**Freescale Semiconductor**
Application Note

Document Number: AN3756
Rev. 0, 10/2008

# Using and Synchronizing the S08's Internal Clock for LIN Slave Implementations

## (for LIN Rev 2.x implementations)

by: Yves BRIANT
Global Sales and Marketing (GSM)

Reliable communication via the asynchronous LIN protocol requires an MCU with a bus clock accurate enough to avoid errors. MCUs that use clocks based on crystal or ceramic resonators easily provide very accurate bus clocks. The LIN protocol was designed to also allow more cost-effective solutions: MCUs with on-chip oscillators can be used successfully to implement LIN slaves, even though the on-chip oscillators have less accuracy than a crystal.

A LIN master initiates each frame by sending a 13-bit break field and a synchronization byte. The data of the synchronization byte is always 0x55. Thus this byte is composed of five falling (recessive to dominant) edges that can be used as a reference for clock and/or baud rate adjustment.

Each derivative of the S08 family embeds an internal oscillator, whose output frequency can be changed by adjusting a trim register. This application note proposes a method for adjusting the frequency of Freescale's S08 internal oscillator by measuring the synchronization byte sent within a LIN frame.

**Contents**





freescale semiconductor

# 1 Baudrate Requirements for LIN

The required clock accuracy is different for a LIN slave and a LIN master. These requirements are summarized in Table 1.

**Table 1. Bit Rate Tolerances Relative to Nominal Bit Rate**

| No. | Bit Rate Tolerance | Name | $\Delta F/F_{Nom}$ |
|---|---|---|---|
| Param 1 | Master node (deviation from nominal bit rate) | $F_{TOL\_RES\_MASTER}$ | <±0.5% |
| Param 2 | Slave node without making use of synchronization (deviation from nominal rate) | $F_{TOL\_RES\_SLAVE}$ | <±1.5% |
| Param 3 | Deviation of slave node bit rate from the nominal bit rate before synchronization; relevant for nodes making use of synchronization and direct break detection | $F_{TOL\_UNSYNC}$ | <±14% |

Since a LIN master initiates each LIN frame and sends the break and the synch fields, it should embed an accurate clock source (<0.5%).

For a LIN slave, two different accuracies are required: < ±14% for the reception of the break and synchronization fields, <1.5% for the reception or emission of the remainder of the LIN frame. These two tolerances are illustrated in Figure 1.

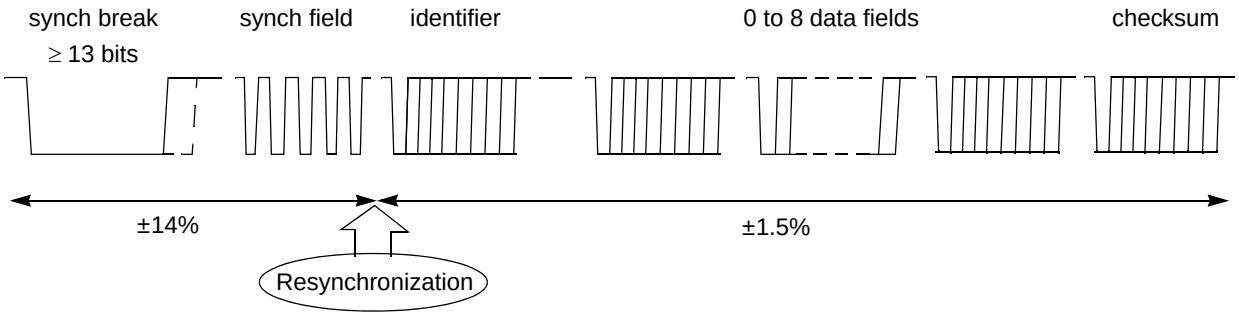


**Figure 1. Baudrate Tolerances for a LIN Slave**

The requirement to have a ±14% accurate clock source, to reliably recognize the break signal, is met by the factory trimming of the oscillator on all of the S08 devices. To achieve the more precise ±1.5% requirement for LIN communication, a further adjustment is required. These adjustments are covered in the next two sections.

# 2 S08's Internal Clock Sources

Three different clock modules exist within the S08 family:

- MCG (Multi-Purpose Clock Generator): for example, embedded in the S08Dx and in the S08EN
- ICS (Internal Clock Source): for example, embedded in the S08EL, SL, SG, QD, QE and QG

- ICG (Internal Clock Generator): for example, embedded in the S08AW and the S08AC

While these clock modules integrate such functions as a PLL (phase locked loop), an FLL (frequency locked loop), a clock monitor, and an external oscillator, this application note focuses only on the internal reference clock. For more details about these clock modules, refer to AN3499: "*Clock Options on the HC9S08 Family*", available at http://www.freescale.com.

The internal reference clock of these three modules behaves the same way: their output frequencies vary with the temperature, supply voltage, and process. The frequency variations can be cancelled by adjusting a "trim register". This trim register can change the output frequency by ±25%, with a resolution finer than 0.4%.

The largest contributor to this frequency deviation is the process variation (±20%). Since this deviation is fixed and does not vary with the time, it can be entirely cancelled by one single initial (in-factory) trim operation.

The frequency deviation due to temperature is guaranteed to be less than ±2% over -40°C to 125°C, whereas the effect of the supply voltage variation can almost be negligible, as long as the supply voltage remains within 5% of the nominal voltage. These two parameters (temperature and supply voltage) vary with the time and their variations should be compensated to assure that the internal clock frequency remains stable.

All the internal clocks on S08 family devices are factory trimmed: a value of the trim register is determined and recorded into flash memory. By copying this trim value into the trim register, the frequency deviation due to the process variation can be almost entirely eliminated. Taking into account the temperature, the supply voltage, and the remaining process variations, the overall accuracy of the factory trimmed internal reference is better than ±14%, which means that the break and synch fields of the incoming frame can be received correctly without any initial synchronization.

Table 2 summarizes the main characteristics of S08's internal clock modules.

**Table 2. Main Characteristics of the S08's Internal Clock**

| | Frequency Range (untrimmed) | Frequency Range (factory trimmed) | Frequency Deviation with Temperature and Supply | Trim Register | Trim Resolution 8 bits | Trim Resolution 9 bits |
|---|---|---|---|---|---|---|
| **MCG** | [25 ; 41.66] kHz = 33.33 kHz ± 25% | [31.25 ; 39.0625] = 35.15 kHz ± 11% | Max ± 2% | 9-bit | Max: ± 0.4%<br>Typ: ± 0.2% | Max: ± 0.2%<br>Typ: ± 0.1% |
| **ICS** | [25 ; 41.66] kHz = 33.33 kHz ± 25% | [31.25 ; 39.0625] = 35.15 kHz ±11% | Max ± 2% | 9-bit | Max: ± 0.4%<br>Typ: ± 0.2% | Max: ± 0.2%<br>Typ: ± 0.1% |
| **ICG** | [182.25 ; 303.75] kHz = 243 kHz ± 25% | 250 kHz ±11% or 243 kHz ±11% | Max ± 2% | 8-bit | Max: ± 0.4% | NA |

Temperature range: [-40; + 125]°C;

Supply range: 5V± 10%;

All parameters minimum and maximum values. Values are for indication only — refer to the device data sheets.

# 3 Implementation of the Internal Clock Synchronization

The implementation described in the following section is predominantely the same for all S08 derivatives. Some details differ slightly depending on the embedded clock module (ICS, ICG, or MCG).

## 3.1 Principle of the Synchronization

The basic principle of the synchronization method is as follows:

- The synch field, sent by the LIN master, is used as a fixed reference clock (in fact, a ±0.5% accurate clock signal). The duration of this synch field can be measured by any LIN slave, using a timer channel.
- Since the timer channel is clocked by the internal oscillator of the slave, the measured duration of the synch field illustrates the clock's deviation from the internal oscillator's expected frequency. If the measured duration is shorter than the expected one (assuming the nominal frequency of the internal oscillator), it means that the internal oscillator is running too fast, and vice-versa.
- From this measured duration, a new trim value can be computed and loaded into the trim register of the internal oscillator. If the internal clock oscillates too fast, the trim value should be increased, and vice-versa.

Once the process variations have been cancelled out by the initial trim, the major contributor to the internal clock deviation is the temperature. Since temperature is a slowly changing parameter, the system may be able to cope with re-synchronizing the clock less often to save CPU bandwidth; however, this strategy must be assessed against the improbable but possible effect of communication failure.

Figure 2 shows how the clock re-synchronization is achieved.

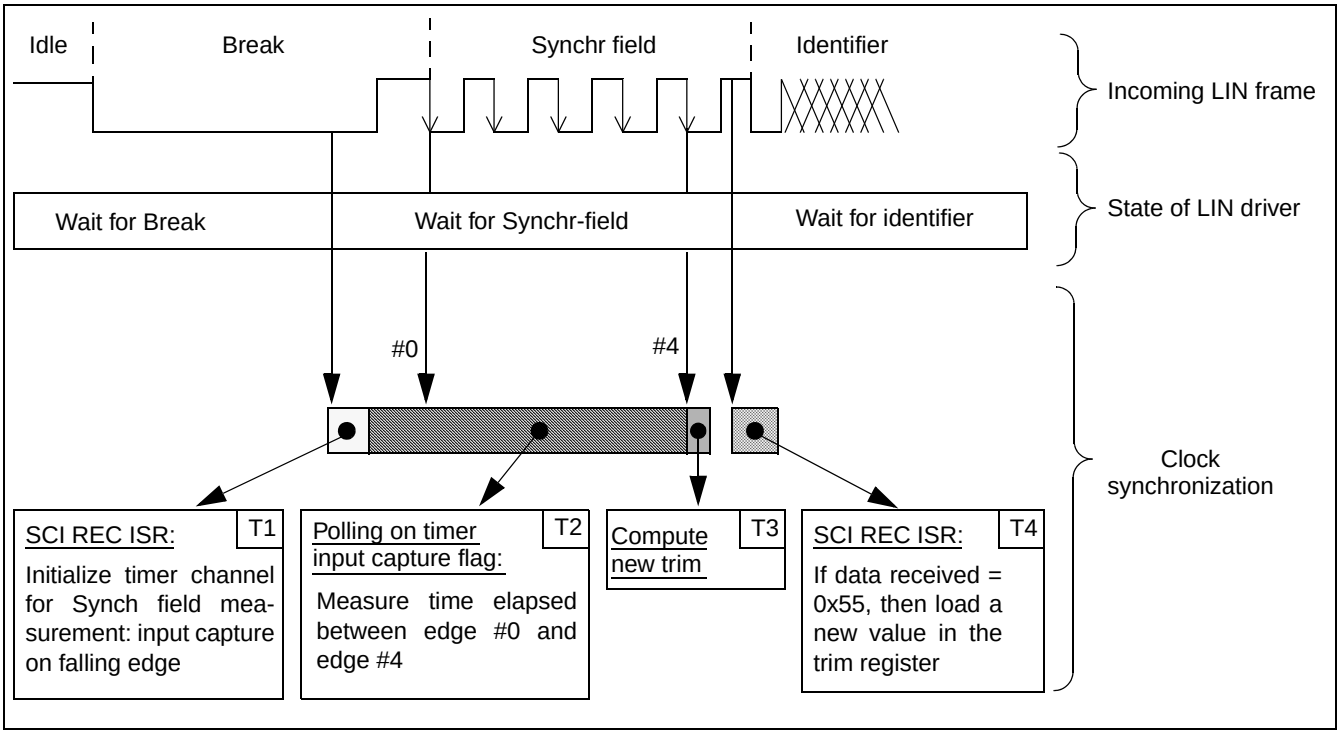


**Figure 2. Principle of the Clock Synchronization**

The measurement of the synch field duration is achieved by either polling the input capture flag or by using a timer input capture interrupt routine. For both methods, all interrupt sources should be disabled (except the timer interrupt if the polling method is not used), to allow the most precise measurement.

## 3.2 Correction of the Internal Clock

### 3.2.1 Computation of the New Trim Value

The duration $T_{synch}$ of the synch field is measured between the first and the fifth falling edges, corresponding to 8 bits. Therefore:

$$T_{synch} = 8Tbits = \frac{8}{LIN\_baudrate} \qquad \textit{Eqn. 1}$$

where *LIN_baudrate* is the nominal baudrate on the LIN network, assuming no clock deviation.

With the SCI module embedded in the S08, the baudrate is generated from the bus frequency, according to the formula in Equation 2,

$$LIN\_baudrate = \frac{F_{bus}^{nom}}{16 \times LIN\_prescaler} \qquad \textit{Eqn. 2}$$

where $F_{bus}^{nom}$ is the nominal bus frequency and *LIN_prescaler* is the 16-bit content of the SCIBDL and SCIBDH registers.

It yields from Equation 1 that:

$$T_{synch} = \frac{8 \times 16 \times LIN\_prescaler}{F_{bus}^{nom}} = \frac{128 \times LIN\_prescaler}{F_{bus}^{nom}} \quad \textbf{Eqn. 3}$$

The measurement of $T_{synch}$ with the S08 timer (TPM) results in the digital value $V_{meas}$, which has following expression.

$$V_{meas} = \frac{T_{synch}}{T_{timer}} \quad \textbf{Eqn. 4}$$

where $T_{timer}$ is the period of the clock source of the timer.

The timer should be clocked by the bus frequency, fed by the internal oscillator. With $F_{bus}^{act}$ referring to the "actual bus frequency", it yields:

$$T_{timer} = \frac{timer\_prescaler}{F_{bus}^{act}} \quad \textbf{Eqn. 5}$$

From Equation 3, Equation 4 and Equation 5:

$$V_{meas} = \frac{T_{synch}}{T_{timer}} = \frac{128 \times LIN\_prescaler}{F_{bus}^{nom}} \times \frac{F_{bus}^{act}}{timer\_prescaler} = \frac{F_{bus}^{act}}{F_{bus}^{nom}} \times \frac{128 \times LIN\_prescaler}{timer\_prescaler}$$

By defining:

$$\frac{128 \times LIN\_prescaler}{timer\_prescaler}$$

as the constant *F_DEV_FACTOR*, the internal oscillator deviation can be calculated directly from $V_{meas}$:

$$\frac{F_{bus}^{act}}{F_{bus}^{nom}} = \frac{V_{meas}}{F\_DEV\_FACTOR} = 1 \pm \Delta \quad \textbf{Eqn. 6}$$

It can be observed from Equation 6 that, if $V_{meas}$ is greater than the constant *F_DEV_FACTOR*, it means that the actual bus frequency is greater than the nominal bus frequency, and that the trim value should be increased.

From Equation 5, we can also get that:

$$\Delta = \left| \frac{V_{meas} - F\_DEV\_FACTOR}{F\_DEV\_FACTOR} \right|$$

The 8-bit trim register of the S08 internal clock sources allows a frequency adjustment with a granularity of 0.4% max (see Table 2). The adjustment of the trim value is thus:

$$\Delta_{trim} = \frac{\Delta}{0.4\%} = 250 \times \Delta = \left| V_{meas} - F\_DEV\_FACTOR \right| \times \frac{250}{F\_DEV\_FACTOR}$$

Defining the constant:

$$F\_CORR\_FACTOR = \frac{250}{F\_DEV\_FACTOR}$$

this means:

$$\Delta_{trim} = \left| V_{meas} - F\_DEV\_FACTOR \right| \times F\_CORR\_FACTOR \qquad \textit{Eqn. 7}$$

**NOTES**

The constants F_DEV_FACTOR and F_CORR_FACTOR are integer constants. This results in an approximation in the clock correction.

The prescaler of the timer should be selected so that the 8-bit counter value may overflow only one time during the measurement of the synch field:

$$256 . T_{timer} > T_{synch} = \frac{8}{LIN\_baudrate} \Leftrightarrow T_{timer} > \frac{1}{32 \times LIN\_baudrate} \Leftrightarrow T_{timer} > \frac{LIN\_prescaler}{2 \times F_{bus}}$$

Since:

$$T_{timer} = \frac{timer\_prescaler}{F_{bus}}$$

this yields Equation 8.

$$timer\_prescaler > \frac{LIN\_prescaler}{2} \qquad \textit{Eqn. 8}$$

### 3.2.2 Algorithm of the Clock Synchronization

The flowcharts in Figure 3 and Figure 4 summarize the contents of Task 3 (“Compute New Trim Value”) and of Task 4 (“Apply New Trim Value”).

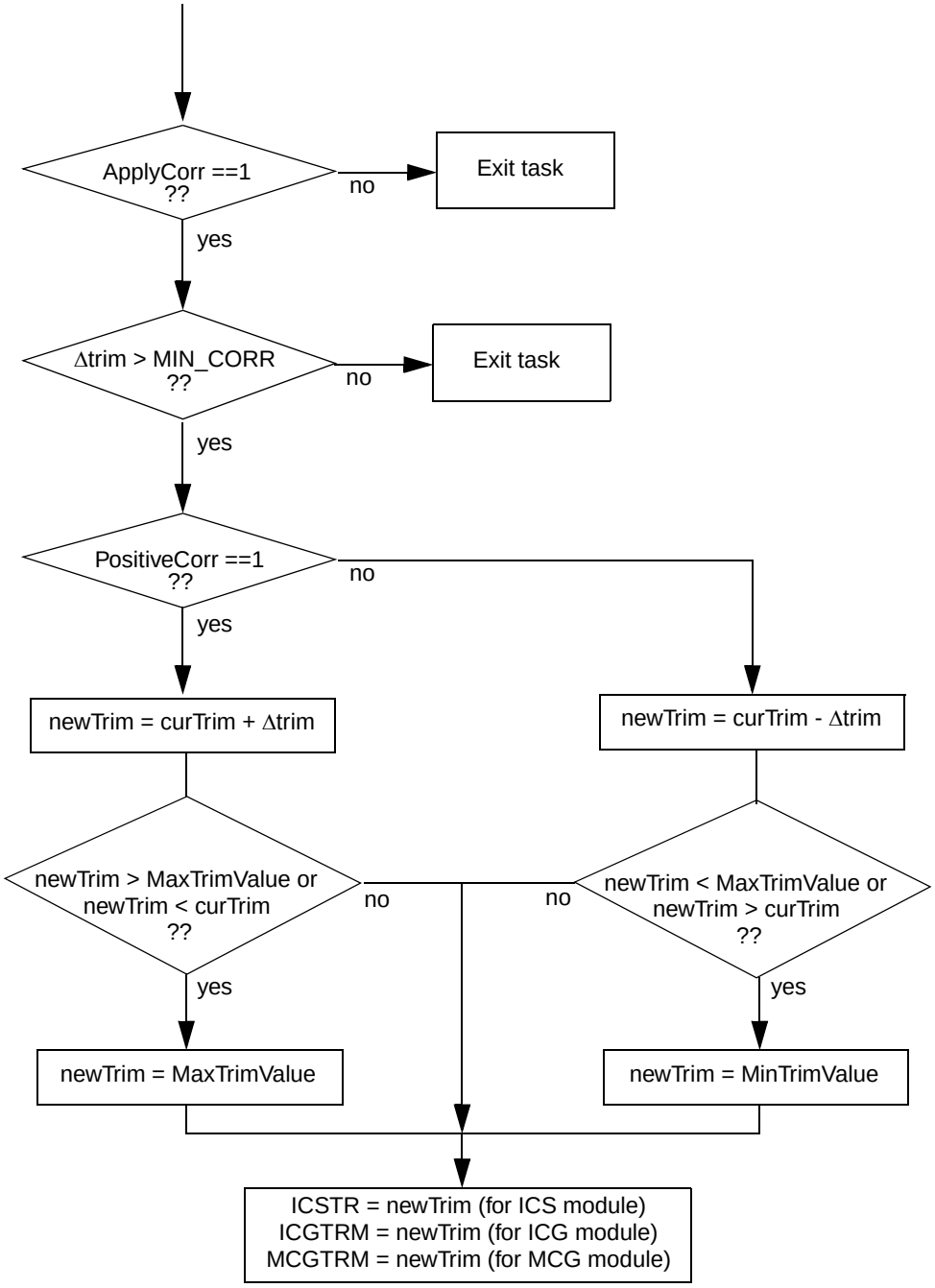


**Figure 3. Flowchart of the "Compute New Trim Value" Task**

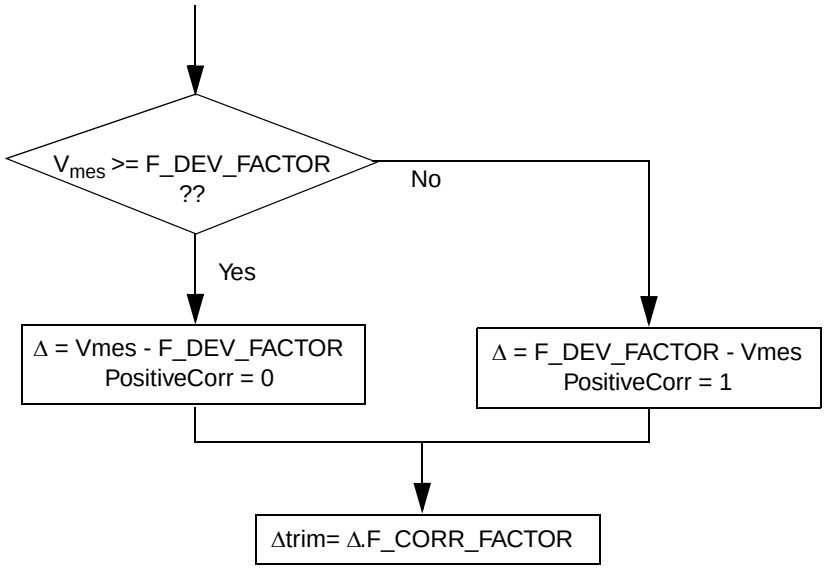


**Figure 4. Flowchart of the "Apply New Trim Value" Task**

*ApplyCorr* is a variable regularly updated (for example by the Real-Time-Counter interrupt) that defines the update rate of the internal clock.

*MIN_CORR* is a constant defining the granularity of the clock synchronization. Any change of the trim value smaller than *MIN_CORR* will not be applied.

*PositiveCorr* is a variable updated by the task "Compute New Trim Value", representing the sign of the trim correction.

*MaxTrimValue* and *MinTrimValue* are constants that define the maximum cumulative correction that could be applied.

### 3.2.3 Numerical Example

Consider the example of a LIN slave implemented with a S08SG containing an ICS module. This slave communicates on a LIN network at a nominal baudrate of 19200 baud, and is clocked at a nominal bus frequency $F_{bus}^{nom}$ = 16 MHz.

The LIN and Timer prescalers can be calculated using Equation 2 and Equation 8:

$$LIN\_prescaler = \frac{16\ \text{MHz}}{16 \times 19200} = 52$$

$$Timer\_prescaler > \frac{LIN\_prescaler}{2} = 26$$

Timer_prescaler = 32 is the best choice on the S08SG.

Therefore:

*F_DEV_FACTOR* = 128 . 52 / 32 = 208

*F_CORR_FACTOR* = 250 / 208 = 1.2 which is approximated to 1.

Consider the situation where the actual bus frequency is not the nominal 16 MHz, but is 15.84 MHz (16 MHz – 1%).

$$V_{meas} = \frac{8}{19200} \times \frac{15{,}84\ MHz}{32} = 206$$

Therefore, according to Equation 7:

$$\Delta_{trim} = (208 - 206) \times 1 = 2$$

The trim value will be increased by 2, which results in a correction of 2 x 0.4% = 0.8%.

## 3.3 Remarks

1. The synchronization process works only if the initial trim value (determined at ambient temperature) is far enough from 0x00 and 0xFF to allow at least a ±2% correction of the temperature. Assuming a 0.2% resolution of the trim register, the initial trim value should be greater than 10 (2%/0.2%) and less than 245 (255-2%/0.2%). The factory-trimmed values programmed by Freescale are within these two limits.
2. The new trim value should be computed in a way that does not lead to over correction. There are several reasons for that:
   - The main cause of the internal clock deviation is the temperature, which varies slowly. Thus a small clock deviation can be corrected by several small adjustments rather than by one unique and big clock correction, that may lead to an overshoot of the clock's frequency.
   - The internal clock feeds (directly or through the FLL) the bus frequency, which clocks the peripherals of the LIN slave. Having a "restrained" clock correction avoids any over-oscillation of the bus frequency.
   - Most of the time, the internal clock is the reference of the FLL, the output of which feeds the bus frequency. If the internal clock correction is too big, there is a small risk that the FLL could lose lock, leading to loss of communication on the LIN network.
3. The resolution of the trim process is 0.4% maximum (see Table 2) but no minimum value is guaranteed. This means that the computed correction (which assumes 0.4% in the example above) may not have the desired impact and be too small to compensate for the frequency deviation. Therefore, two or three correction cycles will be required to allow for the frequency deviation.
4. On several S08 packages, the LIN Rx pin is located next to a timer channel pin (see Figure 5. Using this timer channel eases the routing of the application.

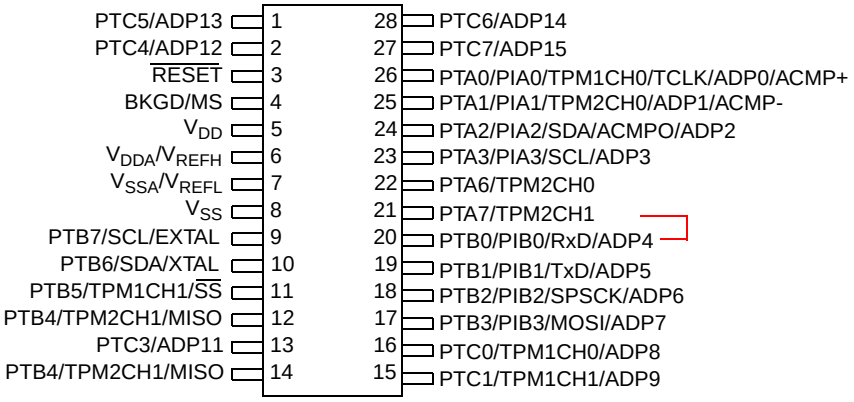


**Figure 5. Use of TPM2CH1 on 28-pin Package to Measure the Synch Field Length**

# 4 Synchronization Performance

The algorithm described in this application note has been implemented and tested on the MC9S08SG32, using the Freescale evaluation board (DEMO9S08SG32AUTO) and CodeWarrior™ for Microcontrollers, Version 6.1. Both the accuracy and the CPU load required by this algorithm have been measured.

## 4.1 Accuracy of the Synchronization

The accuracy of the synchronization has been tested for a deviation in the range -3% to +3% . The nominal baudrate is 9600 baud and the bus frequency is nominally 16 MHz.

The results are shown in Figure 6.

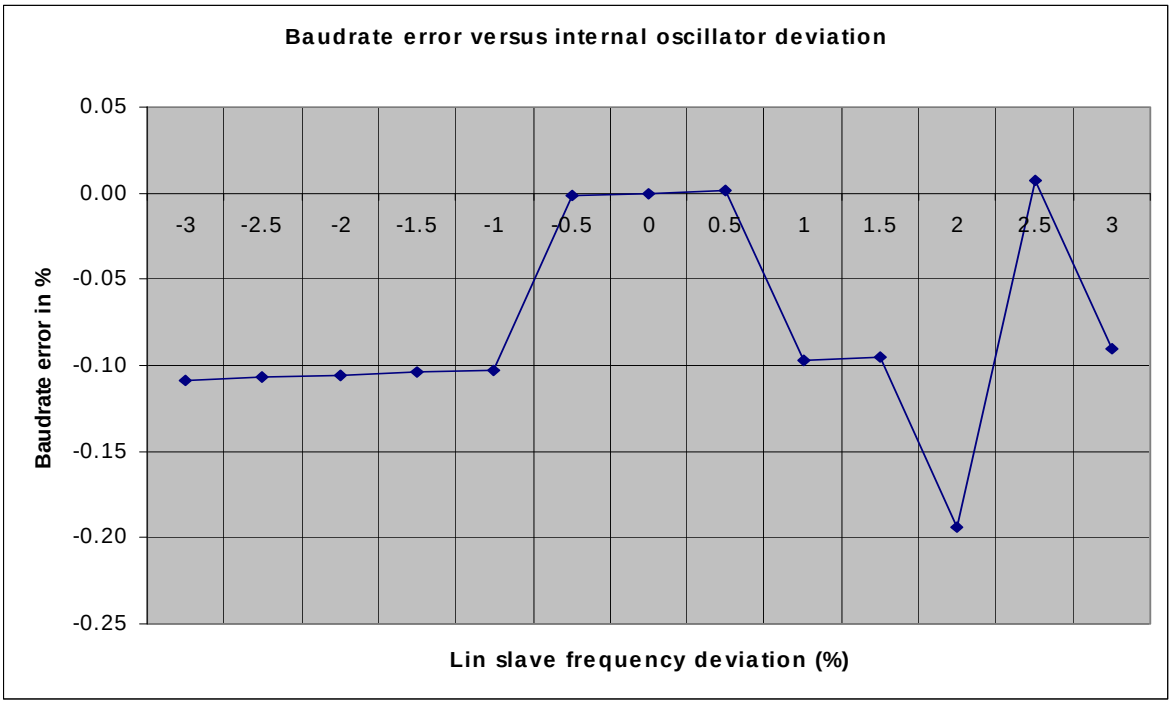


**Figure 6. Accuracy of the Frequency Correction on a S08SG**

Figure 6 shows that the accuracy of the synchronization is always better than 0.2%, which is far better than the LIN communication requirement of 2%.

Several observations can be made on the achieved accuracy:

- The measurement of the synch field with the 8bit timer introduces an error Em:
  Em < 1/(256 x 2) = 0.2%
- The accuracy of one synchronization depends on the height of a trim step (the frequency deviation caused by an increment of the trim value), which is less than or equal to 0.4%. It also depends on the approximation when computing the integer constant *F_DEV_FACTOR* and *F_CORR_FACTOR*. These two factors lead to an underestimation of the correction, which is the recommended behavior (see remark 2 in Section 3.3, “Remarks”).
- This synchronization method is a closed loop control system because the effect of one correction affects the next measured synch field. The accuracy of one correction is not as relevant as other criteria, like the stability and the damping of the response.

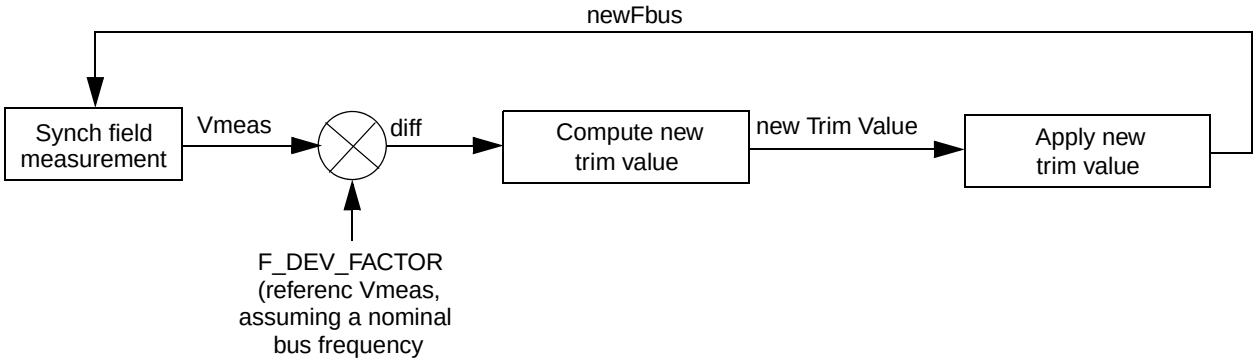


**Figure 7. Synchronization Method shown as a Feedback Control System**

## 4.2 CPU Loading

When the polling method is used (as described in Section 3.1, “Principle of the Synchronization”, the CPU is loaded at 100% during the reception of the first eight bits of the synch field. After this measurement, the computation of the new trim value (see Figure 2) requires 7.4 µsec with the S08 running at 16 MHz.

7.4 µsec should be compared to the duration of the remaining two bits of the synch field (last data bit + stop bit). At 19200 baud, this period (104 µsec) is longer than the time required to compute the correction, so the new trim value is always available at the end of the reception of the synch field (even with a 2 MHz bus frequency, and a 19200 baud LIN network).

## 4.3 Settling Time

The internal clock of the S08 can be selected to feed the FLL, allowing a high bus frequency. When changing the trim register, the reference frequency of the FLL changes, and the FLL may lose its lock, yielding larger frequency deviations until it locks again.

Since the corrections of the trim value are very slight (and can be limited by software) the FLL is not at risk of losing lock. In practice, the FLL will not lose lock with frequency changes less than 2%. Note that the loss of lock can only be monitored, via an interrupt, on devices with an MCG or ICG embedded. Loss of lock on ICS based devices cannot be monitored.

# 5 Conclusions

A ±14% accurate clock source is necessary in order to reliably recognize the break signal; this level of accuracy is achieved by factory trimming of the oscillator on all S08 devices. To achieve the more precise ±1.5% accuracy necessary for LIN communication, further adjustment is required. The method of synchronization described in this application note provides a level of accuracy far beyond the LIN protocol requirements for a slave node, at the expense of some additional CPU loading and one additional timer channel.

# Appendix A Example of Implementation

The CodeWarrior project that serves for the evaluation of this synchronization method is attached to this application note. This project is based on the LINkit for the S08SG32, available at http://www.freescale.com.

To integrate the clock synchronization, the following changes have been made.

- In the function *main()*:
  - After the peripheral initialization, a call to the function *initTimer4LIN()* is added for the configuration of Channel 1 of Timer 2 for the synch field measurement.
  - The function *initTrimSaturation()* is called to compute the maximum and minimum allowable trim value, depending on the maximum amount of clock correction that the user wishes to allow.
- In the interrupt routine *LIN_ISR_SCI_Receive()*:
  - When a break character has been received, the measurement of the synch field is armed. The polling method is used to measure the length of the synch field. After the fifth falling edge, the measurement is stopped, and the value of the clock correction is computed
  - When a synch characacter has been received, the clock correction is applied if the variable *ApplyCorr* is set. In this implementation, *ApplyCorr* is set by RTC interrupt every 10 milliseconds.

**How to Reach Us:**

**Home Page:**
www.freescale.com

**Web Support:**
http://www.freescale.com/support

**USA/Europe or Locations Not Listed:**
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

**Europe, Middle East, and Africa:**
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

**Japan:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3756
Rev. 0
10/2008



freescale™
semiconductor