Freescale Semiconductor
Application Note

Document Number: AN4168
Rev. 1, 07/2010

ADC16 Calibration Procedure and Programmable Delay Block Synchronization

For MC9S08GW64

by: Neeraj Mangla, Inga Harris

1 Introduction

The MC9S08GW64 MCU has integrated 16-bit ADC and programmable delay block (PDB). Two independent channels of PDB are available for two independent instances of ADC. The integration of ADC and PDB is not standardized across different devices. Integration of PDB and ADC in MC9S08GW64 MCU is similar to the integration available in MCF51EM256 MCU. This application note describes the procedure required to perform the analog-to-digital converter (ADC) auto-calibration function as well as defines the multiple configuration of ADC for different applications. This application note also provides the software that may be used as a template for application code development for ADCs.

The second section of this application note describes the integration of the ADC16 and PDB peripherals for the MC9S08GW64 MCU. This section also provides the software configuration of PDB for continuous hardware

Contents

trigger mode. This application note mainly describes how ADC is used in Energy metering application. It also describes how to use ADC and PDB modules together in a simple application and synchronize them.

NOTE

With the exception of mask set errata documents, if any other Freescale document contains information that conflicts with the information in the device reference manual; the reference manual should be considered to have the most current and correct data.

2 ADC16 Calibration

The 16-bit ADC requires self-calibration to improve the ADCs linearity and to ensure that the specified accuracies in the data sheet are met. This calibration must be executed at least once after a power-on. These calibration values must be stored in non-volatile memory like flash. On each reset these calibration values, residing in non-volatile memory, can be uploaded in the ADC calibration registers. If these calibration values are not stored in the non-volatile memory then calibration must be performed after each reset as shown in the accompanying software. The calibration frequency is a trade-off between accuracy and ADC overhead. This calibration must be executed to generate the offset and gain compensation values. These values are automatically subtracted (offset) and scaled (gain) during the conversion sequence to compensate for linearity errors in the SAR's internal DAC output as illustrated in Figure 1. The offset registers are also user-configurable for custom offset.

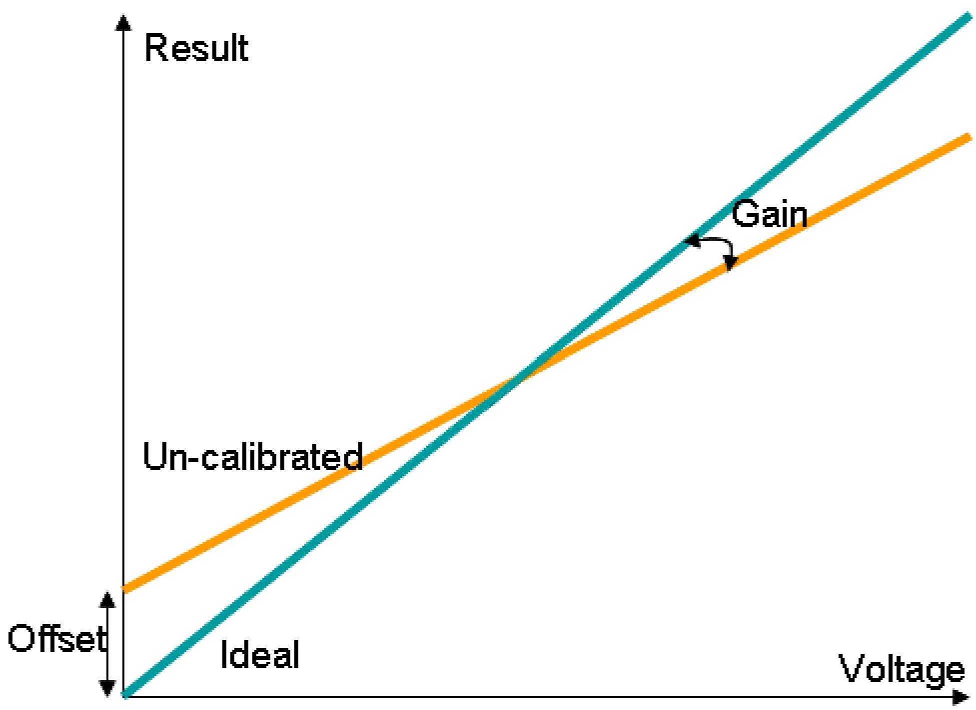

Figure 1. Gain and offset effect on result

Calibration is a three step process:

1. Configure the ADC
2. Initiate the hardware calibration and wait for the COCO flag

3. Generate gain compensation values

The values in the offset register are left-justified two's-complement values. Offset is subtracted from the right-justified successive-approximation conversion result before the compensated value is shifted into the ADC result register. The offset value can be adjusted by the user. However, the adjusted calibration offset value must be stored in the secured memory. The un-calibrated default offset value is zero.

There are two gain registers, Plus-Side (ADCnPGH:L) and Minus-Side (ADCnMGH:L). They are used to compensate for a non-ideal output response of the internal DACs on each side of the differential ADC input. The Plus-Side value is used in single-ended mode and on the Plus-Side input of the differential inputs. The Minus-Side is used only in differential modes and on the Minus-Side of the differential input.

The values in the 16-bit gain registers represent integers from a recommended minimum value of 1.0 (0 x 8000), which is a full sample to a maximum allowed value of 1.03125 (0 x 83FF), with the decimal point fixed between bit 14 and 15 of the 16-bit register. The un-calibrated default register value is 1.0239 (0 x 8310). The input voltage is scaled by the appropriate gain compensation value, the positive input is scaled by the Plus Gain register (ADCnPGH:L), and the negative input is scaled by the Minus Gain register (ADCnMGH:L). See Equation 1 for the single ended mode and Equation 2 for the differential mode.

Single ended:

$$Code_{out} = 2^N\left[\left(\frac{V_{in}}{V_{REFH} - V_{REFL}}\right)\left(\frac{Cs_p}{Cn_p}\right)\right]$$

Eqn. 1

Differential:

$$Code_{out} = 2^N\left[\left(\frac{V_{in_p}Cs_p}{Cn_p} - \frac{V_{in_n}Cs_n}{Cn_n}\right)\left(\frac{1}{V_{REFH} - V_{REFL}}\right)\right]$$

Eqn. 2

Where Cs_x is the sampled capacitor and Cn_x is the nominal capacitor, and x is p-positive or n-negative.

2.1 Calibration Flow

Figure 2 shows the calibration process flow chart. The calibration results can be affected by the clock source, frequency, power and conversion speed settings, voltage reference, hardware average function, the sample time and to a much lesser extent, environment. Therefore, if the application changes any of these settings the user may decide to re-calibrate at each setting change or calibrate for the setting requiring the highest accuracy. For the highest achievable accuracy, the hardware average setting of 32 (maximum) must be used within the register ADCnSC3. The ADC clock frequency must be between 2-4 MHz, the VREFH must be equal to VDDA, and the voltage and temperature must be in line with the running environment. The input channel, conversion mode setting, compare function values, resolution, and differential and single-ended settings have no effect on the calibration result. However, they can be configured prior to calibration to later save time.

The ADTRG bit in ADCnSC2 must be cleared to enable the conversion initiation by the software trigger before initiating the calibration sequence by setting the CAL bit in ADCnSC3. After the calibration sequence has been initiated the software must wait for the conversion complete bit (COCO) to be set.

The CALF is set and the CAL bit cleared if any ADC registers are written during the calibration sequence, if the stop mode is entered during the calibration sequence, or if the CAL bit is set while the ADC is in hardware trigger mode. Application code must be written to avoid or at least manage these scenarios. In the software example in AN4168SW, the calibration error flag is cleared and the code moves on completing the function with the fail calibration results. After the ADC has completed its auto-calibration, the application code must complete the calibration procedure by calculating the gain compensation values for the plus and minus side DACs using the following procedure:

- Initialize (clear) a 16b variable in RAM
- Add the following plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable
- Divide the variable by two
- Set the MSB of the variable
- The previous two steps can be achieved by setting the carry bit, rotating-right through the carry bit on the high byte and again on the low byte
- Store the value in the plus-side gain calibration registers ADCPGH and ADCPGL
- Repeat procedure for the minus-side gain calibration value.

The example code in AN4168SW is written in C to demonstrate the intension. However, for increased efficiency, the above routine can be written in assembly and then divided by 2 and set MSB part, this is achieved efficiently by setting the carry bit and rotating-right through the carry bit on the high byte and again on the low byte. Further calibrations can be initiated by clearing and then setting the CAL bit in ADCnSC3. To allow repeated calls to the example calibration function, the CAL bit can be cleared prior to exit. However, this may leave the application open to unwanted calibrations in the event of the CAL bit being written unintentionally.

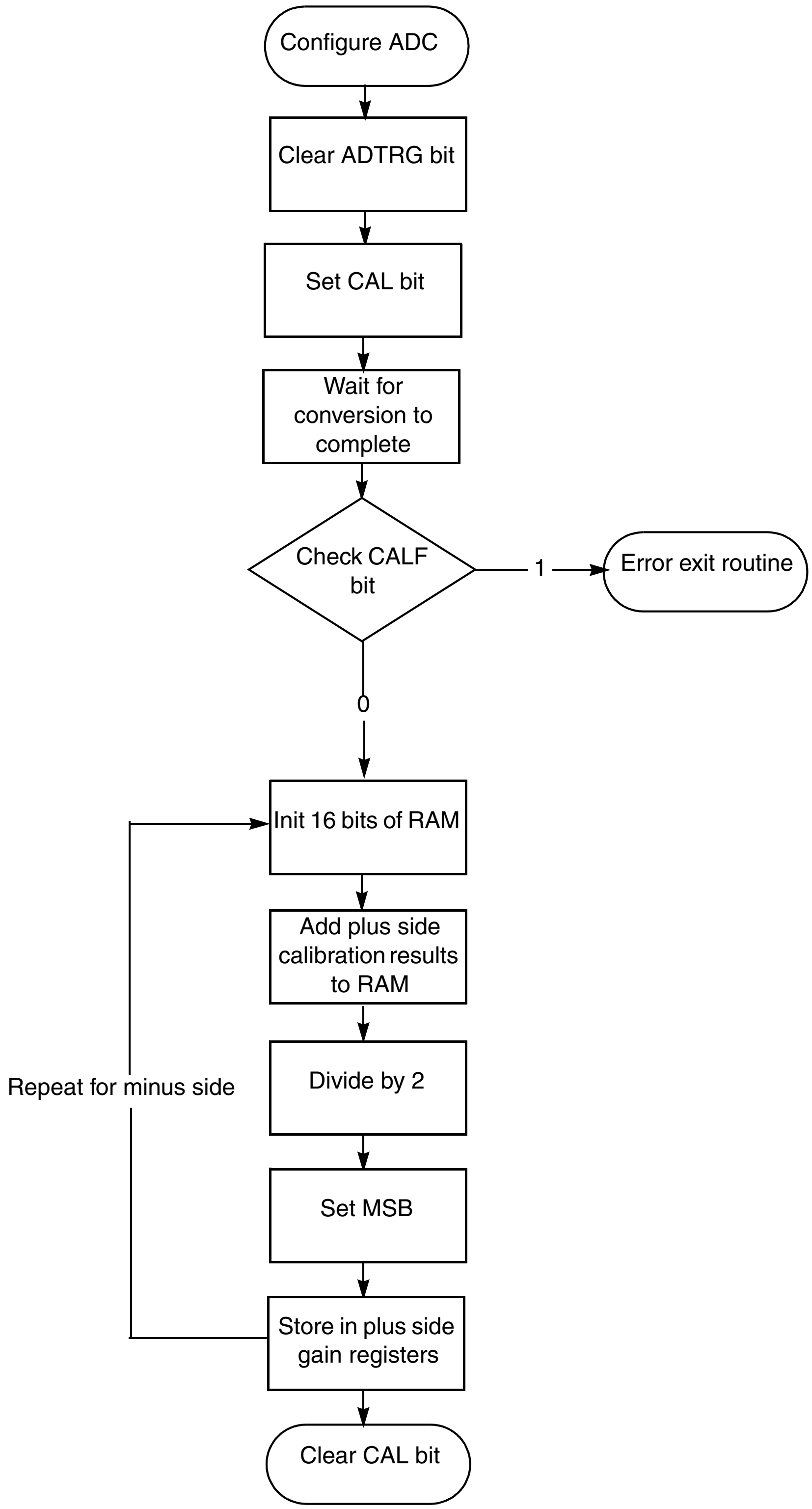

Figure 2. Calibration flow chart for the ADC16 module

2.2 Calibration Latency

The calibration routine may take as many as 15,000 ADCK cycles plus 100 bus cycles. To reduce this overhead, the calibration values, offset, plus and minus side gain, and plus and minus side calibration values, can be stored in the secured flash (non volatile memory) by the application code after the initial calibration. This reduces the calibration overhead to 20 register store operations on subsequent POR, internal reset, and stop2 mode recoveries.

Other ways to reduce the time taken to run calibration are:

- Reduce amount of hardware averaging used (this has an effect on the accuracy).
- Disable interrupts to avoid calibration fails and get results as soon as possible.
- Use the described assembly instructions (setting the carry bit and rotating-right through the carrybit)
- While doing single ended calibrations, the Minus Side Gain register computation can be skipped.

3 ADC16 and Programmable Delay Block (PDB) Integration

This integration has been optimized for use in metering applications. Energy metering applications take two measurements for every reading (voltage and current) at regular intervals (due to fundamental frequency of mains supply) with only one interrupt per cycle. By using the PDB, hardware triggers can be sent to the ADC to initiate conversions at pre-set time intervals for detailed control of ADC conversion timing. Each ADC module has two status and control 1 registers (ADCnSC1A:B) and two result registers (ADCnRA:B) which correspond to a PDB trigger derived from one event and the two delay timings, PDBCHnDELA and PDBCHnDELB.

Table 1 shows how the PDB Channels, Pre-triggers, and ADC hardware trigger signals correspond on the MC9S08GW64.

Table 1. PDB channel pre-triggers/ADC hardware triggers

PDB Channel PreTrigger	ADC Trigger Select
PDBCH1 PreTriggerA	ADC1HWTSA
PDBCH1 PreTriggerB	ADC1HWTSB
PDBCH2 PreTriggerA	ADC2HWTSA
PDBCH2 PreTriggerB	ADC2HWTSB

Every ADC has associated differential input, temperature sensor, bandgap, Vrefo, and Vrefl channels. The single ended channels are also associated with different ADCs. A detailed table of the MC9S08GW64 ADC channel assignments are given in Table 2.

Table 2. ADC Channel assignments

ADCH	Input function of ADC0	Input function of ADC1
00000(0)	DADP0/DADM0	DADM1
00001(1)	DADM0 (As Single Ended)	DADP1/DADM1
00010(2)	AD2	NA1
00011(3)	NA	AD3
00100(4)	AD42	RESERVED
00101(5)	AD5	NA
00110(6)	VLL1	VREFH
00111(7)	VCAP1	VREFL
01000(8)	NA	AD6
01001(9)	NA	AD7
01010(10)	VLL2	NA
01011(11)	VCAP2	RESERVED
01100(12)	AD8	RESERVED
01101(13)	AD9	RESERVED
01110(14)	RESERVED	RESERVED
01111(15)	VREF	OUT
10000(16)	NA	AD11
10001(17)	PMC	VREF
10010(18)	NA	NA
10011(19)	NA	NA
10100(20)	AD12	PRACMP0
10101(21)	AD13	PRACMP1
10110(22)	NA	AD15
10111(23)	NA	AD14
11000(24)	NA	PRACMP2
11001(25)	NA	NA
11010(26)	Temperature Sensor	Temperature Sensor
11011(27)	Bandgap	Bandgap
11100(28)	NA	NA
11101(29)	NA	NA
11110(30)	NA	NA
11111(31)	NA	NA

CAUTION

Regardless of whether the application requires the RTC module or not the RTC and tamper must be initialized to stop interrupts and resets. There is a finite amount of time to do this. It must be executed before any application functions are called (that is in the initialization code).

```
// Disable write protection sequence 00 - 01 - 11 - 10
IRTC_CTRL = 0; //0b00;
IRTC_CTRL = 1; //0b01;
IRTC_CTRL = 3; //0b11;
IRTC_CTRL = 2; //0b10;

// RTC - Disable All RTC interrupts including tamper
IRTC_IER = 0;
IRTC_CTRL = 15<<9; // maximum Tamper duration
IRTC_CTRL |= 0x02; // Enable Write protect
```

The RTC control bit field (iRTC_CTRL) must run a specific sequence “key” before the interrupts can be disabled. Tamper duration can then be maximized and the write protection enabled.

3.1 Trigger Timings

The MC9S08GW64 series ADC block contains duplicate control and result registers (each dedicated for the PDB trigger), allowing them to operate in a ping-pong fashion, A-B-A-B-A and so on, alternating conversions between two different analog sources (per converter). The Pre-trigger signals are used to indicate which ADC channel is sampled next.

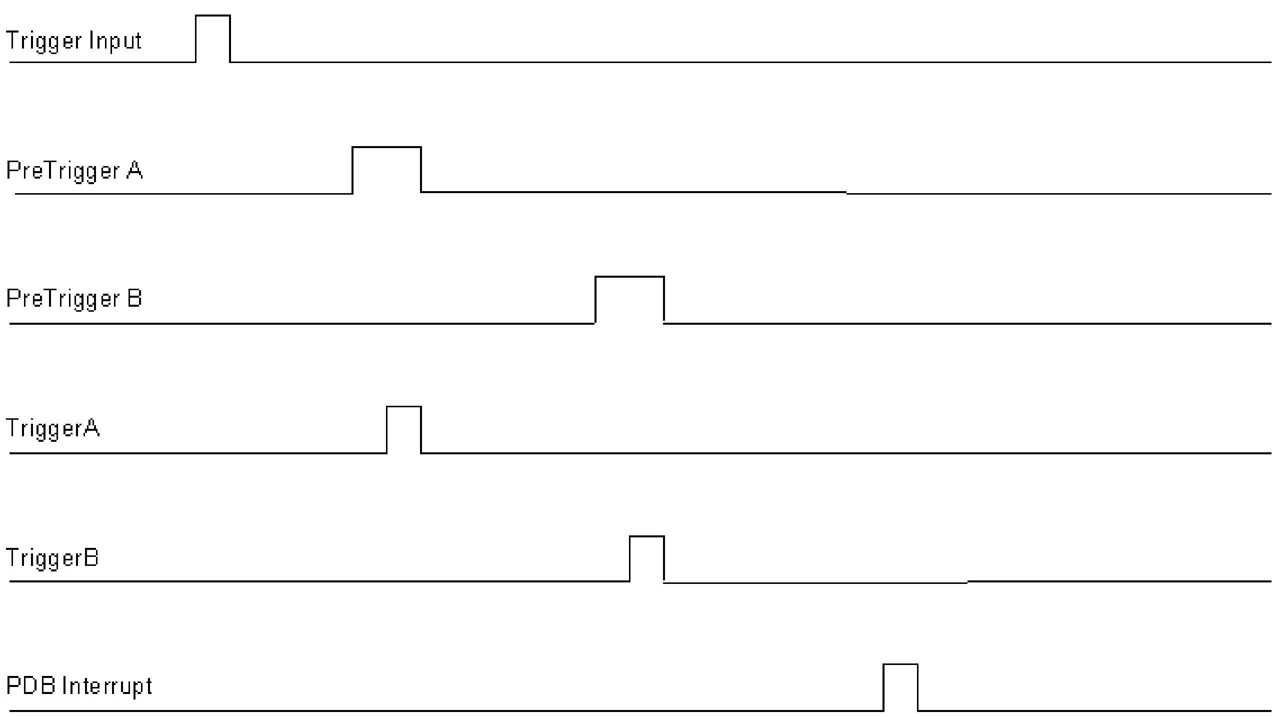

Figure 3. PDB trigger generation timing diagram

NOTE

Care must be taken with the timings. There is only one SAR within each ADC supporting only one conversion per ADC module at any time.

The time between PreTriggerA and PreTriggerB must be scheduled with a large safety margin for the previous conversion to be completed as shown in Equation 3. In this example, a safety margin multiplier of 2.5 is used, but the system design needs to leave enough time only for the previous conversion to complete (accounting for any foreseeable error conditions).

$$CHnDELB – CHnDELA(TypicalConversionTime \times 2.5) / BusClock \qquad \textbf{Eqn. 3}$$

The same is true for the time between the 2nd PreTrigger and the PDB interrupt at time IDELAY as shown in Equation 4 with the same safety margin multiplier.

$$IDELAY – CHnDELB(TypicalConversionTime \times 2.5) / BusClock \qquad \textbf{Eqn. 4}$$

The PDB interrupt occurs if a comparison event has been detected; that is the 16-bit count equals the 16-bit IDELAY value when the module and PDB interrupts are enabled. IDELAY must be set for a safe time after the second conversion is complete. IDELAY can also be used as a single interrupt source to service all ADCs and to calculate the power, save data to memory, and so on.

The PDB module generates a sequence error interrupt (PDB_ERR) if a sequence error is detected on TriggerA or TriggerB. This happens when the CHnDELx has timed out before the previous ADC conversion has completed to enable the application to recover in the event of an error. This error is generated for any channel, cannot be disabled, and is non-maskable. It is imperative that an interrupt service routine be implemented for the PDB_ERR interrupt vector if the PDB is to be used for hardware triggering of the ADC.

4 ADC16 and Programmable Delay Block (PDB) Synchronization

Metering applications need to synchronize the time at which multiple ADC samples are taken with respect to an external trigger or event. The intended function of the PDB on the MC9S08GW64 is to provide controllable delays from either an external trigger or a programmable interval tick to the sample trigger input of one or more ADCs. The MC9S08GW64 has two independent ADCs to allow measurement of independent channels at the same time. The ADC measures the line voltage, and then the line current at a specified time later on each ADC module.

4.1 Scheduling Conversions

Figure 4 shows how the scheduling of conversions in a 1-phase electricity meter might be executed. MC9S08GW64 has got four ADC channels that can be sampled simultaneously, trigger controlled by the PDB.

Each channel of PDB can sends two triggers one after other for one ADC. Voltage signal can be mapped to the channel A of ADC1 and the Current channel can be mapped to the Channel A of ADC2. Remaining channel of ADC1 and ADC2 can be used for other purpose like current sample in case of multi gain stage meter and neutral current sensing.

Both channels of PDB will trigger ADC1 and ADC2 conversion at the same time and result will be available in the result registerA of ADC1 and ADC2 as shown in Figure 4.

Delay between the TriggerA and TriggerB is controlled by the IDLYA and IDLYB registers for each channel.

The PDB interrupt towards the end of the PDB count cycle is controlled by PDBIDLY to occur at a time relative to the counter starting, and can be used to prepare the PDB channels and the ADC modules for the next conversion. Both interrupts are not necessarily required. The PDB interrupt has a higher priority than the ADC interrupts, so the scheduling may not include both interrupts.

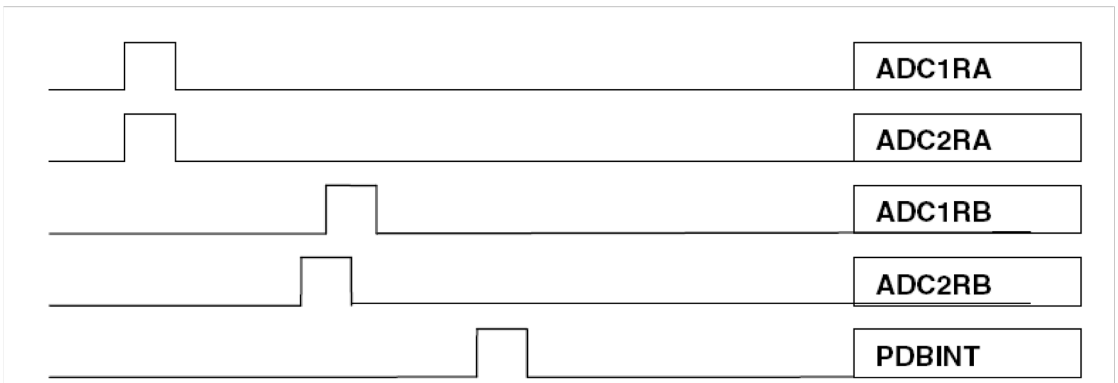

Figure 4. 1-phase electricity meter scheduling example timing diagram

The PDB modulus value (PDBMOD) depends on the electricity fundamental frequency in the country it is to be installed to enable the measurement of the 21_{st} harmonic using Nyquist theorem; that is in a 60 Hz country the sampling frequency could be 2 x 21 x 60 = 2.52 kHz. The maximum bus speed of the MC9S08GW64 is 20 MHz. With a prescaler of 1, the PDB modulus value would be set as $26C0_{hex}$ = 25,000/2.52.

4.2 Synchronizing

AN4168SW contains the example software used to set-up a synchronized PDB and ADC for a system with synchronized inputs as described in this section.

4.2.1 Module Initialization

4.2.1.1 Analog-to-Digital Converter (ADC)

The worst case conversion time of the ADC for the settings must be known before the PDB trigger timings can be calculated.

NOTE

This application note does not recommend ADC settings for best accuracy or timing. The application note titled *Differences Between Controller Continuum ADC Modules* (document AN3827) has information on how settings affect these parameters.

The majority of the ADC initialization would have been executed during the calibration of the module, but because the input channel, conversion mode, compare function values, resolution, and differential and single-ended settings have no effect on the calibration result, these may need to be set along with the trigger type select and interrupts, post-calibration.

The following calculation is based on these settings (20 MHz) and taken from the accompanying software AN4168SW:

```
ADCCFG1 = (ADLPC_NORMAL|ADIV_2|ADLSMP_SHORT|MODE_16|ADICLK_BUS_2)
ADCCFG2 = (ADACKEN_DISABLED|ADHSC_HISPEED|ADLSTS_2)
ADCSC1A = (AIEN_OFF|DIFF_SINGLE|AD5)
ADCSC1B = (AIEN_OFF|DIFF_SINGLE|Bandgap)
ADCSC2 = (ADTRG_HW|ACFE_DISABLED|ACREN_DISABLED|REFSEL_EXT)
ADCSC3 = (ADCO_SINGLE|AVGE_DISABLED)
The conversion time for this configuration is 12.05 Éþs; Calculated from BCT + High Speed Adder
+ Single Time Adder
= (25 ADACK) + (4 ADACK) + (5us + 5 ADACK + 5 bus clock cycles)
= (34 @ 5MHzADACK) + (5 @ 20MHzBUS) + 5us
```

4.2.1.2 Programmable Delay Block (PDB)

With a 20 MHz bus to record the 21st harmonic of a 60 Hz signal, a modulus of 1F00 hex is needed (20 MHz and 2 x 21 x 60 Hz).

As shown in Figure 5, the first set of Triggers, A, happen at the start of the cycle. Figure 3 shows that the time between TriggerA and B in this example must be at least 2.5 times the conversion time; that is at least 30.125 Éþs (PDBCHnDELB – PDBCHnDELA (Typical Conversion Time x 2.5) / Bus Clock) which is 602.5 bus clock cycles with the ADC setting used in Section 4.2.1.1, "Analog-to-Digital Converter (ADC). The minimum TriggerB delay is the TriggerA delay + 25B hex.

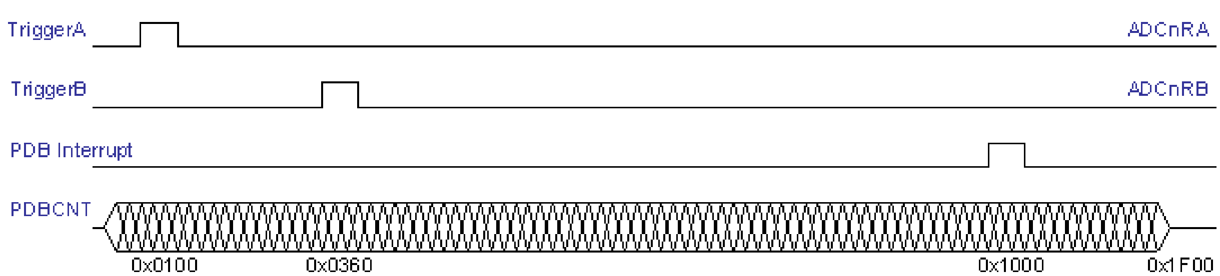

Figure 5. PDB timing

The interrupt delay (IDELAY) must be set to happen before the modulus match occurs, but early enough for the interrupt to be serviced along with all related activities.

4.2.1.3 Interpolation

Interpolation is a method used to construct unmeasured data points from a discrete set of measured data points. For example, in an electricity metering application voltage can be assumed to be a perfect sine wave and therefore the error in the voltage using the interpolation method is insignificant (~ 0.0301%).

Figure 6 shows an example of a case where interpolation can be used. As shown in the "Standard" phase, if the voltage is measured at the start (V_{meas}) and the current must be measured at I_{meas} (for example, small phase delay due to the current transformers), but the ADC conversion takes longer to complete than the allowed phase delay, then the current measurement needs to be taken before the first ADC channel has completed.

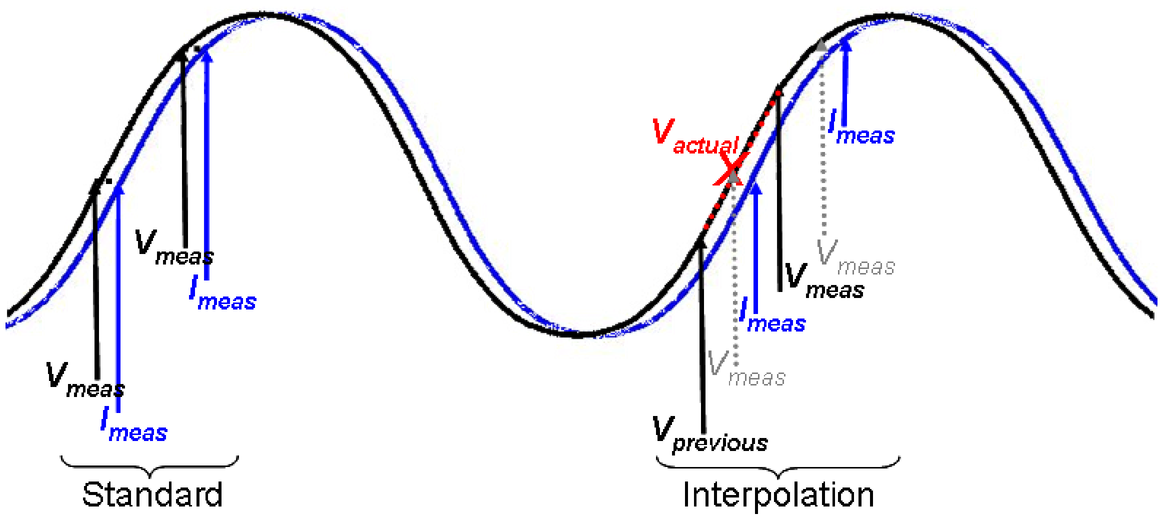

Figure 6. Interpolation example

In this case, when the phase delay for the current comes too close to the time for the voltage, move the voltage measurement to 50% of the measurement period (as shown in the "Interpolation" phase) and interpolate using the current and previous voltage measurements.

Below is a pseudo C example of how this can be achieved:

```
InterpolateMeasurements()
{
static Vmeas, Vprevious;
Vmeas = readVoltageFromADC;
Imeas = readCurrentFromADC;
Vactual = (Vnow + Vprevious) >> 1; // One shift right is divide by two
Vprevious = Vmeas; // Save voltage reading for use next function call
}/ /returns Vactual and Imeas with correct phase
```

4.3 Monitoring Correct Operation

The PreTriggers pre-condition of the ADC to store the next conversion in either result register A or B. If a conversion was already running when another PreTrigger occurred in the same module, a corrupted sequence of results are recorded. Sequence errors, ERRA and ERRB are set when a PreTrigger is requested by the delay time-out before the last conversion was completed. This error generates a non-maskable interrupt.

The PDB sequence error interrupt must check to see which channel caused the error and then the application code can clear the flag and deal with the event accordingly. For example, save all valid data and restart the ADC and PDB, losing only a few records or perform a soft reset.

NOTE

In a robust system environment this event is extremely rare and may be due to significant changes in ADC clocking frequencies. If this interrupt occurs frequently, the system integrity must be investigated. Clocks and external trigger events are most likely the cause.

Figure 7 shows the window where a second ADC trigger generates a sequence error event.

Figure 7. Sequence error window

5 Summary

Calibration is required to achieve the accuracies specified in the data sheet and to meet the systems accuracy requirements as shown in Figure 2. The calibration process does use up some of the application's functional time and if not scheduled appropriately, adds unnecessary overhead to the application. This application note has highlighted the best practice calibration and explained how to minimize that latency. After the ADC is calibrated, the next item to address is the scheduling and synchronization of conversions using the ADC's hardware triggers from the PDB module. Figure 3, Figure 4, Figure 5 and Figure 6 illustrate the integration and application of the interlaced modules that enable system architects to generate robust synchronized conversions.

6 References

AN3896 — *MCF51EM256 Performance Assessment with Algorithms used in Metering Applications* by Paulo Knirsch

AN3827 — *Differences Between Controller Continuum ADC Modules,* AN3949—*ADC16 Calibration Procedure and programmable delay block synchronization* by Inga Harris

MC9S08GW64 Reference Manual

MC9S08GW64 Data sheet

AN4168SW — *Referenced Software*

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 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 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4168
Rev. 1
07/2010