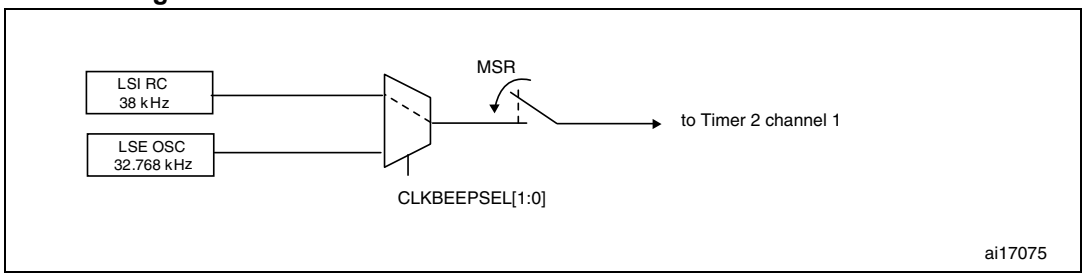
So, the duration of the calibration process with a maximum allowed error is lower than or equal to the duration of calibration when using the minimum frequency error process.

Note: The factor 2 in the CalibDuration formula above is due to the fact that there is no synchronization between reference signal and timer start counting.

3 Low speed internal oscillator measurement

The low-speed internal oscillator (LSI) is a low power clock source that can be kept running in Active-halt and Run modes for the independent watchdog (IWDG). The LSI oscillator can be switched on and off using the LSION bit in the internal clock register (CLK_ICKCR). The LSI clock is an RC oscillator that can vary, due to environment temperature and supply voltage, from 26 kHz to 56 kHz. This is the reason why a measurement of its real value is required to avoid an unexpected behavior due to LSI frequency variation. In the STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers, an internal connection between Timer 2 channel 1 and the LSI clock is provided to simplify the LSI measurement procedure, as shown in *Figure 10*.

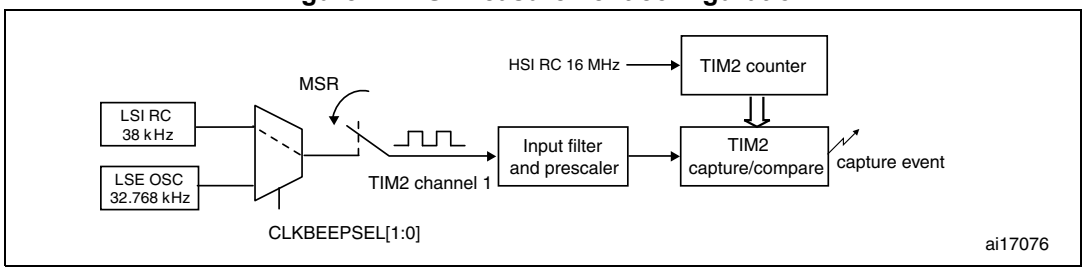
Figure 10. Internal connection between LSI and Timer 2 channel 1



3.1 LSI measurement principle

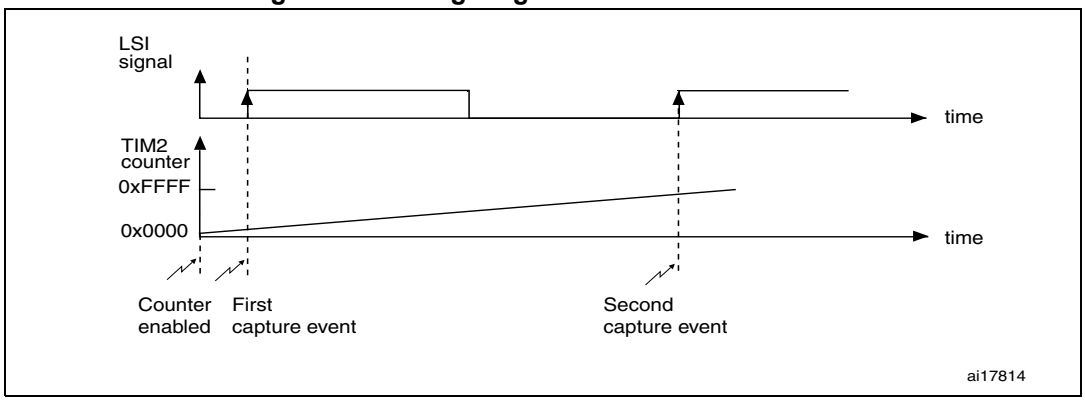
The low-speed internal oscillator (LSI) measurement procedure consists in running the TIM2 counter using the HSI clock, configuring the Timer 2 channel 1 in input capture mode and then connecting the LSI clock to Timer 2 channel 1. *Figure 11* shows the configuration used to perform the LSI measurement.

Figure 11. LSI measurement configuration



After enabling the timer counter, when the first rising edge occurs, the timer counter value is registered in IC1ReadValue1. On the second rising edge, the timer counter is captured in IC1ReadValue2. The elapsed time between two consecutive rising edges of the LSI clock represents an entire period. *Figure 12* shows the timing diagram of LSI measurement.

Figure 12. Timing diagram of LSI measurement



The LSI frequency value is computed as shown by the following formula:

LSI_Frequency = HSI_Value / Capture

where:

- HSI_Value is the HSI frequency value (typically 16 MHz);
- Capture represents an entire LSI period: IC1ReadValue2 - IC1ReadValue1.

As you can conclude from the formula above, the LSI measurement accuracy depends on the HSI frequency accuracy. Consequently, if an external reference signal is available, the user can run the HSI calibration routine described in *Section 2* before performing the LSI measurement procedure in order to get a more accurate HSI_Value.

The Input capture divider (or prescaler) can be used for better measurement accuracy. In this case, the formula above becomes:

LSI_Frequency = InputCaptureDivider * HSI_Value / Capture_Value.

3.2 Description of the LSI clock measurement firmware

The LSI clock measurement firmware provided with this application note includes one major function: *LSI_FreqMeasure(void)*.

The *LSI_FreqMeasure()* function measures the LSI frequency value. It measures a predefined number of LSI periods. Then it returns the average value to minimize the error of measured frequency. The user can change this parameter (number of LSI periods) in the lsi_measurement.h file.

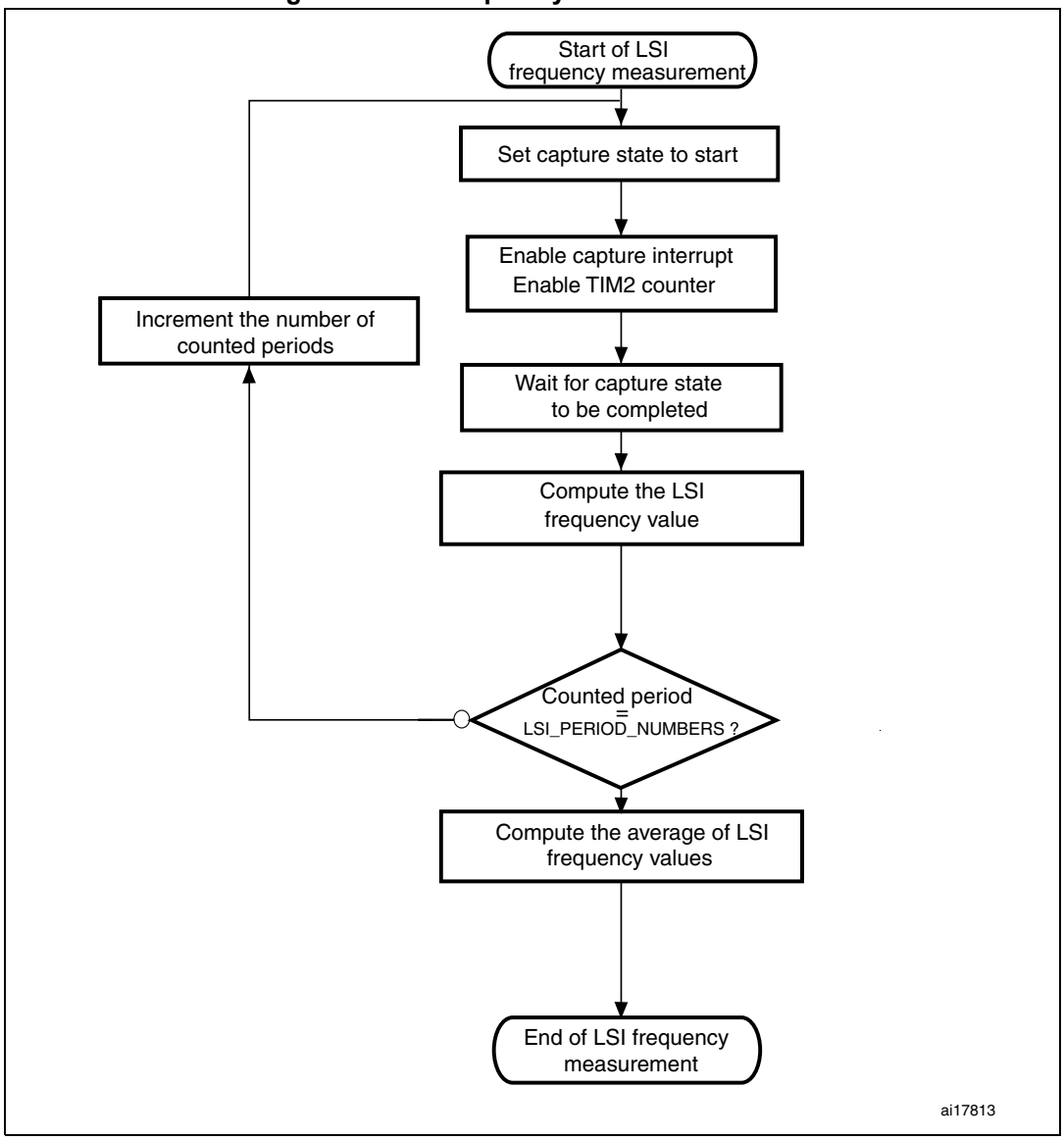
```
#define LSI_PERIOD_NUMBERS 10
```

By default, the number of measured periods per frequency is set to 10.

To start the LSI measurement, save the user clock configuration and set the system configuration for LSI measurement (select the HSI clock as system clock, configure Timer 2 channel 1 in input capture mode, connect LSI to Timer 2 channel 1...).

After system configuration, Timer 2 counter and capture 1 interrupt are enabled. This makes it possible to save the captured values for the predefined number of LSI periods. Once the measurement is performed, the average value is computed and returned after restoring the system clock configuration. *Figure 13* gives the flowchart of the LSI measurement.

Figure 13. LSI frequency measurement flow



3.3 LSI measurement demo description

The demo provided with this application note shows the importance of measuring the LSI frequency value before starting to use peripherals clocked by this type of clock. In this example the independent watchdog (IWDG) is used to generate a reset after 12 ms.

Since the LSI clock is not accurate, the reset will occur slightly after or before the required time-out. Hence, measuring the LSI frequency enhances the time-out accuracy.

4 Conclusions

Even if internal RC oscillators are factory calibrated, the user should calibrate them in the operating environment if a high clock accuracy is required in the application.

This application note gives two routines:

- one for the high speed internal clock calibration: How to tune the HSI clock to 16 MHz?
- one for the low speed internal clock measurement: How to get the exact LSI frequency value?

Several frequency sources can be used to calibrate the HSI oscillator: the LSE crystal, the AC line, etc. Whatever the reference frequency source, the HSI calibration principle is the same: a reference signal must be provided to be measured by a timer. The higher the accuracy of the reference signal frequency, the better the accuracy of the HSI frequency measurement. The error is computed as the absolute value of the typical HSI frequency value and the measured one for each HSITRIMR register configuration. From this, the calibration value is calculated and then programmed in the HSITRIMR register.

The second section of this application note focuses on the LSI clock measurement. The internal connection between the LSI clock and channel 1 of Timer 2 provided in STM8L05xxx/15xxx, STM8L162xx and STM8AL31xx/3Lxx microcontrollers is used to get the LSI frequency value. The Timer 2 is clocked using the HSI clock and Timer 2 channel 1 is configured in input capture mode. So the time between two consecutive rising edges of LSI clock represents an entire period.

5 Revision history

Table 2. Document revision history

Date	Revision	Changes
22-Jan-2010	1	Initial release.
08-Sep-2010	2	Updated *Section 2.3.2: HSI calibration with fixed error*. Updated *Section 2.3.4: HSI calibration demo description*.
11-Feb-2011	3	Modified first page (references to temperature and voltage) Modified *Section 2: High speed internal oscillator calibration* Updated *Section 2.3.1: HSI calibration with minimum error*, *Section 2.3.2: HSI calibration with fixed error* and *Section 2.5.1: Duration of the calibration process*
20-Sep-2012	4	Document updated to include STM8L05xx devices. Added: *Table 1: Applicable products*.
22-Nov-2012	5	Document updated to include STM8AL31xx and STM8AL3Lxx devices.
20-Jan-2015	6	Updated *Section 2: High speed internal oscillator calibration*, *Section 2.3.1: HSI calibration with minimum error*, *Section 2.3.2: HSI calibration with fixed error* and *Section 2.5.1: Duration of the calibration process*. Updated *Figure 1: Example of HSI oscillator trimming characteristics*, *Figure 6: HSI calibration flow: finding the minimum frequency error*, *Figure 7: "Spring loop"* , *Figure 8: HSI calibration flow: maximum allowed frequency error* and *Figure 9: HSI frequency measurement flow*.