Freescale Semiconductor
Application Note

Document Number: AN4310
Rev. 0, 07/2011

IRTC Clock Compensation Mechanism in the MCF51EM Family

by: Christian Michel Sendis

Contents

1 Introduction

This document shows the results of a study done to evaluate the compensation mechanism present in the MCF51EM256 Independent Real Time Clock (IRTC) module to achieve a clock signal that is accurate down to a few parts per million of frequency drift across a temperature range of –45 °C to +85 °C.

32,768 KHz crystals, commonly used in time-keeping applications, may deviate from their nominal 32,768 KHz frequency due to manufacturing processes, aging, and temperature. These three factors may contribute to the crystal presenting deviations that range from the tenths of parts per million (PPM) up to hundreds of PPM.

The IRTC module inside the Freescale MCF51EM256 features a clock compensation mechanism that is designed to compensate the effect of these deviations. It allows adjusting frequency drifts from as low as 0.119 PPM up to 3906 PPM.

1.1 Compensation mechanism inside the MCF51EM256

The MCF51EM256's IRTC module provides the application with several interrupt signals that can be used to keep track of time. These interrupts are triggered periodically and have frequencies with values of 1 Hz, 2 Hz, 4 Hz, 8 Hz, and so on,

freescale semiconductor

up to 512 Hz. Of all these interrupts, the 1 Hz interrupt deserves a special mention because it is the only one that can be compensated and is the basis for the rest of the date and time functions present in the IRTC module. This is the signal that drives the increment of the IRTC's time registers.

The compensation mechanism present in the IRTC module allows compensating this 1 Hz signal. The mechanism is simple. The 32768 KHz crystal increments an internal hardware counter with each oscillation of the crystal. When 32768 oscillations have been counted, the IRTC module issues the 1 Hz interrupt, updates the time registers, and resets the counter back to zeroes. If the crystal is oscillating at exactly 32768 Hz, then this interrupt has a period of exactly one second. If not, then the compensation mechanism allows to subtract or to add a number of oscillations beyond the 32768 theoretical value so that the 1 Hz interrupt triggers before or after the counter reaches the 32768 counts, depending on whether the crystal is too slow or too fast.

These extra oscillations can be programmed to be added every second, or on every multiple of a second, up to 255 seconds. This interval of time is called the compensation interval. The number of oscillations to add or subtract is a signed integer number and can range from –127 to +127. This is called the compensation value.

A programmer can play with different combinations of compensation intervals and compensation values to achieve the exact number of PPMs wanted to compensate for.

Figure 1. Illustration of the compensation mechanism

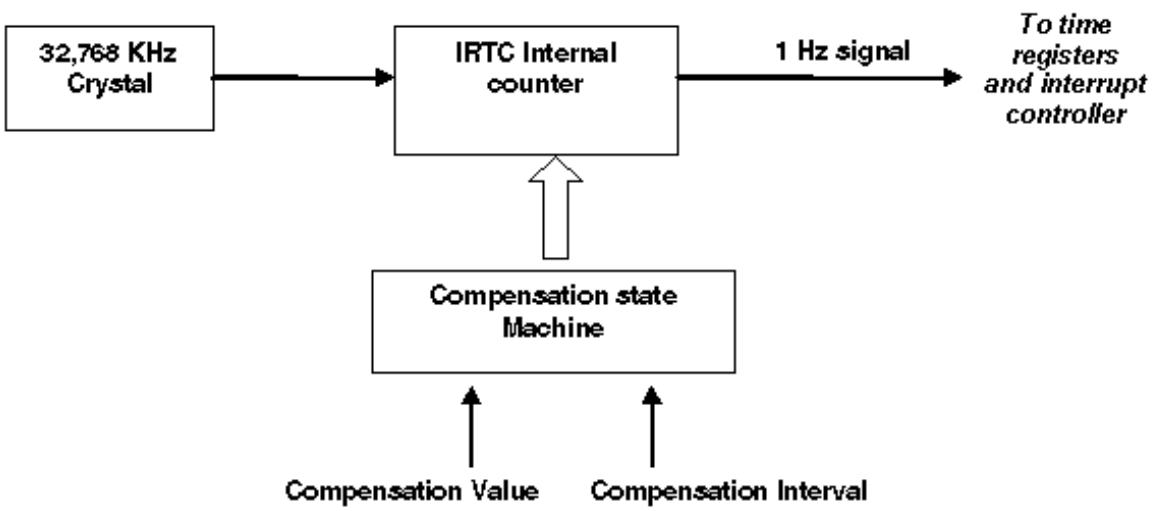
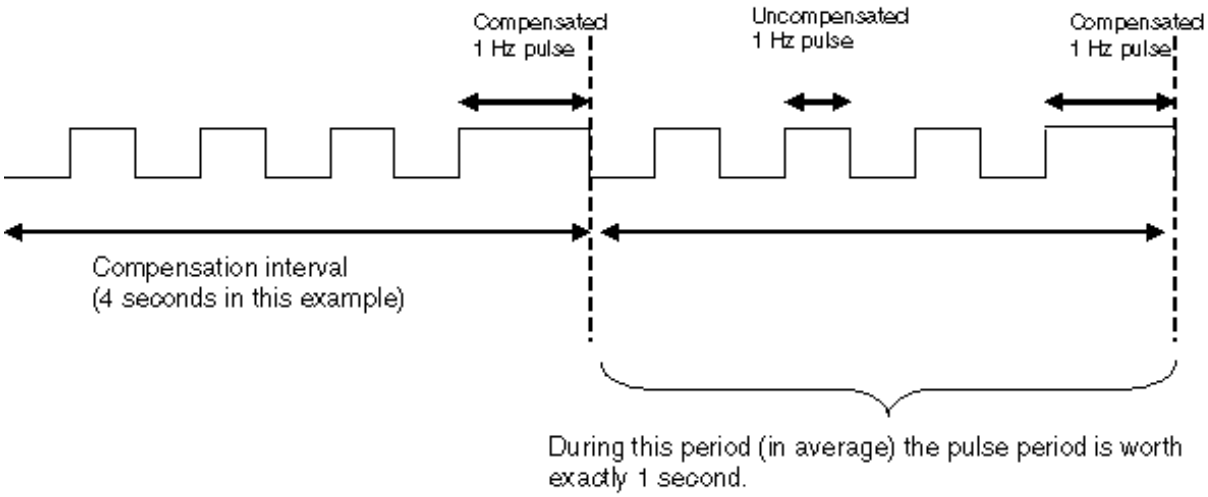


Figure 2. Compensated 1 Hz signal issued by the IRTC module



N, the number of clock cycles to add or substract to compensate for D, a given deviation in PPMs, with a given compensation interval value I, is given by:

1.2 Crystal behavior across temperature

Crystal manufacturers typically design a crystal that behaves the closest to its nominal frequency at room temperature (25 °C). Crystal manufacturers provide a table or curve that describes the behavior in frequency deviation of the crystal across a temperature range.

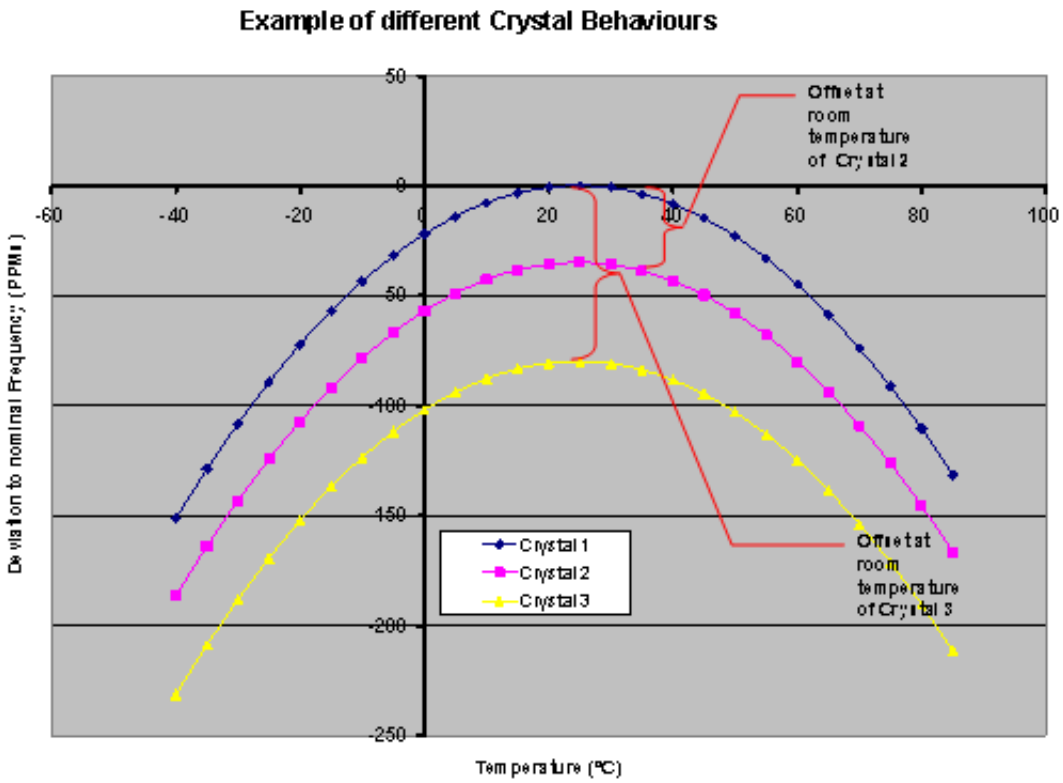
This deviation because of temperature will add to a “static” deviation that the crystal may also present, due to manufacturing processes and aging.

The following figure illustrates the hypothetical behavior of three different crystals from the same hypothetical manufacturer, that is the same part number against different temperatures.

In one, the first approach, consider that the frequency deviation of the crystal behaves as a parabolic function of temperature. Identical part numbers may show different offsets at room temperature, this is due to the manufacturing processes and the age of the part. This offset at room temperature defines the position of the curve in the graph. Besides this offset, the rest of the crystal behaviors remain very similar between crystals. All these curves can in one first simplifying approach be considered to be the same curve with only different offsets*.

A simplified approach to how the oscillation frequency of a crystal behaves when the temperature changes. This figure illustrates the hypothetical drift in frequency that 3 hypothetical crystals from the same crystal model may exhibit.

Figure 3. Different crystal behaviors



NOTE

In reality, each crystal presents a curve that has its own parabolic shape, and the differences between crystals from the same model go beyond just the " offset at 25 ° C " value. The differences in each crystal behavior strongly emerges as temperatures go to the extremes where the parabolic branches have more important slopes.

2 Development of an Algorithm to Compensate for Temperature Changes

2.1 Compensation algorithm behavior

Obtaining a table that describes the behavior of the crystal model is at the core of the compensation algorithm presented here. The table lists the expected frequency deviations for a given temperature. This table can either be obtained from the crystal manufacturer or be created through measurements in the lab. In the next section, notice how this was done in the case presented here.

After the table is obtained, formula 1.1 can help transform this " Deviation in PPM versus Temperature " table into a " Compensation register value versus Temperature " table.

As a reminder the compensation register contains two values, compensation interval and compensation value. For simplicity and easy implementation maintain a constant compensation interval. In this case, the value selected is ten seconds.

This means that the generation of a compensated 1 Hz pulse (refer to figure 1.2) will always occur every ten seconds. With this compensation value fixed, a table is constructed with "compensation values versus temperature".

This table is saved in the microcontroller flash memory to be referenced by the algorithm.

The algorithm is simple. It measures the ambient temperature using an external temperature sensor. Temperature is measured once per second, and every compensation interval seconds, the average temperature over the last compensation interval is computed. Based on this average temperature value, the corresponding compensation value is extracted from the compensation table and written into the compensation register.

These tests were made using an external temperature sensor from analog devices, part number TMP36GRTZ. In the software that accompanies this document this external sensor is expected to be connected to AD channel number 5, which in the DEMOEM256 board is easily available for connection in jumper J38.

In the example shown here, the compensation value is re-evaluated at the end of every compensation interval. This does not always need to be the case. An application can work with a certain compensation interval (for example ten seconds) and reevaluate the compensation value every minute.

The reevaluation of the compensation value should be executed at a pace that makes sense for the context of the application. This pace is the pace at which temperature is likely to change.

The compensation interval needs to be chosen depending on the deviation values that one wishes to compensate for.

For example, with a one second compensation interval, the smallest fraction that can be compensated is when one clock cycle every second is subtracted, and the biggest fraction is when 127 clock cycles every second is subtracted, (one clock cycle being the inverse of 32768 Hz). These choices yield the following deviation-correction values: 30.5 PPMs up to 3875 PPMs, in steps of 30.5 PPMs.

If instead of substracting clock cycles every second and instead substract them every ten seconds, then the possible correction values change by a factor of ten. This means that with an interval of ten seconds, deviations ranging from 3.05 PPMs up to 387.5 PPMs in steps of 3.05 PPMs can be compensated.

Choosing a smaller compensation interval allows to compensate for bigger values of deviation, but with a bigger step resolution can be lost concerning the values that can exactly be compensated for.

Choosing a bigger compensation interval increases resolution, but does not allow compensating for very big values of deviation. In these tests, the crystals used to test this algorithm were exhibiting deviations that could reach values around 300 PPMs at high temperatures. Therefore a compensation interval of 10 seconds as a good compromise between maximum values and acceptable resolution was used.

This means that every 10 seconds, the algorithm produces a 1 Hz pulse that is shorter or longer than the uncompensated pulse. Nine pulses will be uncompensated, one pulse will be compensated, and in in an ideal case the total of these ten pulses will sum to give an average time of exactly1 second.

For simplicity, the algorithm reevaluates the compensation value every ten seconds, even if this may be too frequent for monitoring temperature that changes only because of the weather.

To achieve the most accurate compensation, the procedure requires each crystal to be calibrated individually. This calibration is the measurement of the crystal’s deviation at room temperature, mentioned previously. The measurement of this offset at room temperature is used to adjust the compensation table to the offset shown by that particular crystal. Later in this document a method for measuring this by firmware is described.

3 Implementation Details, Testing, and Results

In this section are the steps that were followed to evaluate the performance of the compensation algorithm inside a temperature range of –40°C up to +85°C.

NOTE

The present study was carried on a particular Crystal model which is not the model that comes as standard crystal in the DEMOEM256 Freescale boards. To adjust the algorithm to the crystal present in the DEMOEM256 board, the compensation table needs to be created for that particular model. You may use this section as guidance on how to do this.

3.1 Obtaining the compensation table

In this application note we decided to directly measure the deviation shown by a crystal in function of temperature and build our own compensation table. The steps described here can be repeated when obtaining a new table for a new crystal model.

3.1.1 Measurement of the Deviation Versus Temperature Curve

For this study, the behavior of 3 different crystals (same part number) was measured and computed an average behavior that represents the crystal model. For each crystal, the frequency deviation was measured at different temperatures.

The IRTC 1 Hz interrupt flag to toggle a GPIO pin from the microcontroller every second was used. This 1 Hz interrupt is directly related to the crystal's oscillating frequency. Toggling of this GPIO every 1 Hz produces a square signal whose periodicity is 2 seconds. This signal is fed to a universal counter that provides the period of the signal down to the nanosecond. It also provides statistics such as an average over many samples and standard deviation of the set of measurements. This was verified by looking at the standard deviation of the period of the signal, that toggling the pin inside the IRTC interrupt does not introduce a jitter in the signal caused by different interrupt servicing latencies. The standard deviation of all measurements indicated that all periods were identical down to less than a tenth of a microsecond.

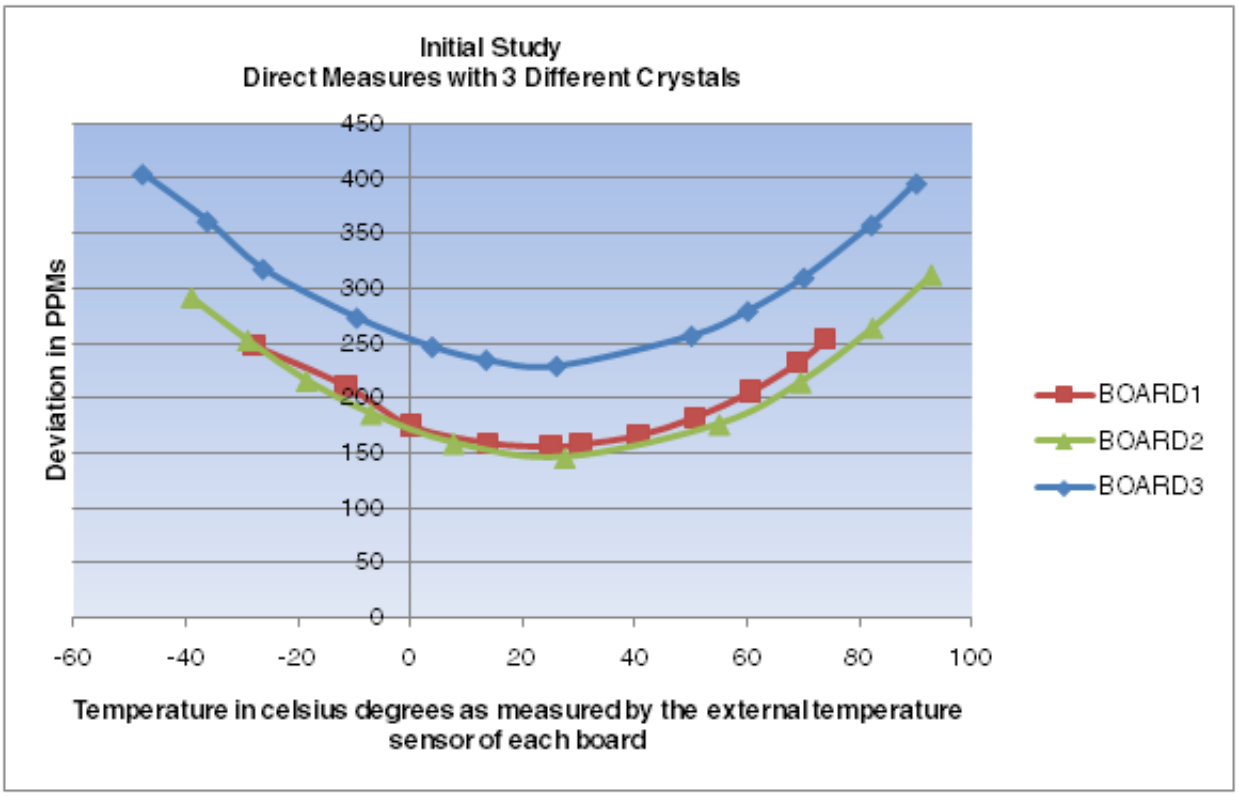
In the software accompanying this document the specific pin being toggled is PTA5, in the DEMOEM256 Board it is easily accessible through jumper J41.

If the crystal is exactly synchronized to the time base of the universal counter, this period will be exactly equal to 2 seconds. If it is not, you will observe a different value. The number of microseconds above or below the 2 seconds value gives us the deviation of the signal in parts per million.

For example, if the signal measured is 2.000320 seconds, this means this signal is 320 microseconds too long. Therefore every 2 seconds the signal shows a deviation of 320 millionths of a second, or, 320 parts per million. This means every 1 second the signal shows a deviation of 320/2 = 160 parts per million. 160 parts per million is then recorded as the deviation of the crystal at the temperature at which this measurement was made.

By following this process and placing the MCF51EM256 inside a temperature chamber (Tenney-TempGard IV model), the deviation of each of the 3 crystals at several temperatures was measured.

Figure 4. Initial study



NOTE

The frequency offset at +25 °C shown in this figure is large. Though the described method is able to compensate this strong frequency offset, we note that large offsets may be caused by a mismatch of the nominal load capacitor of the tuning fork crystal versus the effective load capacitance in the actual circuit. By an appropriate matching of the load capacitors, the frequency offset can be reduced from more than 150 ppm down to about +/–30 ppm, using standard crystals with a make tolerance of +/–20ppm at +25 °C. The study here was done in the context of an external design with a fixed bill of materials and it was decided not to modify the values of the load capacitance in the circuit.

3.1.2 Obtaining an average curve for all temperatures

Obtain an average curve that represents the crystal model and provides numeric values for all temperatures. For this, it was decided to use a trendline estimation computation to approximate these three curves to three polynomials whose equation we know. Once the equation is known, the equation is used to get values even for data points that were not directly measured.

The temperature range that is of interest is, -40°C to +85°C.

To be able to accurately approximate these measurements to a polynomial a polynomial of 4th order was used. An initial study showed it was difficult to match all these measured results with little error, if the description of each crystal's curve to a 2nd order polynomial is limited.

By using trendline computations in Excell, (the details on how to do this can be found in the appendix) a polynomial of the form was obtained for each crystal.

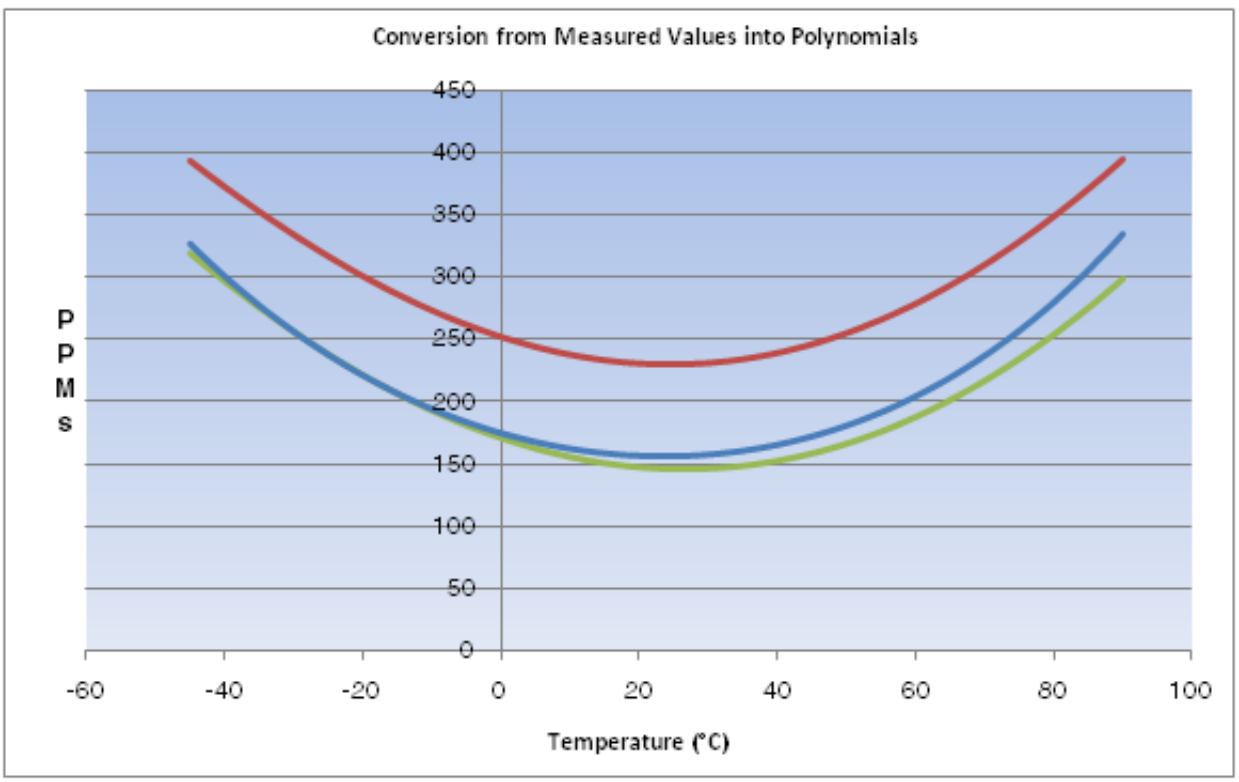
Deviation (temp) = C4* temp^4 + C3*temp^3 + C2*temp^2 + C1*temp + C0

The following table summarizes the values found for the coefficients that describe the polynomials of each crystal:

	BOARD1	BOARD2	BOARD3
C4	1.024E-06	-9.713E-08	-4.146E-07
C3	-6.335E-05	3.968E-05	7.258E-05
C2	0.0348651	0.0335794	0.0343092
C1	-1.5919723	-1.8581479	-1.7858503
C0	174.8454	171.21547	252.25097

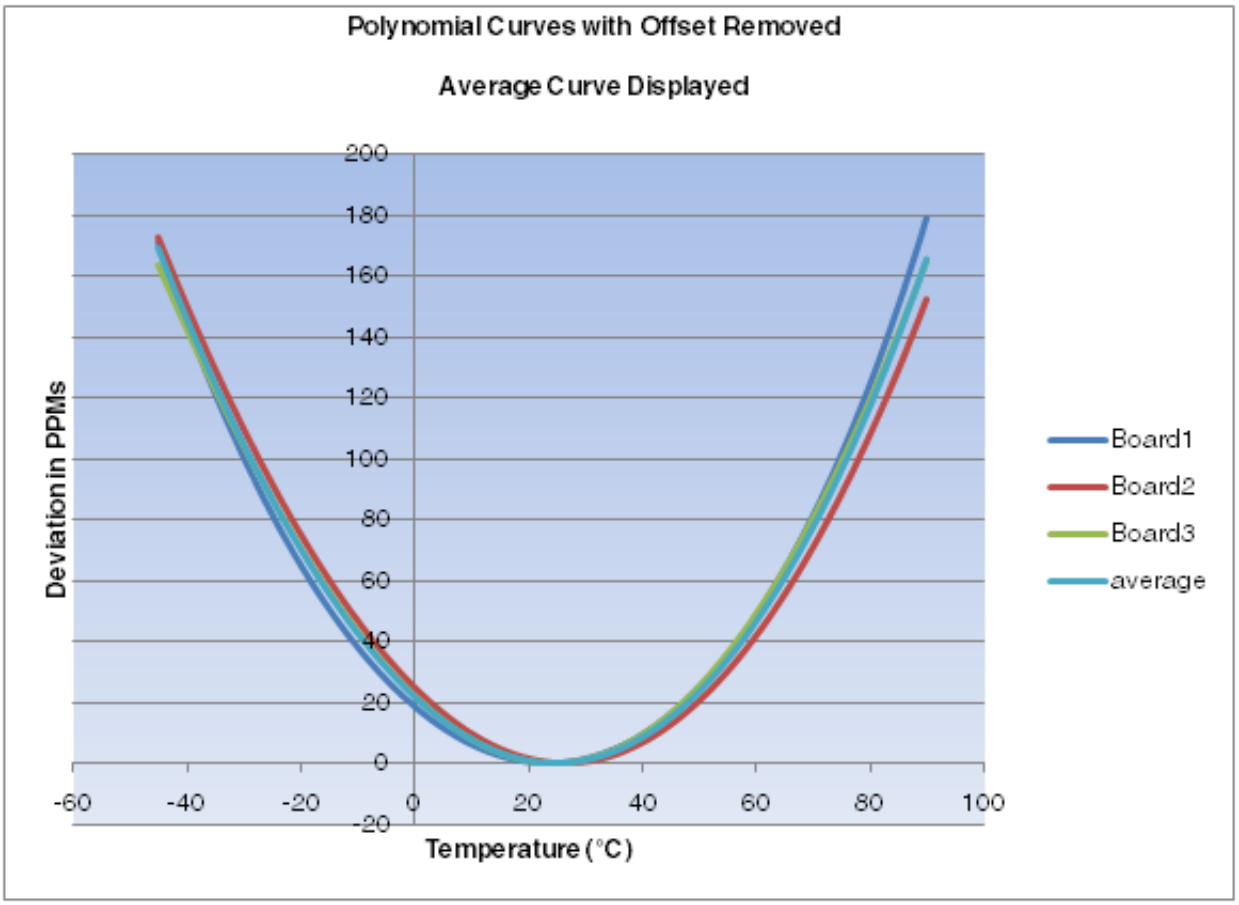
The following graph is the visual representation of these 3 polynomials.

Figure 5. Conversion to polynomials



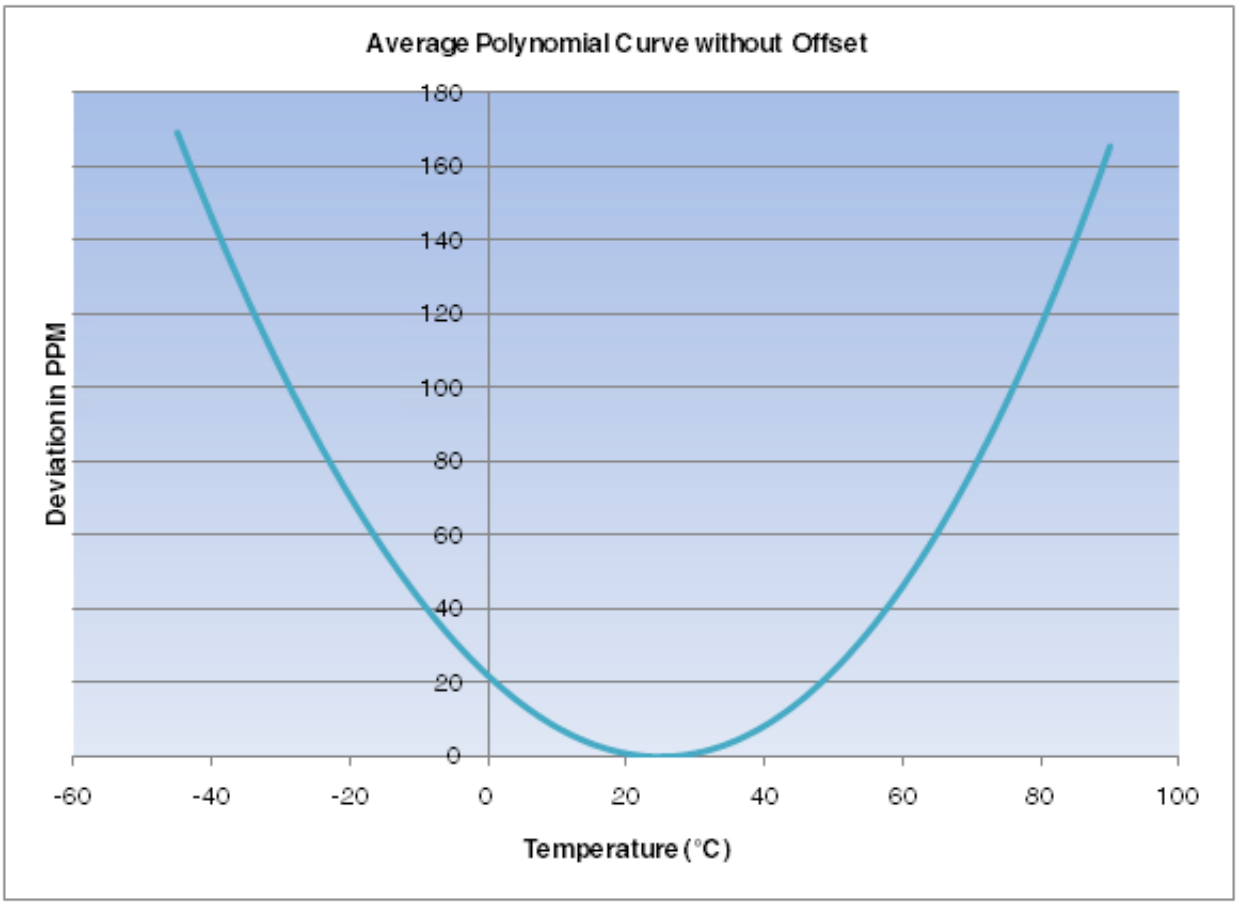
The next step is to remove the offset presented by each crystal at room temperature (25 °C), to create curves that show no deviation at this temperature. To do this, subtract to each curve its value at 25°C. In the next figure we plot again these curves with no “offset”

Figure 6. Polynomial curves offset removed



Then compute, for every temperature inside the range, the average of these 3 curves. The resulting average curve is considered to be the representative of this crystal model. This will be used as basis to create the compensation table:

Figure 7. Average Polynomial curve



Using formula 1.1 that translates deviation into compensation value the compensation table can be created. This table assumes that the crystal shows no deviation at 25 °C. It is necessary then, for each crystal, to compute the offset effectively shown at 25 °C and add this personalized value to the algorithm, so it compensates for every crystal with its appropriate offset.

The computation of this offset is called the calibration procedure. This calibration has to be done for every crystal individually. The following section suggests a method to quickly measure this in a production environment.

4 Calibration Procedure

Calibration carries the idea of fine-adjusting an algorithm to a specific case. It is important to note that when measuring the deviation of each crystal to create a compensation table, the universal counter's internal time base was implicitly used as reference for these measurements. It is important to calibrate each crystal with the same time base used for the compensation table, this is to maintain coherency in the approach.

The fundamental idea is to compare two time signals, one proportional to the individual crystal frequency, and the other one, proportional to the reference time base that is assumed to be exact.

As signal that is proportional to the crystal frequency continue using the same 2 second-period signal generated by the 1 Hz IRTC interrupt that was used earlier. As reference time base use a 1 MHZ square wave signal that alternates between 0 and 3 Volts (the microcontroller's logical 0 and 1 values). This is the signal whose frequency is considered to be exact.

In this case this signal is generated by a waveform generator. The precaution was taken of doing a service calibration of the waveform generator together with the universal counter used for the compensation table computation, so that both of their internal time bases are the same.*

The MCF51EM256's TPM module was used with the TPM clock configured to use an external signal as its clock source. This external signal is the 1 MHZ reference signal. By injecting this signal into the TPM's TPMCLK pin, the TPM will increment its internal counter at the pace of the1 MHz signal. Like this, 1 oscillation of our reference signal will increment the TPM's internal counter by 1 count.

The fundamental idea is to start counting reference pulses at the start of the IRTC's 1 second pulse and finish counting at the end of this pulse. The TPM is configured to start counting with the first IRTC's 1 Hz interrupt and is stopped in the next 1 Hz interrupt.

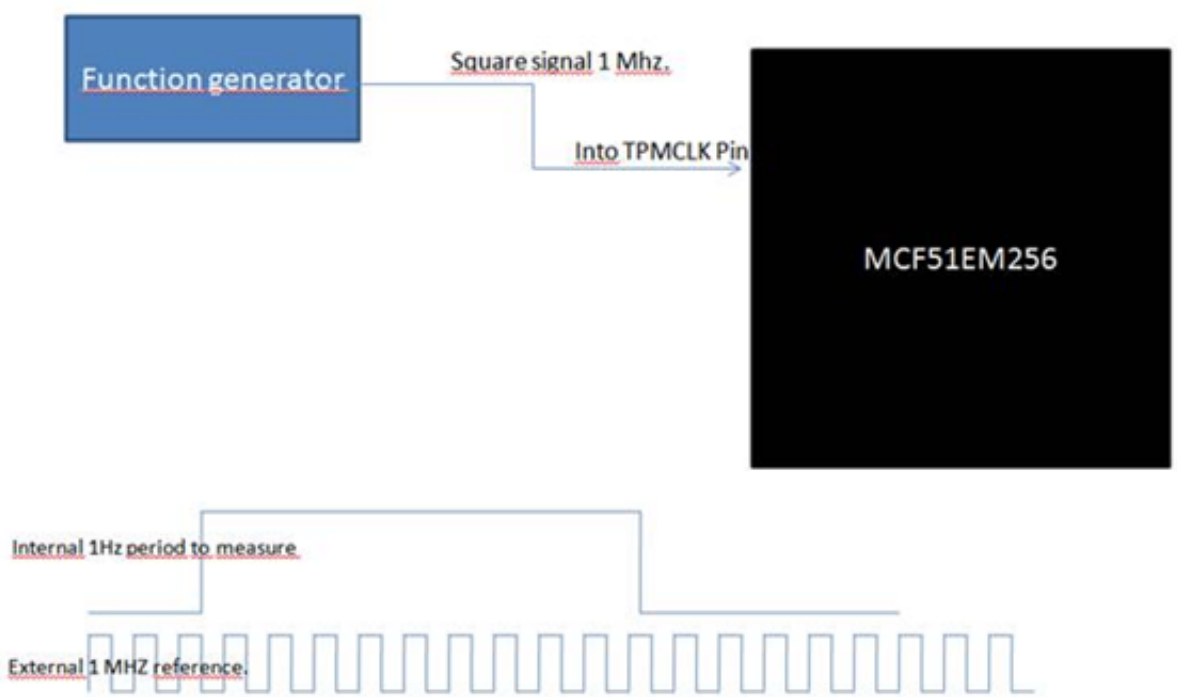
If both the IRTC Crystal and the reference signal time bases are identical, expect to see exactly 1 million counts at the end of the 1 second period. The number of counts above or below this 1 million number directly gives the deviation of the crystal in parts per million.

The TPM counter is a 16-bit counter. It is therefore impossible to count up to 1 million without overflowing the counter. It is necessary to keep count of the number of overflows to be able to reconstruct at the end of the exercise the number of counts that actually took place.

The following figure illustrates this set-up:

*To further increase precision another recommendation would be to use a GPS-clock based 10MHz reference as an external clock to generate the 1MHz reference signal as well as the monitoring frequency counter. Even if a service calibration was performed to make sure that the 1MHz reference generator and the frequency counter show a very good relative accuracy against each other, there is a residual risk for the absolute accuracy of the crystal frequency measurement. According to the method described here, if the frequency generator has a frequency error of +5ppm, but the crystal is "perfect", the result in the TPM counter will be 1000005, and the misleading result is that the crystal shows a frequency offset of 5ppm.

Figure 8. Calibrate



4.1 Calibration procedure results

These are the results obtained with this method. In this table the manual measure refers to the procedure where you measure the deviation directly with the universal counter, and not by firmware. Automatic measure refers to the procedure described here. In the four crystals used for this test, we observed always an almost constant difference in the measure of ~3.5 PPM. After researching the cause of this, this offset was justified by an internal synchronization mechanism of the TPM module. This delay can be considered constant and can be corrected simply by considering it in the formula that the software uses to display the result in PPMs.

	Manual Measure	Automatic Measure	Difference
Crystal 1	156 PPM	152.5 PPM	3.5 PPM
Crystal 2	146 PPM	142.5 PPM	3.5 PPM
Crystal 3	230 PPM	226.5 PPM	3.5 PPM
Crystal 4	152 PPM	148 PPM	4 PPM

This formula is as follows:

Deviation (in PPM) = (Number of TPM Counter Overflows * 65536) + (Final content of TPM counter) – 1000000 + 3.5

With this formula, obtain a very similar measure (within +/- 1PPM) to the measure done manually.

We ran tests where the same signal deviation was measured 30 times, to understand the consistency of the measure. In all 4 crystals there is little variation in these measures, showing typically no differences in the measure, and occasionally, no more than +/- 1 PPM difference inside this 30 samples set.

Therefore it is conclude that measuring one single sample (1 second) is enough to provide a measurement with a relative accuracy of ~ +/- 1 PPM.

Notice that when launching the calibration routine, the software waits for the 1 Hz interrupt flag to be set one first time to start counting. In a worst case scenario, the maximum time to wait for this flag is 1 second. Therefore the calibration routine takes 2 seconds to execute, 1 second to wait plus 1 second of measurement.

4.2 Calibration plus compensation combined results

To test this approach this methodology was applied to two crystals never before seen from the same model. These two crystals were not the same as those used to generate the compensation table.

Measurements in several points across the entire temperature range, to study the effect of compensation was taken.

To measure the effect of compensation it is important to understand that you now have to look at the average duration of the 1 Hz pulse, as mentioned in the first part. In average expect this pulse to be exactly 1 second if the compensation is ideal.

The measurement is the same as before, inject the 1 Hz-toggled GPIO signal into the universal counter and now look at the average value of the signal period. The test conditions were as follows: We waited for the temperature to stabilize inside the temperature chamber for 15 minutes (by monitoring the output of the external temperature sensor). Computed was the statistical average of the signal period over a set of 400 seconds.

Below are the results of the behavior of these two crystals. The last data points were taken for a temperature above the maximum requested range of 85 °C. (measured at 88 °C)

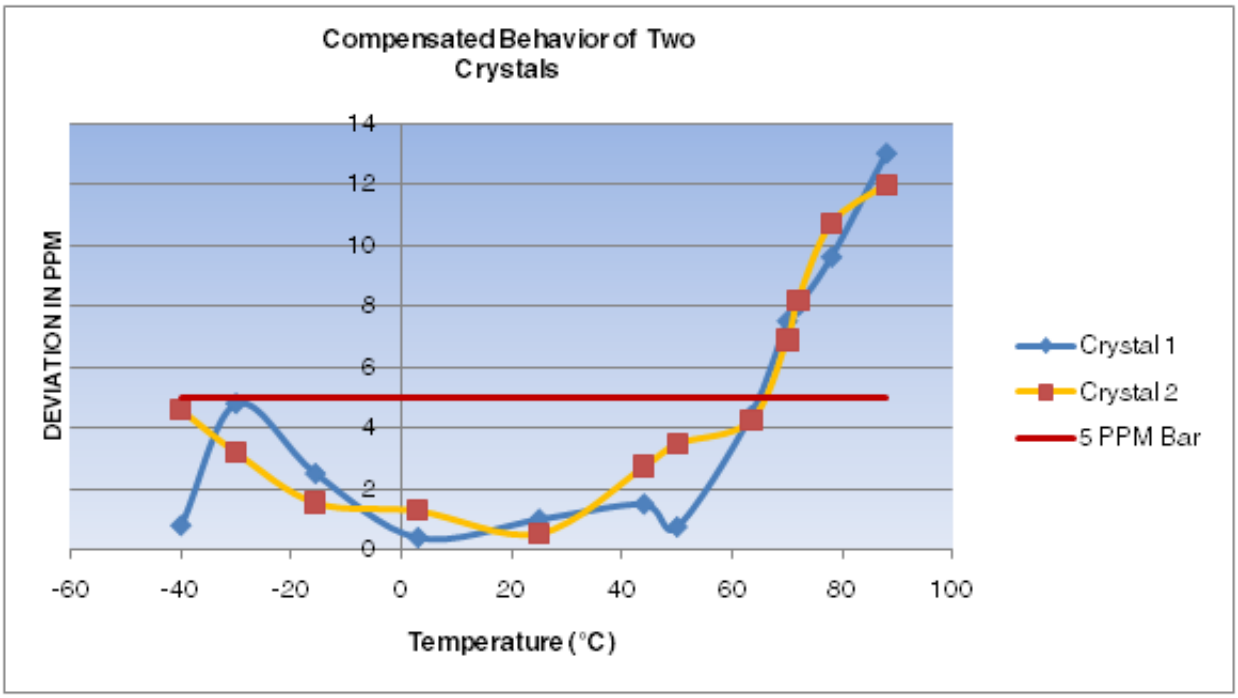


Figure 9. Compensated behavior

5 Conclusion

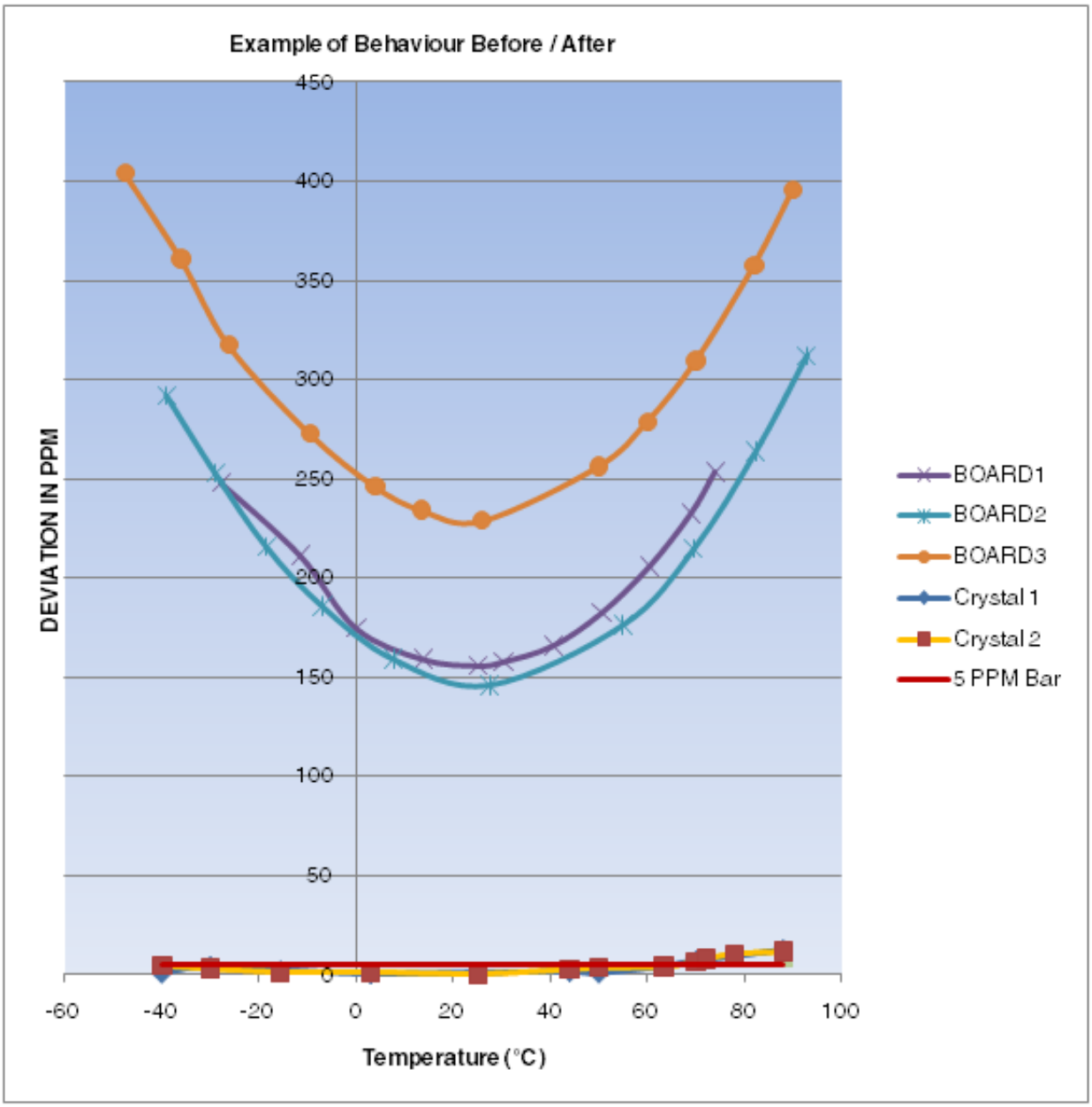
It is possible to dramatically improve the precision of the IRTC time keeping functions across a wide temperature range by using the internal compensation mechanism. Crystals that can originally present up to 400 PPM of deviation in extreme temperatures are shown here adjusted down to a 5 PPM threshold inside a temperature range going from -40 °C up to +65 °C approximately. Outside this range, the 2 crystals studied here showed that the algorithm kept their deviation under 14 PPMs in the 65 °C to -85 °C range. Only one calibration point (at 25 °C) was needed for these results.

This study was made on a reduced amount of crystals. A larger amount would be more suitable to produce more statistically valid data. But the power of the compensation mechanism is correctly shown here.

As a last figure we present in the same graph, the set of 3 crystals with their original deviations next to the other two compensated crystals to visually appreciate the benefits of the compensation across temperature.

*J4)

Figure 10.



NOTE

In this figure the crystals used for the “before” and “after” behaviors are not physically the same crystals (but are the same model).

Appendix A Software Implementation

In this appendix the description of the main functions are provided, the macros and variables that were written to implement this study. More information can be found inside the source code, where care was taken to write explanatory comments.

The system configuration is the following:

- Busclock running from FEI mode at 10 MHZ
- Watchdog disabled
- UART enabled on SCI3 at 9600 Bauds, no parity, 1 stop bit, no flow control.

The UART is used only for feedback messages sent by the calibration routine.

A.1 Macros

Below is a list of symbols implemented as #defines in the example software that are important to understand for the usage of the application.

COMPEN_INTERVAL

This is the value that determines the period in seconds at which the compensation interval will insert a compensated pulse. This value is important and affects directly the values that are present in the compensation table. If this value is modified, the compensation table values need to be modified accordingly. This value impacts the PPM resolution that can be corrected. Higher values may allow to correct the finer values of PPMs, but the maximum amount of PPMs that can be corrected is reduced in the same proportion.

COMPEN_TABLE_NUMBER_OF_ENTRIES

This value has to be equal to the number of entries in the compensation table. Modifying the compensation table by adding or removing elements needs to be followed by adjusting this value.

REEVALUATION_INTERVAL

This value determines how many times vfnRTCCompensate function needs to be called by the application before reevaluating the values written into the compensation registers. This value should be chosen in function of how fast the temperature is prone to change.

OFFSET_PPM

This value is present in case you want to evaluate the algorithm without using the Automatic Calibration routine (which needs a hardware setup to be in place).

If this is the case then you must manually measure the deviation of the crystal at room temperature (25 °C) and write the measured value into this macro. This macro expects units of deviation to be expressed in tenths of PPMs. For example, if you measured 1234 PPMs of deviation at room temperature, you must write the value 1234 into this macro.

If you decide to use the manual calibration instead of automatic, then :

1. Write the appropriate value into OFFSET_PPM macro
2. Comment out the call to vfnRTCCalibrate() inside the code
3. Inside the code of vfnRTCCompensate(), in the COMPEN_ONGOING state, instead of calling s8ComputeCompenValue function with the global variable Deviation as argument, call it with the macro OFFSET_REGVALUE.

```
s8ComputeCompenValue(ADC1Average, OFFSET_REGVALUE);
```

OFFSET_REGVALUE

This macro is only used in case you wish to use the manual calibration and should be used as explained above. You do not need to write any value into this macro as the preprocessor automatically computes the appropriate value for this symbol based on the value you write into OFFSET_PPM.

NUMBER_OF_CALIBRATION_MEASURES

This macro is used by the calibration routine. It typically has the value 2. This represents the number of times the calibration routine measures the internal 1 Hz pulse by counting external reference pulses. The minimum amount to choose is the value 2 because the first measure will always contain garbage values because it is impossible to synchronize the launch of the TPM counter and the 1Hz signal at the same time. Therefore when the first 1 Hz interrupt comes this is used by the code to clear the counter and start synchronizing. It is until the second 1 Hz pulse that useful values are produced. This macro was implemented during the test phase of this study to generate several consecutive measures and allow to compare the stability of the measure.

A.2 Variables

Most variables used in the code are self explanatory and their usage can be deduced from the comments in the code. One mention worth having is for the array that constitutes the compensation table:

tCompenEntry CompTable [COMPEN_TABLE_NUMBER_OF_ENTRIES]

The application expects to have one array called ComTable which is an array of COMPEN_TABLE_NUMBER_OF_ENTRIES elements of type tCompenEntry.

Each element in the array is simply two numbers, the first represents the temperature in tenths of Celsius degrees and the second element represents the ideal compensation register value for that temperature.

For example :

```
tCompenEntry CompTable[COMPEN_TABLE_NUMBER_OF_ENTRIES]=
{
  -450,-55,
-440,-54,
-430,-52,
-420,-51,

….

}
```

In the code above 4 entries are displayed, indicating that for temperatures -45,-44, -43, and -42 celsius degrees, the algorithm should write the values -55, -54,-52, and -51 into the IRTC compensation value register, respectively.

A.3 Functions

Name of function—s8ComputeCompensationValue

Arguments—signed long Temparg, signed char Offset

Return Value—signed char CompensationValue

This function returns the appropriate compensation value (CompensationValue) to be appied when provided by the current temperature (Temparg) and the offset (Offset) of the crystal.

To do this, it looks up the compensation table. It also applies an interpolation function to decide what compensation value to return if the temperature provided in the argument is not an entry in the table, but is a value in between two entries. In this case it returns a linear interpolation between the two closest neighboring values in the table.

Name of function—vfnIRTC_WriteCompensationValue

Arguments—unsigned char u8Interval, signed char s8Value

Return Value—Void

This function writes to the IRTC registers, the values for the Compensation Interval, and for the compensation value. u8Interval represents the compensation interval in seconds, and its values can range from 0 to 255. s8Value represents the signed number of clock cycles to add or subtract.

Name of function—vfnRTCCompensate()

Arguments—Void

Return Value—Void

This function represents the compensation state machine. It has two simple states. The first one (INITIALIZE state) resets the IRTC compensation registers to zeroes and initializes internal variables to zeroes prior to launching the first temperature measurement by triggering an ADC measurement. The second state represents the ONGOING state where a measurement of temperature is taken for every call of the vfnRTCCompensate function. When a number of temperature measurements have been done, an average temperature value is computed, and the compensation value is adjusted based on the new average temperature.

The user has to call this function periodically for the algorithm to react to changes in temperature. The rate at which you call this function, together with the value chosen for the macro REEVALUATION_INTERVAL, determines the rate at which you will be reevaluating the compensation values written into the IRTC registers.

The macro REEVALUATION_INTERVAL defines the number of times that the function vfnRTCCompensate() has to be called before reevaluating the compensation value.

For example, if the VfnRTCCompensate() function is called each 3 seconds, and REEVALUATION_INTERVAL is worth 10, then the compensation value will be re-evaluated every 3 x 10 = 30 seconds. Every 30 seconds a new average temperature representing the past 30 seconds will be computed and the compensation registers are re-written based on this.

Name of function—vfnRTCCalibrate()

Arguments—Void

Return Value—Void

It disables/enables interrupts and leaves interrups enabled. It stores the measured value in the Deviation global variable which is later used by the vfnRTCCompensate() routine.

NOTE

Make sure that the reference signal is injected into the clock pin of the TPM module prior to calling this routine, otherwise this routine will not produce coherent results. The reference signal must be a square wave (0 volts – 3 Volts) with a frequency of exactly 1 Mhz.

Important

This routine has to be called when the crystal is at room temperature. (25 °C) .

A.4 Interrupt Service Routines

For understanding example, three interrupt service routines are worth mentioning.

RTC_ISR

The RTC module is configured to trigger an interrupt every second. This is the interrupt that triggered in a delay or rushed fashion when the compensation mechanism is activated.

This routine performs three actions:

- Setting up a software flag that signals a one-second period has elapsed (Flag_1Hz). This flag can be monitored by the application to schedule periodic tasks every one second.
- Toggling a GPIO pin that can be used to externally measure the duration of the second by a counter (explained in the first part of this document)
- Store TPM counter values and TPM overflow values that are used by the calibration routine to measure the duration of a 1 Hz pulse when compared to a reference signal.

ADC1_ISR

This interrupt is triggered on conversion of an ADC conversion. The ADC is configured to measure channel AD5, where an external temperature sensor is connected. The measured value represents temperature.

On completion of a conversion, this ISR:

- Stores the finished conversion value in a buffer

- Performs a scale change to transform the value into centigrades
- Keeps track of how many conversions have been finished in the past
- Every REEVALUATION_INTERVAL, computes an average temperature value
- Signals the application that the new conversion data is ready by setting a software flag (ADC1DataReadyFlag)

TPMOverflow_ISR

This interrupt is triggered every time the main 16-bit counter of the TPM module overflows from 0xFFFF to 0x0000. It increments a counter to keep track of the number of times this counter has overflown. This is used by the calibration routine.

A.5 General Description

Inside the application main function, a simple endless loop is implemented where the application checks for the Flag_1Hz flag. Every second, calls the routine vfnRTCCompensate(), which in turn, triggers ADC conversions for temperature measurement. At the end of REEVALUATION_INTERVAL seconds, the same vfnRTCCompensate() function updates the RTC compensation registers by calling the s8ComputeCompenValue() and vfnIRTC_WriteCompensationValue() functions.

Prior to entering the main endless loop, the application calls the function vfnRTC_Calibrate(), which supposes that a reference signal of frequency 1 Mhz is being injected in the TPMClock input pin.

NOTE

This study was carried on a particular crystal model which is not the model that comes as a standard crystal in the DEMOEM256 Freescale boards. To adjust the algorithm to the crystal present in the DEMOEM256 board, the compensation table needs to be modified as explained in the beginning of this document.

Appendix B Formulas

In this appendix are the equations that can be used for trendline computations in Excel to generate the trendline coefficients. You can use these formulas to calculate predicted and the values for given values of x.

These equations assume that your sheet has two named ranges, x and y.

Linear Trendline

```
Equation: y = m * x + b
m: =SLOPE(y,x)
b: =INTERCEPT(y,x)
```

2nd Order Polynomial Trendline

```
Equation: y = (c2 * x^2) + (c1 * x ^1) + b
c2: =INDEX(LINEST(y,x^{1,2}),1)
C1: =INDEX(LINEST(y,x^{1,2}),1,2)
b = =INDEX(LINEST(y,x^{1,2}),1,3)
```

3rd Order Polynomial Trendline

```
Equation: y = (c3 * x^3) + (c2 * x^2) + (c1 * x^1) + b
c3: =INDEX(LINEST(y,x^{1,2,3}),1)
c2: =INDEX(LINEST(y,x^{1,2,3}),1,2)
C1: =INDEX(LINEST(y,x^{1,2,3}),1,3)
b: =INDEX(LINEST(y,x^{1,2,3}),1,4)
```

Higher Order Polynomial Trendline

Notice the pattern in the two preceding sets of formulas.

Source—The Spreadsheet page for Excel users and developers, http://spreadsheetpage.com/index.php/tip/chart_trendline_formulas/

How to Reach Us:

Home Page:
www.freescale.com

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

USA/Europe or Locations Not Listed:
Freescale Semiconductor
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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4310
Rev. 0, 07/2011

freescale semiconductor