AN8000.05
Application Note

Using the Sensing Machine, Setting the ZoomingADC™ Parameters, a sample application.

Table of Contents

1 INTRODUCTION

In this chapter you will find:
What is in this document and where it is
What you are supposed to have read to fully profit from this document
Where you will find additional information

This document is a guide to a successful start with the XE8000 Sensing Machines. It goes through the main aspects of the most specific peripherals and shows a short sample application.

The reader is expected to have some understanding of C-programming and to have read the SEMTECH technical note TN8000.16 "Coding with RIDE quick start" to fully exploit the sample application. The sample application also supposes that the reader has access to a ProStart for one of the Sensing Machines (we suggest the ProStart set made of the XE8000SW, the XE8000MP and the XE8000EV101) and a PC with a serial port and a terminal application.

Detailed product description and performances can be found in the corresponding datasheets. This document does not replace the datasheets and includes only partial information with the sole purpose of teaching "how it works and how it can be used".

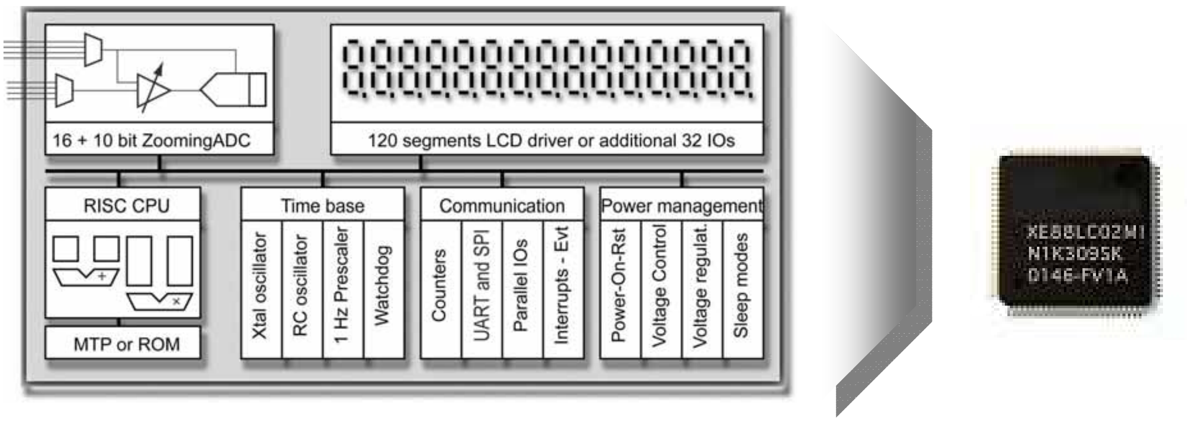
An Excel spreadsheet is also available for download on the Semtech web site, along with the sample application that puts together all the parameters of the ZoomingADC and helps the user to determine the very best settings for his application.

The rest of this document is in 4 parts: "The ADC", "The Zoom", "Advanced concepts" and "Calling the ZoomingADC in an XE8000 application".

The sample application will be briefly explained below so that the reader can use it to test his own understanding during the first chapters. It is explained again, in greater detail, in the last chapter.

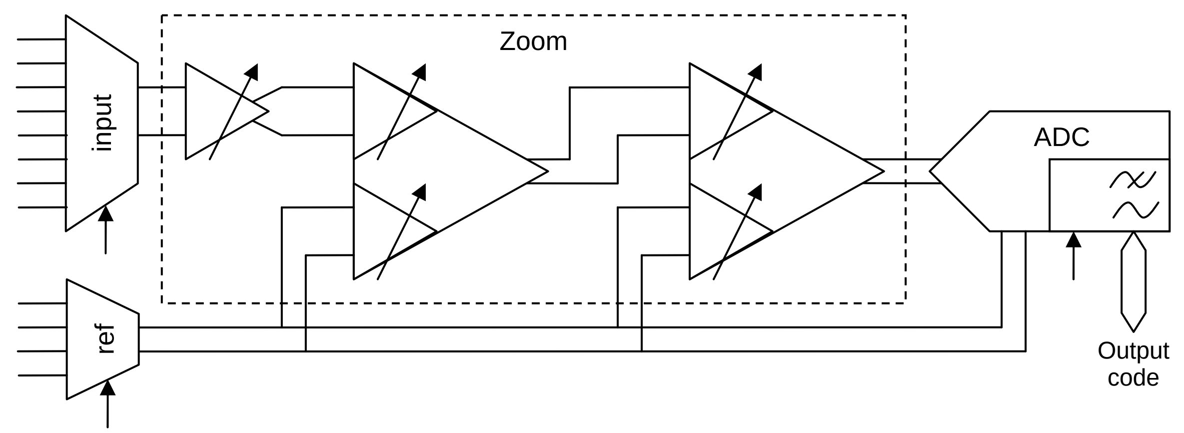
1.1 THE SENSING MACHINE AND THE ZOOMINGADC

The Sensing Machine is a system on chip (SoC) designed for sensor interfacing. Its main functional block is the ZoomingADC™ (see below). The Sensing Machine also embeds a comprehensive set of functional blocks to manage the complete acquisition process. This includes clock generators, a MCU core, non-volatile program memory, volatile data memory and communication peripherals.



The XE8802 Sensing Machine with LCD driver.

The ZoomingADC is built to enable the acquisition of several different signals. It is highly versatile, and its usage can be complex. Nevertheless some basics are quite simple and this document will help the user to get off to an easy start with the ZoomingADC.



The complete ZoomingADC™, with the input and reference selection, the Zoom (PGAs) and the ADC.

The ADC is the analog to digital converter itself, and it can be used without the Zoom for sufficiently large signals. The resolution of the ADC can be programmed. Using the ADC alone considerably simplifies the control of the block. The input and reference selection is described in the ADC chapter.

The Zoom is an extension to the ADC to make the acquisition of small signals, like the ones generated by Wheatstone bridges of piezoresistive sensors or magnetic sensors. The Zoom can also add or subtract a controlled offset from the input signal to accommodate sensors with large offsets. We suggest the reader first gets used to the ADC alone, and then studies the Zoom if required.

The last chapter will show and explain a very short XE8000 sample application that calls the ZoomingADC to make an acquisition and transfers the result to the UART port. It makes it possible to very easily test all features explained below, one by one, and see the result on a PC using a terminal. This application is not optimized for speed, power consumption or code size.

To exercise all information given below in the sample application, one just has to update the ZoomingADC settings (function InitADC) as shown in the following listing. The rest of the sample application will do the acquisition according to these settings and push the result to the UART.

```
/*******************************************************************
** --------------------General Parameters--------------------------**
** Default    : 0x00 - 0x01      g1     [1 reset to default     ] **
** Freq       : 0x00 - 0x03      g2     [1/4, 1/8, 1/32*RC, 8kHz] **
** Nelconv    : 0x00 - 0x03      g3     [1, 2, 4, 8 conversions ] **
** OSR        : 0x00 - 0x07      g4     [8,16,32,...,1024   OSR ] **
** Cont       : 0x00 - 0x01      g5     [continued,discontinued ] **
** Enable     : 0x00 - 0x0F      g6     [Pga3, Pga2, Pga1, Adc  ] **
** ---------------------Pga parameters------------------------- **
** Pga1Gain   : 0x00 - 0x01      p1     [1, 10                  ] **
** Pga2Gain   : 0x00 - 0x03      p2     [1, 2, 5, 10            ] **
** Pga3Gain   : 0x00 - 0x7F      p3     [0.083, ..., 10         ] **
** Pga2Off    : 0x00 - 0x0F      p4     [-1, -0.8, ..., 1       ] **
** Pga3Off    : 0x00 - 0x7F      p5     [-5.25,-5.16,...,5.25   ] **
** BiasAdc    : 0x00 - 0x03      p6     [25, 50 ,75 100% *Inom  ] **
** BiasPga    : 0x00 - 0x03      p7     [25, 50 ,75 100% *Inom  ] **
**---------------------Mux parameters---------------------------**
** AMuxMode  : 0x00 - 0x01       m1     [differential, common  ]  **
** AMuxSign  : 0x00 - 0x01       m2     [normal, inverted      ]  **
** AMuxChan  : 0x00 - 0x07       m3     [channel selection 1-8 ]  **
** VMuxChan  : 0x00 - 0x01       m4     [R1-R0, R3-R2          ]  **
*******************************************************************/
//      InitAdc parameters:
//      g1    g2    g3    g4    g5    g6
//      p1    p2    p3    p4    p5    p6    p7
//      m1    m2    m3    m4
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

The default UART settings in the sample application are: 38400 kbaud, 8 bits, 1 stop, no parity check. The default RC clock is 1.23 MHz (it is set in the UartInit() function).

1.2 NOTATION

Numbers will be written like in C language: 0x0A is a hexadecimal number, 10 is a decimal number. The following notation "0b00001010" will be used for binary numbers.

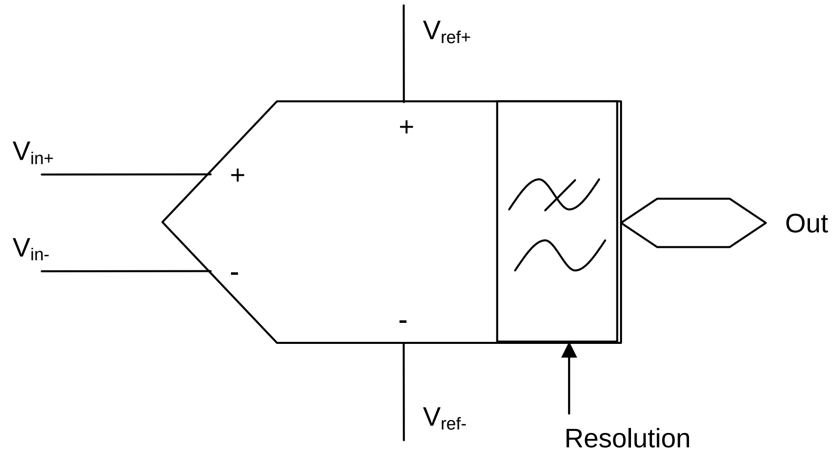
2 THE ADC

In this chapter, you will find:

- What the analog to digital converter (ADC) is
- How to select the right signal input and the right reference input
- How to select the resolution
- How much time will the conversion take

2.1 INTRODUCTION

The analog to digital converter (ADC) takes an input signal, compares it to a reference and issues a code proportional to the ratio of the signal over the reference. The input signal and the reference are differential voltages.



The analog to digital converter. It can be set for different resolutions and includes the low-pass filter.

In this converter, the output code is $Out \cong \frac{(V_{in1} - V_{in-})}{(V_{ref+} - V_{ref-})}$ (normalized on ± 0.5).

One can program the ADC resolution between 6 and 16 bits. The resolution setting influences the acquisition time.

If the signal at the input of the ADC is bigger than the maximum value, or smaller than the minimum value, the output code is saturated to the maximum or minimum value for the given resolution.

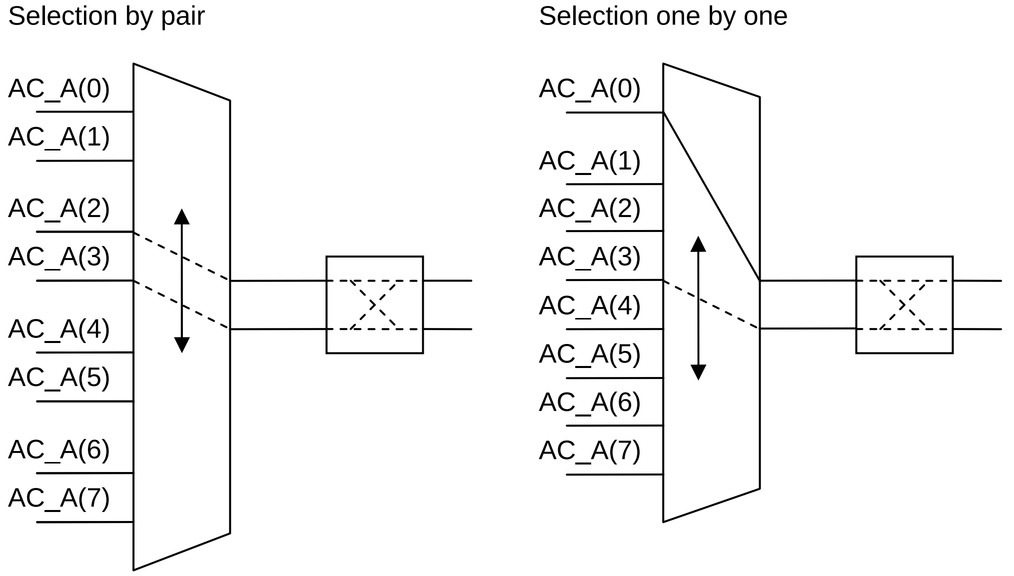
The signal at the input of the ADC and on the reference must be kept between the low supply voltage (V_{SS}) and the high supply voltage (V_{bat}).

2.2 SELECTING THE INPUT CHANNEL

The ZoomingADC has a multiplexer to select the input channel. The multiplexer has 8 input lines and 2 output lines. The output lines of the input multiplexer are connected to the ADC (or to the preamplifiers when using the Zoom). The two output lines of the input multiplexer carry the differential signal to the ADC.

The input lines can be selected in pairs or one by one. When a pair is selected, the two lines are connected to the two output lines as a differential signal. When the lines are selected one by one, line AC_A(0) has a special meaning and is always selected to be one line of the differential signal, the selected input being the other line of the differential signal.

In addition the positive and negative inputs of the differential signal can be inverted for offset cancellation.



Input channel selection, by pair or one by one

The input channel, the pair by pair or one by one selection mode as well as the differential signal inversion can be selected by the bits AMUX of the ZoomingADC register **RegAcCfg5**. A complete selection table is available in the datasheet. Default for the sample application is all AMUX bits to 0, which means AC_A(1) is the positive input V_{in+} and AC_A(0) is the negative input V_{in-}.

Note:

Selection by pair and selection one by one can be mixed to detect different types of signal. For 3 single ended signals and 2 differential signals, the single ended signals will be connected to AC_A(1), AC_A(2) and AC_A(3), AC_A(0) will be used as potential reference for these 3 inputs, and the differential signals will be connected to AC_A(5)-AC_A(4) and AC_A(7)-AC_A(6).

2.3 SELECTING THE REFERENCE CHANNEL

There are two differential reference channels for the ADC. They can be selected to measure two different sensors with different references.

The voltage on the selected reference must be positive. Reducing the voltage on the reference increases the input sensitivity.

Note:

The reference signal is used during the whole conversion. Any noise or variation of the reference input will deteriorate the conversion precision. One must handle the reference input with the same care as the signal input to have low noise, high resolution conversions.

The reference channel can be selected by the bit VMUX of the ZoomingADC register **RegAcCfg5**. A complete table is available in the datasheet. Default for the sample application is VMUX set to 0, which means AC_R(1) is the positive reference input V_{ref+} and AC_R(0) is the negative reference input V_{ref-}.

2.4 OUTPUT FORMAT

The output code is always a 16-bit signed number in complement to 2 (even when the resolution is smaller than 16 bits). When the conversion resolution is smaller than 16 bits, the bit following the LSB is always 1, and the next bits are always 0. The code 0x0000 does not exist when the converter operates with less than 16 bits resolution.

Example: Different input values and the corresponding output codes for 3 resolutions

V_{in+}	V_{in-}	V_{ref+}	V_{ref-}	Out (8 bits conversion)	Out (12 bits conversion)	Out (16 bits conversion)
			comments	*1	*2	*3
0.5 V	0.0 V	1 V	0 V	0x7F80 (32640)	0x7FF8 (32760)	0x7FFF (32767)
0.5 V	0.5 V	1 V	0 V	0x0080 (128) or 0x8F80 (-128)	0x0008 (8) or 0x8FF8 (-8)	0x0000 (0) or 0x8FFF (-1)
0.5 V	1.0 V	1 V	0 V	0x8080 (-32640)	0x8008 (-32760)	0x8000 (-32768)
0.0 V	0.5 V	1 V	0 V	0x8080 (-32640)	0x8008 (-32760)	0x8000 (-32768)

*1 ZoomingADC settings are:
```
InitAdc(0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```
*2 ZoomingADC settings are:
```
InitAdc(0x00, 0x00, 0x00, 0x03, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```
*3 ZoomingADC settings are:
```
InitAdc(0x00, 0x00, 0x00, 0x05, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

Notes:

The input 0 is at equal distance from the highest negative code and the lowest positive code (there is an even number of steps, therefore no step is exactly equivalent to the input 0 and the error will always be ± 0.5 LSB).

If one could have a converter with a 2 bits output on a ±1 V range, this would mean that there would be 4 codes: (10), (11), (00) and (01).
(10) for Vin < -0.5 V, (11) for -0.5 V < Vin < 0, (00) for 0 < Vin < +0.5 V and (01) for Vin > +0.5 V. For such an converter, Vin = 0 V would be alternating between (11) and (00).

2.5 SETTING THE RESOLUTION OF THE ADC

The ADC of the ZoomingADC is an oversampled converter. This means that it uses several input samples to generate one output code (or output sample). The resolution that is achieved in the output code depends on the number of input samples used to generate this output code. This resolution can be increased by realizing several conversions per acquisition, this will be explained in a next chapter.

The number of input samples used to generate one output sample (also named "over-sampling ratio" or OSR), and therefore the resolution, is defined by the ADC parameter SET_OSR in register **`RegAcCfg0`**.

$$OSR = 2^{3+SET\_OSR}$$

$$Resolution_{ADC} = \min(6 + 2 \cdot SET\_OSR, 16) \text{ in bits}$$

See example above for 8 bit, 12 bit and 16 bits resolution simple conversions.

2.6 TIME REQUIRED TO REALIZE A CONVERSION

A conversion is made of several input samples, sampled at a given frequency fs. Therefore the time required for one conversion (T_{con}) depends on fs and the number of input samples required to make the conversion, plus some extra set-up time.

> **Note:**
>
> The total time required for a conversion may also depend on extra features like preamplification and number of conversions per sample (see below)

fs is set by the bits FIN of **`RegAcCfg2`**. Default value for FIN is 0, which means that fs is the RC frequency divided by 4 in the XE8801 and XE8805 or the RC frequency divided by 8 in the XE8802. Maximal fs in any case is 512 kHz.

> **Note:**
>
> To reach the maximum acquisition frequency on the XE8802, one can set the RC oscillator to 4 MHz, and divide the CPU frequency by a factor of 2 (instruction FREQ).

All timings given here will be for fs = 512 kHz unless otherwise mentioned. The default fs in the sample application when loaded on an XE8801 or an XE8805 is 300 kHz, it is 150 kHz when loaded on an XE8802.

$T_{con} = (OSR + 2) / fs$ for a conversion without amplification or multiple conversions per samples (bits SET_NELC are 0).

Example: Time required making different simple conversions

resolution	Required OSR	SET_OSR	Time at fs=512 kHz	Comments
8 bits	16	0b001	35.2 µs	*1
12 bits	64	0b011	129 µs	*2
16 bits	256	0b101	504 µs	*3

*1 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

*2 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x03, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

*3 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x05, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

2.7 LOWERING NOISE AND ELIMINATING ACQUISITION CHAIN OFFSET BY INCREASING THE NUMBER OF CONVERSIONS PER ACQUISITION

This is a more advanced concept that can improve the results of the normal conversion as described above. It is the connection of several conversions (as described above) into one output sample (one acquisition). The ZoomingADC is able to take several such conversions one after each other, inverting the input signal sign between each of these samples. The acquisition result is the average of all conversions. This reduces offsets from the acquisition path and further filters the input signal.

This multiple conversion and filtering process also increases the resolution of the acquisition, and increases the time required for a complete acquisition.

The number of conversions per acquisition (NELCONV) is set by the bits SET_NELC is the register **RegAcCfg0**. Default value is 0 for this document (one conversion per acquisition).

$$NELCONV = 2^{SET\_NELC}$$

The achieved resolution is now

$$Resolution_{ADC} = \min(6 + 2 \cdot SET\_OSR + SET\_NELC, 16) \text{ in bits.}$$

The time required for an acquisition is now

$$T_{con} = (NELCONV(OSR + 1) + 1) / fs$$

Note:

Increasing NELCONV does not increase the resolution as fast as increasing OSR.

NELCONV is combined with a first order filter that can be helpful to reduce the sensor noise and the input noise.

Using NELCONV=2 or above rejects most of the acquisition path offset and offset drift.

Example: Time required making acquisitions with multiple conversion compared with acquisitions with a single conversion. Sample application has a 300 kHz fs. Maximal fs is 512 kHz.

resolution	Required OSR & NELCONV	SET_OSR	SET_NELC	Offset cancel	Time at fs = 300 kHz	Time at fs = 512 kHz	Notes
12 bits	64 & 1	0b011	0b00	No	220 µs	129 µs	*1
13 bits	64 & 2	0b011	0b01	Yes	437 µs	256 µs	*2
14 bits	64 & 4	0b011	0b10	Yes	870 µs	510 µs	*3
14 bits	128 & 1	0b100	0b00	No	433 µs	254 µs	*4

*1 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x03, 0x00, 0x01,
         0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
         0x00, 0x00, 0x00, 0x00);
```

*2 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x01, 0x03, 0x00, 0x01,
         0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
         0x00, 0x00, 0x00, 0x00);
```

*3 ZoomingADC settings are:

```
InitAdc(0x00, 0x00, 0x02, 0x03, 0x00, 0x01,
         0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
         0x00, 0x00, 0x00, 0x00);
```

*4 ZoomingADC settings are:

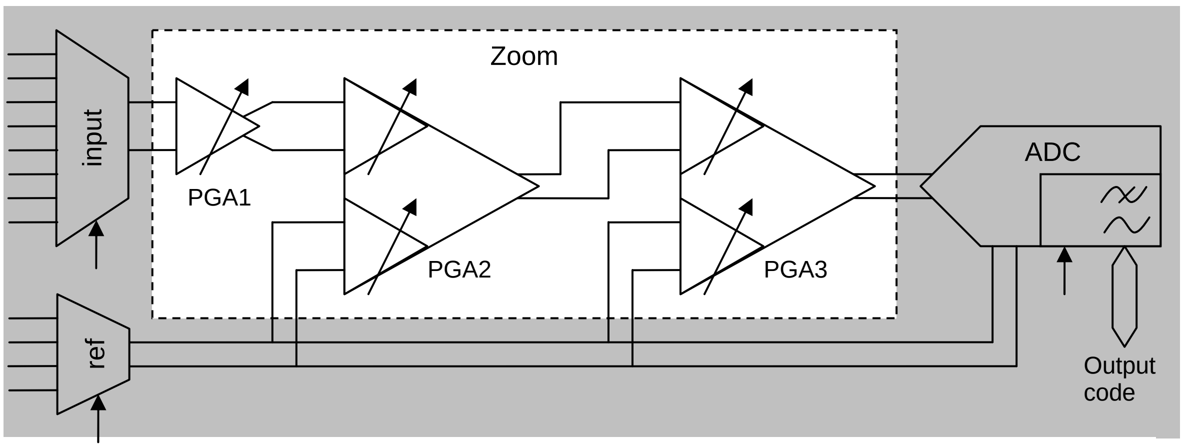
```
InitAdc(0x00, 0x00, 0x00, 0x04, 0x00, 0x01,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

3 THE ZOOM

In this chapter, you will find:
What the Zoom function of the ZoomingADC is
How to select the gain and the offset of each PGA
How much time it will add to the conversion

3.1 INTRODUCTION

The Zoom is a series of 3 PGAs, one with one programmable differential input, and two with two programmable differential inputs. When a PGA is not used, it is by-passed.



The three PGAs that form the Zoom.

The PGA2 and PGA3 have two gain settings. The gain of the PGA on its input that is connected to the previous PGA or to the input selector is named Pga(i)Gain and it is used to set the gain of the acquisition path. The gain of the PGA on its input that is connected to the reference selector is named Pga(i)Offset and it is used to add a controlled offset to the signal path. A PGA that is not used can be by-passed, adding no noise or current requirement to the system. If only one PGA is used, it should be PGA3 that is the most versatile and has the highest linearity. Therefore the explanations below will start with PGA3.

3.2 USING PGA3

PGA 3 can be set for gains between 1/12 and 10. The output of PGA3 is directly connected to the input of the ADC.

Note:

As the output of the PGA3 is directly connected to the input of the ADC, its signal range must pass in the input range of the ADC, that is roughly ±0.5 x Vref.

The gain of the PGA3 is set by the 7 bits Pga3Gain of register **`RegAcCfg3`**. The gain is proportional to the value of the register, the LSB value being 1/12. PGA3 has a gain of 1 for Pga3Gain = 12 = 0x0C.

Note:

Input signal and reference signal are differential. All gains and signal values are differences of potentials and the input signal can be positive or negative.

Example: Gain setting for the PGA3, no offset compensation.

AC_A(1) range	AC_A(0) range	AC_R(1)	AC_R(0)	Pga3 Gain	PGA3 out range	ADC code range	
0.0 V to 1.0 V	0.5 V	1.0 V	0.0 V	1	- 0.5 V to + 0.5 V	0x8000 0x7FFF	*1
0.0 V to 0.5 V	0.0 V	1.0 V	0.0 V	1	0.0 V to + 0.5 V	0x0000 0x7FFF	*1,*3
0.0 V to 1.0 V	0.0 V	1.0 V	0.0 V	0.5	0.0 V to + 0.5 V	0x0000 0x7FFF	*2,*3
0.0 V to 1.0 V	0.0 V to 1.0 V	1.0 V	0.0 V	0.5	- 0.5 V to + 0.5 V	0x8000 0x7FFF	*2

*1 ZoomingADC settings are (PGA3 gain in bold):

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x09,
        0x00, 0x00, 0x0C, 0x00, 0x00, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

*2 ZoomingADC settings are (PGA3 gain in bold):

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x09,
        0x00, 0x00, 0x06, 0x00, 0x00, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

*3 ADC input range is not completely used !

As can be seen from the above example, for some input ranges, it is not possible to use the full input range of the ADC using only gain settings because some values are never reached. This could be compensated for by changing the potential offset of the input signal, but this may require external elements. The ZoomingADC has another mechanism included to compensate for offsets of the input signal: this is the offset input of the PGAs. This is a second programmable input that is connected to the reference voltage. This input can be programmed in gain and sign, so that an offset can be added to or subtracted from the input signal.

The offset amplifier of PGA3 is controlled by the 7 bits Pga3Off of register **RegAcCfg4**. These bits multiply Vref/12 and subtract it from the amplified input signal. Offset setting can be positive or negative.

Example: Gain setting for the PGA3, with offset compensation.

AC_A(1) range	AC_A(0) range	AC_R(1)	AC_R(0)	Pga3 Gain	Pga3 Offset	PGA3 out range	ADC code range	
0.0 V to 0.5 V	0.0 V	1.0 V	0.0 V	2 (=24/12)	+0.5 (=6/12)	-0.5 V to +0.5 V	0x8000 0x7FFF	*1
0.0 V to 1.0 V	0.0 V	1.0 V	0.0 V	1 (=12/12)	+0.5 (=6/12)	-0.5 V to +0.5 V	0x8000 0x7FFF	*2

*1 ZoomingADC settings are (PGA3 gain and offset in bold):

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x09,
        0x00, 0x00, 0x18, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

*2 ZoomingADC settings are (PGA3 gain and offset in bold): :

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x09,
        0x00, 0x00, 0x0C, 0x00, 0x06, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

Note:

The output range of the PGA3 must fit within the input range of the ADC. But the individual outputs of the gain or offset amplifiers of the PGA3 do not need to be within a given range as they do not really physically exist.

3.3 USING PGA1 AND PGA2

If the gain obtained by the PGA3 is not sufficient to have an input signal that covers the full input range of the ADC, one can increase it further by using PGA2 (gain 1, 2, 5 or 10) and/or PGA1 (gain 1 or 10). The output of an amplifier is quite easy to compute (all voltages are differential, Vin and Vout can be positive or negative, Vref is positive):

Vout(PGA1) = Vin(PGA1) * Pga1Gain

Vout(PGA2) = Vin(PGA2) * Pga2Gain - Vref * Pga2Offset

Vout(PGA3) = Vin(PGA3) * Pga3Gain - Vref * Pga3Offset

Notes:

The (differential) output range of any PGA **must** fit within ± Vbat. Output range of PGA1 should be within ±Vbat/5 and output range of PGA2 should be within ±Vbat/2 otherwise linearity will be strongly degraded. The (differential) output range of PGA3 must fit within ± Vref/2. Otherwise part of the signal will saturate the ADC.

For a good linearity, a maximal gain must be applied to the last amplification stages, reducing the output amplitude of the first gain stages.

3.4 CHOOSING THE RIGHT PARAMETERS FOR PGA1, PGA2 AND PGA3

There are a few rules to set the correct Zoom parameters. The goal is to map the input signal range on the full input range of the ADC, so that a maximum of its resolution can be exploited.

1) Put the gain at the end of the amplification chain.

2) Check the output voltage range of **each** PGA for absolute min-max of the input signal. If PGA2 saturates and PGA3 gain is low, saturation will occur below the min-max values of the ADC!

3) Check the output voltage of PGA3 versus the input range of the ADC for absolute min-max of the input signal **and** of the reference signal.

4) Keep some margin for the absolute precision of the parameters (15% if not offset cancellation, 25% if offset cancellation).

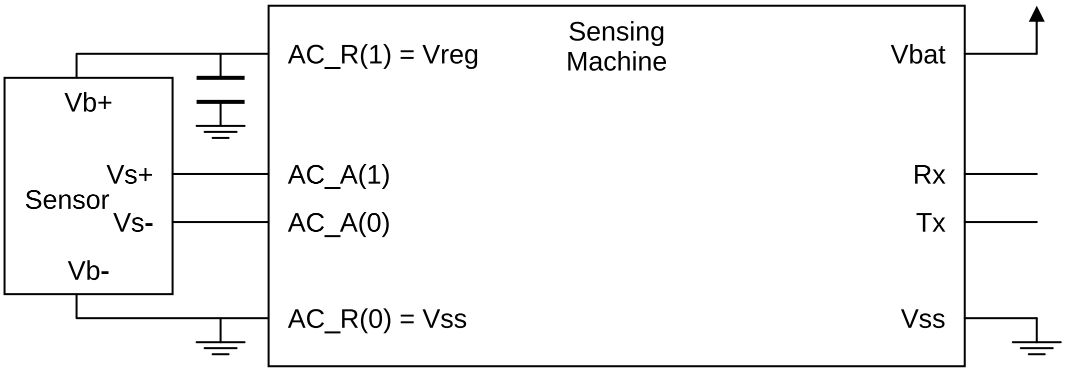
5) Keep some margin for the temperature drift (depends on temperature range and on its effect on the input signal and on the reference).

6) If the output of the ADC does not cover the full 0x8000 to 0x7FFF as planned (deduction made of the margins), check again points 2 and 3.

7) If linearity seems to be a problem, check again point 1, or check the input signal's impedance.

8) For noise improvements, one can check the reference stability and increase OSR and NELCONV.

Example: PGAs settings for a pressure sensor.



The whole system is supplied between V_{bat} and V_{ss}. It communicates via the UART on lines Tx and Rx.

The pressure sensor is directly connected to the Sensing Machine. In this case, it is supplied by the Vreg voltage (see Advanced Concept chapter for more information on this trick). The capacitor on Vreg is the necessary capacitor of the voltage regulator (1 µF). Vreg may vary from chip to chip and with temperature, for this example we assume Vreg = 2.1 V.

The sensor is biased by the voltage between V_{b+} and V_{b-}. Its signal is read between V_{s+} and V_{s-}.

The sensor has the following characteristics:

Sensitivity = 15 mV/V /bar (or, under 2.1 V bias, 0 V at 0 bar, 0.015 * 2.1 = 31.5 mV at 1 bar)

We will compute the PGAs settings for two types of sensors: one absolute going from vacuum to 1.2 bar, and the other barometric going from 0.8 to 1.2 bar, maximal gain being on the last PGA (0 mean PGA is disabled).

Gain is set to maximum without saturation (signal amplitude at the input of the ADC within ±Vref/2).

Type	Minimal input	Maximal input	Input range	Gain	Gain - 25%	Gain settings PGA1, PGA2, PGA3
Absolute	0 mV/V	18 mV/V	18 mV/V	55.5	41.6	0, 5, 8.3 = 41.5 -> 0b0, 0b10, 100
Barometric	12 mV/V	18 mV/V	6 mV/V	167	125	10, 2, 6.2 = 124 -> 0b1, 0b01, 74

We will now compute the offset cancellation values (keep the mean output of each amplifier as near to 0 as possible).

Type	PGA1 mean input	PGA2 mean input	PGA2 gain output	PGA2 Offset Setting	PGA3 mean input	PGA3 gain output	PGA3 Offset Setting
Absolute	disabled	9 mV/V	45 mV/V	0 V/V -> 0b000	45 mV/V	373 mV/V -> 4	+0.33 V/V
Barometric	15 mV/V	150 mV/V	300 mV/V	+ 0.2 V/V -> 0b001	100 mV/V	620 mV/V -> 7	+0.58 V/V

ZoomingADC for absolute sensor settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x0d,
        0x00, 0x02, 0x6e, 0x00, 0x04, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

ZoomingADC for barometric sensor settings are:

```
InitAdc(0x00, 0x00, 0x00, 0x07, 0x00, 0x0f,
        0x01, 0x01, 0x4a, 0x01, 0x07, 0x03, 0x03,
        0x00, 0x00, 0x00, 0x00);
```

3.5 PGA SETTLING TIME

The PGAs are differential amplifiers; therefore they have to manage their own common mode output signal. This is done with a low frequency feedback loop that requires time to stabilize. As the precision of the required stabilization depends on the resolution that one wants to reach, the settling time of the PGA depends on the ADC resolution. The settling time (in number of periods of fs) is equal to the oversampling ratio (OSR). Therefore when using the PGAs, a delay must be placed in the software between the last access to one of the **RegAcCfg1-5** register and the triggering of the ZoomingADC start. And this delay must be equal to (OSR * the period of fs).

Notes:

No such delay is required when the PGAs are off.

No such delay is required between two samples when the ZoomingADC settings are not modified (no write to registers **RegAcCfg1-5**) between these samples.

By making a fake conversion with all registers set as required by the real conversion but SET_NELC = 0x00, waiting for the fake conversion completion, one creates a delay of the correct duration.

4 ADVANCED CONCEPTS

In this chapter, you will find:
The ZoomingADC impedance
Low power operation
Ratiometric measurement

4.1 INTRODUCTION

This chapter introduced some more advanced concepts related to the ZoomingADC. Most of these are not necessary for normal usage of the ZoomingADC, but they may help the reader to get the most from their Sensing Machine.

4.2 THE ZOOMINGADC IMPEDANCE

The ZoomingADC is a switched capacitor based block. This means that it does not use resistors to fix gains, but capacitors and switches. This has important implications on the nature of the input impedance of the block.

Basically, a switched capacitor is a way to emulate a resistor by using a capacitor. The capacitors are much easier to realize on CMOS technologies and they show a very good matching precision.

The switched capacitor principle

A resistor is characterized by the current that flows through it (positive current leaves node V_1):

$I = R/ (V_1 - V_2)$

One can verify that the mean current leaving node V1 with a capacitor switched at frequency f is:

$\langle I \rangle = (C/f) / (V_1 - V_2)$

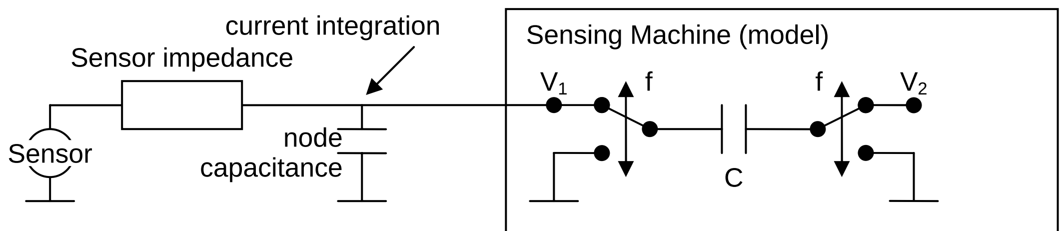
Therefore as a mean value, the switched capacitor C/f is equivalent to a resistor.

It is important to consider that this is only a mean value. If the current is not integrated (low impedance source), the impedance is infinite during the whole time but the transition.

What does it mean for the ZoomingADC?

If the fs clock is reduced, the mean impedance is increased. By dividing the fs clock by a factor 10, the impedance is increased by a factor 10.

One can reduce the capacitor that is switched by using an amplifier set to its minimal gain. In particular if PGA1 is used with gain 1, its mean impedance is 10x bigger than when it is used with gain 10.



The switched capacitor principle

One can increase the effective impedance by increasing the electrical bandwidth of the sensor node so that the switching current is absorbed through the sensor before the switching period is over. Measuring the sensor node will show short voltage spikes at the frequency fs, but these will not influence the measurement. Whereas if the bandwidth of the node is lower, no spikes will arise, but a small offset can be generated by the integration of the charges generated by the switched capacitors, this corresponds to the mean impedance effect.

Note:

One can increase the mean input impedance of the ZoomingADC by lowering the acquisition clock fs.

One can increase the mean input impedance of the ZoomingADC by decreasing the gain of the first enabled amplifier.

One can increase the effective input impedance of the ZoomingADC by having a source with a high electrical bandwidth (sensor electrical bandwidth much higher than fs).

4.3 REDUCING THE CURRENT OF THE ADC AND OF THE PGAS

The ADC and the PGAs include amplifiers which bias current is set so that they have a sufficient bandwidth to achieve the full resolution of the ADC at its maximal speed.

When the ADC is used at a lower speed (reducing fs), the current in the ADC and in the PGA can be reduced by the same amount.

One can also notice that it is good practice to disable any unused PGA. Enabling and setting the bias current is made with the bits Enable, BiasPga and BiasAdc of the register **`RegAcCfg1`**.

When the ZoomingADC is not used, it should be turned off.

4.4 RATIOMETRIC MEASUREMENTS

The principle of the ZoomingADC is ratiometric. This means that the whole measurement is proportional to the reference voltage. Therefore one could put a ratiometric sensor in front of the ZoomingADC, and use its bias voltage as a reference voltage for the ZoomingADC and the result should not depend on the applied bias. This is globally true, but the linearity of the gain with respect to the reference voltage is not as high as the input signal linearity of the ZoomingADC, therefore for maximal precision one should operate at fixed reference voltage.

One relatively fixed voltage that is available on the Sensing Machine is the regulated voltage Vreg. It is not precise but its variation with the voltage supply variation is low, and therefore this voltage is much more stable than the supply voltage. It can be used directly to bias highly resistive loads or it can be buffered (using one of the DAC buffers of the XE8805 Sensing Machine as an example) to bias other loads.

Check the datasheet for numbers on the ZoomingADC PSRR (Power Supply Rejection Ratio).

5 CALLING THE ZOOMINGADC IN AN XE8000 APPLICATION

In this chapter, you will find:
How to build a simple application using the ZoomingADC
The initialization of such an application
Some other relevant software tricks

5.1 INTRODUCTION

A very basic sample application is proposed, this is an application that can be downloaded from the SEMTECH web site. It makes acquisitions with the settings chosen by the user and outputs the acquisition results to the UART so that one can read the result with any terminal application.

This application can be used to test the concepts proposed in this booklet or as a base for a first complete acquisition application.

See the SEMTECH technical note TN8000.16 for explanations on the usage of the ProStart and of the RIDE development environment. Reference code is compiled at optimization level 1.

5.2 INSTALLING THE SAMPLE APPLICATION

To install and run the sample application we suggest using the following procedure:

1) RIDE development environment for XE8000 must be installed. The RIDE application should not be open.

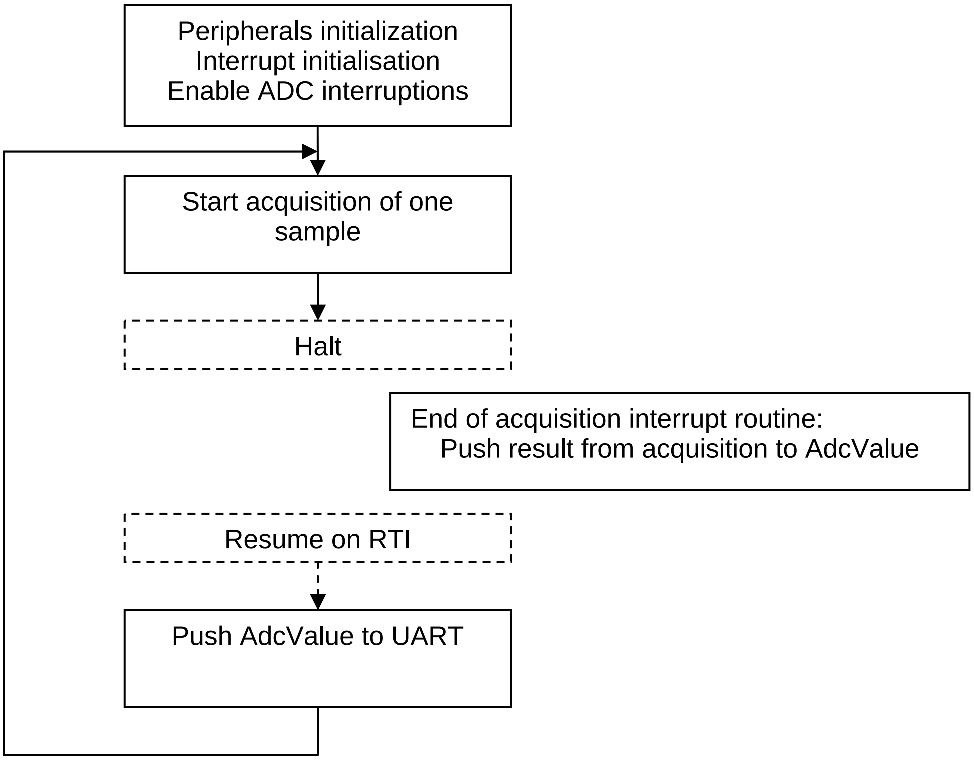
2) Download the "ZoomingADCDemo.zip" file that contains the sample application from SEMTECH web site.

3) Decompress it (option "Use folder name" must be set). Now your project files are in the "ZoomingADCDemo" folder, the source files are in "ZoomingADCDemo/Src" folder and the documentation files are in the "ZoomingADCDemo/Doc" folder (including the spreadsheets to help evaluate the ZoomingADC settings).

4) Double click on the file "zoomingadc demo.prj" in the folder "ZoomingADCDemo". The project is now ready to be compiled or downloaded to the ProStart (see RIDE manuals to know how to do that). For simulation, the project has to be modified as the code is calling real time functions that can take a long time to simulate (crystal oscillator start-up).

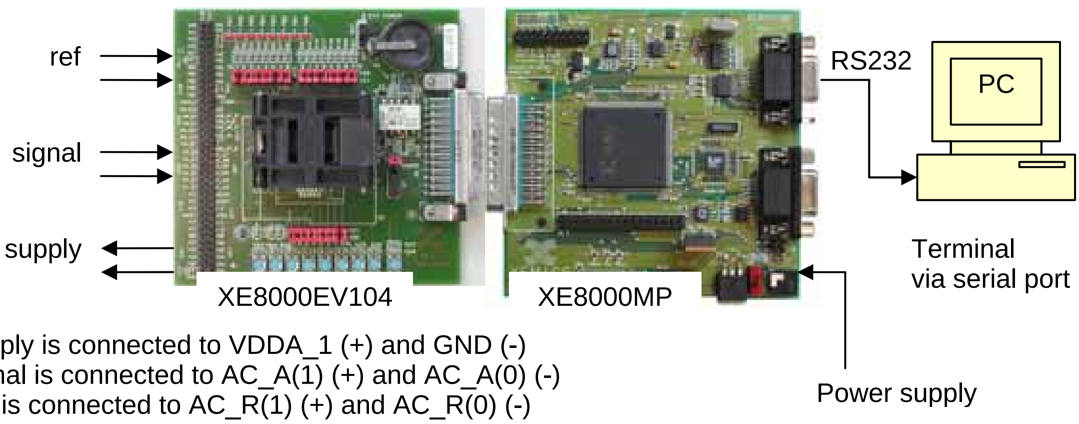
5.3 ZOOMINGADC SAMPLE APPLICATION PRINCIPLE.

The sample application first initializes all peripherals, then programs the settings of the ZoomingADC and starts an acquisition. It waits for the result, sends it the UART, and starts a new conversion.



Acquisition program working on interrupts

5.4 SETTINGS FOR THE SAMPLE APPLICATION



Supply is connected to VDDA_1 (+) and GND (-)
Signal is connected to AC_A(1) (+) and AC_A(0) (-)
Ref is connected to AC_R(1) (+) and AC_R(0) (-)

The PC will receive the information that has been read by the ZoomingADC. To do that one can use a terminal application like HyperTerminal. The terminal settings must be 38400 bauds, 8-bits ASCII, no parity, and 1 stop bit.

The sample application will send the value it has read in ASCII format to the terminal via the UART of the XE8000MP.

5.5 INITIALIZATION

The initialization is made by calling the initialization routines (all initialization routines are in "Initialisation.c"):

```
        InitPortA();                      // Initializes PortA
        InitPortB();                      // Initializes PortB
        InitPortC();                      // Initializes PortC
        InitUART(38400, 1, 0, 0, 2);      // Initialises the UART communications
                                          // InitUART will set the RC to 1.23 MHz
        InitCounters();                   // Initializes Counters
```

Port A is set as input with pull-ups.

Port B and C are set as outputs.

RC oscillator is set in the UART initialization routine.

Counters are set to 0.

The UART is slightly more complex as it is directly set to its operating value. In the sample application, one sets it to 38 kbaud, 8 bits, no parity. This routine will set the RC oscillator frequency to 1.23 MHz. It needs the Xtal oscillator to precisely set the RC oscillator.

Note:

The 32 kHz crystal resonator must be connected to the chip for the InitUART function to operate.

The ZoomingADC initialization will be discussed below.

When the initialization is done, one enables the interrupts:

```
        RegIrqEnHig |= 0x80;  // Enable interrupts
```

5.6 ZOOMINGADC INITIALIZATION

The ZoomingADC initialization routine takes all parameters for the ZoomingADC and puts them in the right bit of the right registers.

One calls the ZoomingAdcInit with the following parameters:

```
/*********************************************************************
** -------------------General Parameters-------------------------**
** Default    : 0x00 - 0x01      g1     [1 reset to default     ] **
** Freq       : 0x00 - 0x03      g2     [1/4, 1/8, 1/32*RC, 8kHz] **
** Nelconv    : 0x00 - 0x03      g3     [1, 2, 4, 8 conversions ] **
** OSR        : 0x00 - 0x07      g4     [8,16,32,...,1024   OSR ] **
** Cont       : 0x00 - 0x01      g5     [continued,discontinued ] **
** Enable     : 0x00 - 0x0F      g6     [Pga3, Pga2, Pga1, Adc  ] **
** --------------------Pga parameters------------------------- **
** Pga1Gain   : 0x00 - 0x01      p1     [1, 10                  ] **
** Pga2Gain   : 0x00 - 0x03      p2     [1, 2, 5, 10            ] **
** Pga3Gain   : 0x00 - 0x7F      p3     [0.083, ..., 10         ] **
** Pga2Off    : 0x00 - 0x0F      p4     [-1, -0.8, ..., 1       ] **
** Pga3Off    : 0x00 - 0x7F      p5     [-5.25,-5.16,...,5.25   ] **
** BiasAdc    : 0x00 - 0x03      p6     [25, 50 ,75 100% *Inom  ] **
** BiasPga    : 0x00 - 0x03      p7     [25, 50 ,75 100% *Inom  ] **
**---------------------Mux parameters--------------------------**
** AMuxMode  : 0x00 - 0x01      m1     [differential, common  ]  **
** AMuxSign  : 0x00 - 0x01      m2     [normal, inverted      ]  **
** AMuxChan  : 0x00 - 0x07      m3     [channel selection 1-8 ]  **
** VMuxChan  : 0x00 - 0x01      m4     [R1-R0, R3-R2          ]  **
*********************************************************************/
//      g1 g2 g3 g4    g5 g6 p1 p2 p3    p4 p5    p6 p7 m1 m2 m3 m4
InitAdc(0, 0, 0, 0x07, 0, 1, 0, 0, 0x0C, 0, 0x06, 3, 3, 0, 0, 0, 0);
```

No check is made on the parameters, they are just shifted to their correct location and masked.

Internally, the routine is the following (_U8 is an unsigned 8 bit value):

```
_U8 InitAdc(_U8 Default, _U8 Freq, _U8 Nelconv, _U8 Osr, _U8 Cont,
            _U8 Enable, _U8 Pga1Gain, _U8 Pga2Gain, _U8 Pga3Gain,
            _U8 Pga2Off, _U8 Pga3Off, _U8 BiasAdc, _U8 BiasPga,
            _U8 AMuxMode, _U8 AMuxSign, _U8 AMuxChan, _U8 VMuxChan ){

RegAcCfg0 = (((Nelconv << 5 ) & 0x60) | ((Osr << 2) & 0x1C) | ((Cont << 1) & 0x02));
RegAcCfg1 = (((BiasAdc << 6) & 0xC0) | ((BiasPga << 4 ) & 0x30) | (Enable & 0x0F));
RegAcCfg2 = (((Freq << 6) & 0xC0) | ((Pga2Gain << 4) & 0x30 ) | (Pga2Off & 0x0F));
RegAcCfg3 = (((Pga1Gain << 7) & 0x80) | (Pga3Gain & 0x7F));
RegAcCfg4 = (Pga3Off & 0x7F);
RegAcCfg5 = (((Default << 6) & 0x40) | ((AMuxMode  << 5) & 0x20)|
        ((AMuxSign << 4) & 0x10) |((AMuxChan << 1) & 0x0E) | (VMuxChan & 0x01));
} // InitAdc()
```

5.7 ZOOMINGADC ROUTINES

A few routines have been written to control directly the ZoomingADC. The main one is the InitAdc() described in the previous chapter.

In addition routines to stop the ZoomingADC (go to low power) and read the results from the ADC are proposed here.

The first one triggers the ZoomingADC. It starts its conversion according to the settings used in InitAdc().

```
void Start_ZoomingAdc (void){
        RegAcCfg0 |= 0x80;               // Start by setting start bit
}
```

The second function stops the ZoomingADC and sets it in its lowest power mode. One uses InitAdc() to reset the ZoomingADC for a new conversion.

```
void Stop_ZoomingAdc (void){
// Disables the ZoomingADC, the 3 Pga's and reduce the bias to the min
        RegAcCfg1 &= 0xFF;
}
```

The third function reads the result of the conversion and returns it as an integer. Type _S16 is a signed integer on 16 bits. Type T_S16_U8 is a hybrid that can be used as an integer or a set of two bytes. It is used to read the peripheral result and return it as an integer.

```
_S16 R_RegAcOut(void){
        T_S16_U8 v;              // use T_S16_U8 type to order the Msb and Lsb
        v.U8.Msb = RegAcOutMsb;   // copy byte per byte
        v.U8.Lsb = RegAcOutLsb;
        return v.S16;              // return signed result
}
```

The last function just waits for the busy bit of the ZoomingADC to come back to its low value (conversion is finished). This is used when the software is not operating on interrupts. Be aware that the busy bit is only set after one cycle of the ZoomingADC clock (at frequency fs), to that if one checks the busy bit just after triggering the ZoomingADC to start a new conversion; it will not be high yet. This function is not used when using the interrupt to detect the end of conversion.

```
void PoolingADC(void){
        while(RegAcCfg5 & 0x80);
}
```

5.8 UART ROUTINES

The UART control routines are in the UartHandler.c file. These are not optimized for speed but for readability.

Note:

The UartInit() UART initialization routine "Initialize.c" has been partially disabled (no baudrate selection anymore) to limit the application code size. It can be enabled again by changing the "/** /" comment starter at the begin of the switch statement to "/**/" (erase the space between the second "*" and the second "/").

5.9 COMPLETE SAMPLE APPLICATION, MAIN CODE

```
        InitPortA();                       // Initializes PortA
        InitPortB();                       // Initializes PortB
        InitPortC();                       // Initializes PortC
        InitUART(38400, 1, 0, 0, 2);       // Initialises the UART communications
                                           // InitUART will set RC to 1.2 MHz
        InitCounters();                    // Initializes Counters

        InitAdc(0, 0, 2, 7, 0, 1, 0, 0, 0x0C, 0, 0x06, 3, 3, 0, 0, 3, 0  );

        RegIrqEnHig |= 0x80;  // Enable interrupts

        while(1){

             Start_ZoomingAdc(); // trigger the start bit of the ZoomingADC

             asm("halt");   // wait for next interrupt
                              // the return from acquisition will be handled
                              // by the ZoomingADC IRQ handler

             SendString("Out: ");  // send value to UART
             SendInt(AdcValue);
             SendString("\r\n");
        }   // while(1)
}  // main()
```

6 APPENDIX

In this chapter, you will find:

- Useful tricks and tips

6.1 OPERATING THE SENSING MACHINE ON THE PROSTART AT DIFFERENT VOLTAGES

The XE8000MP board of the ProStart has an on-board voltage regulator. The voltage generated by this regulator is the Vbat of the Sensing Machine and it is available on pin VDDA_1 of the ProStart.

The user can select this voltage with in RIDE with Options->Debug->Other tools->Advanced options. This selection lets the user choose between 2.4 V, 3.0 V and 5.0 V operations. The user can also use a DOS level command "prostartinterface 2 com1 -s 2.4" where com1 is the PC port name and 2.4 is the selected voltage. With this command, the voltage can be set in steps of 0.1 V from 1.0 V to 5.5 V. The precision of this regulator is not very high. For precision measurements, it is suggested that the user measures the generated voltage to have its exact value.

In both cases (voltage selection through RIDE or through the DOS level command), the XE8000MP must be connected to the PC com port via its PROG RS-232 connector.

6.2 HEX - DEC - BIN TABLE

Hex	Dec	Bin	Hex	Dec	Bin	Hex	Dec	Bin	Hex	Dec	Bin
0x00	0	0000 0000	0x40	64	0100 0000	0x80	128	1000 0000	0xC0	192	1100 0000
0x01	1	0000 0001	0x41	65	0100 0001	0x81	129	1000 0001	0xC1	193	1100 0001
0x02	2	0000 0010	0x42	66	0100 0010	0x82	130	1000 0010	0xC2	194	1100 0010
0x03	3	0000 0011	0x43	67	0100 0011	0x83	131	1000 0011	0xC3	195	1100 0011
0x04	4	0000 0100	0x44	68	0100 0100	0x84	132	1000 0100	0xC4	196	1100 0100
0x05	5	0000 0101	0x45	69	0100 0101	0x85	133	1000 0101	0xC5	197	1100 0101
0x06	6	0000 0110	0x46	70	0100 0110	0x86	134	1000 0110	0xC6	198	1100 0110
0x07	7	0000 0111	0x47	71	0100 0111	0x87	135	1000 0111	0xC7	199	1100 0111
0x08	8	0000 1000	0x48	72	0100 1000	0x88	136	1000 1000	0xC8	200	1100 1000
0x09	9	0000 1001	0x49	73	0100 1001	0x89	137	1000 1001	0xC9	201	1100 1001
0x0A	10	0000 1010	0x4A	74	0100 1010	0x8A	138	1000 1010	0xCA	202	1100 1010
0x0B	11	0000 1011	0x4B	75	0100 1011	0x8B	139	1000 1011	0xCB	203	1100 1011
0x0C	12	0000 1100	0x4C	76	0100 1100	0x8C	140	1000 1100	0xCC	204	1100 1100
0x0D	13	0000 1101	0x4D	77	0100 1101	0x8D	141	1000 1101	0xCD	205	1100 1101
0x0E	14	0000 1110	0x4E	78	0100 1110	0x8E	142	1000 1110	0xCE	206	1100 1110
0x0F	15	0000 1111	0x4F	79	0100 1111	0x8F	143	1000 1111	0xCF	207	1100 1111
0x10	16	0001 0000	0x50	80	0101 0000	0x90	144	1001 0000	0xD0	208	1101 0000
0x11	17	0001 0001	0x51	81	0101 0001	0x91	145	1001 0001	0xD1	209	1101 0001
0x12	18	0001 0010	0x52	82	0101 0010	0x92	146	1001 0010	0xD2	210	1101 0010
0x13	19	0001 0011	0x53	83	0101 0011	0x93	147	1001 0011	0xD3	211	1101 0011
0x14	20	0001 0100	0x54	84	0101 0100	0x94	148	1001 0100	0xD4	212	1101 0100
0x15	21	0001 0101	0x55	85	0101 0101	0x95	149	1001 0101	0xD5	213	1101 0101
0x16	22	0001 0110	0x56	86	0101 0110	0x96	150	1001 0110	0xD6	214	1101 0110
0x17	23	0001 0111	0x57	87	0101 0111	0x97	151	1001 0111	0xD7	215	1101 0111
0x18	24	0001 1000	0x58	88	0101 1000	0x98	152	1001 1000	0xD8	216	1101 1000
0x19	25	0001 1001	0x59	89	0101 1001	0x99	153	1001 1001	0xD9	217	1101 1001
0x1A	26	0001 1010	0x5A	90	0101 1010	0x9A	154	1001 1010	0xDA	218	1101 1010
0x1B	27	0001 1011	0x5B	91	0101 1011	0x9B	155	1001 1011	0xDB	219	1101 1011
0x1C	28	0001 1100	0x5C	92	0101 1100	0x9C	156	1001 1100	0xDC	220	1101 1100
0x1D	29	0001 1101	0x5D	93	0101 1101	0x9D	157	1001 1101	0xDD	221	1101 1101
0x1E	30	0001 1110	0x5E	94	0101 1110	0x9E	158	1001 1110	0xDE	222	1101 1110
0x1F	31	0001 1111	0x5F	95	0101 1111	0x9F	159	1001 1111	0xDF	223	1101 1111
0x20	32	0010 0000	0x60	96	0110 0000	0xA0	160	1010 0000	0xE0	224	1110 0000
0x21	33	0010 0001	0x61	97	0110 0001	0xA1	161	1010 0001	0xE1	225	1110 0001
0x22	34	0010 0010	0x62	98	0110 0010	0xA2	162	1010 0010	0xE2	226	1110 0010
0x23	35	0010 0011	0x63	99	0110 0011	0xA3	163	1010 0011	0xE3	227	1110 0011
0x24	36	0010 0100	0x64	100	0110 0100	0xA4	164	1010 0100	0xE4	228	1110 0100
0x25	37	0010 0101	0x65	101	0110 0101	0xA5	165	1010 0101	0xE5	229	1110 0101
0x26	38	0010 0110	0x66	102	0110 0110	0xA6	166	1010 0110	0xE6	230	1110 0110
0x27	39	0010 0111	0x67	103	0110 0111	0xA7	167	1010 0111	0xE7	231	1110 0111
0x28	40	0010 1000	0x68	104	0110 1000	0xA8	168	1010 1000	0xE8	232	1110 1000
0x29	41	0010 1001	0x69	105	0110 1001	0xA9	169	1010 1001	0xE9	233	1110 1001
0x2A	42	0010 1010	0x6A	106	0110 1010	0xAA	170	1010 1010	0xEA	234	1110 1010
0x2B	43	0010 1011	0x6B	107	0110 1011	0xAB	171	1010 1011	0xEB	235	1110 1011
0x2C	44	0010 1100	0x6C	108	0110 1100	0xAC	172	1010 1100	0xEC	236	1110 1100
0x2D	45	0010 1101	0x6D	109	0110 1101	0xAD	173	1010 1101	0xED	237	1110 1101
0x2E	46	0010 1110	0x6E	110	0110 1110	0xAE	174	1010 1110	0xEE	238	1110 1110
0x2F	47	0010 1111	0x6F	111	0110 1111	0xAF	175	1010 1111	0xEF	239	1110 1111
0x30	48	0011 0000	0x70	112	0111 0000	0xB0	176	1011 0000	0xF0	240	1111 0000
0x31	49	0011 0001	0x71	113	0111 0001	0xB1	177	1011 0001	0xF1	241	1111 0001
0x32	50	0011 0010	0x72	114	0111 0010	0xB2	178	1011 0010	0xF2	242	1111 0010
0x33	51	0011 0011	0x73	115	0111 0011	0xB3	179	1011 0011	0xF3	243	1111 0011
0x34	52	0011 0100	0x74	116	0111 0100	0xB4	180	1011 0100	0xF4	244	1111 0100
0x35	53	0011 0101	0x75	117	0111 0101	0xB5	181	1011 0101	0xF5	245	1111 0101
0x36	54	0011 0110	0x76	118	0111 0110	0xB6	182	1011 0110	0xF6	246	1111 0110
0x37	55	0011 0111	0x77	119	0111 0111	0xB7	183	1011 0111	0xF7	247	1111 0111
0x38	56	0011 1000	0x78	120	0111 1000	0xB8	184	1011 1000	0xF8	248	1111 1000
0x39	57	0011 1001	0x79	121	0111 1001	0xB9	185	1011 1001	0xF9	249	1111 1001
0x3A	58	0011 1010	0x7A	122	0111 1010	0xBA	186	1011 1010	0xFA	250	1111 1010
0x3B	59	0011 1011	0x7B	123	0111 1011	0xBB	187	1011 1011	0xFB	251	1111 1011
0x3C	60	0011 1100	0x7C	124	0111 1100	0xBC	188	1011 1100	0xFC	252	1111 1100
0x3D	61	0011 1101	0x7D	125	0111 1101	0xBD	189	1011 1101	0xFD	253	1111 1101
0x3E	62	0011 1110	0x7E	126	0111 1110	0xBE	190	1011 1110	0xFE	254	1111 1110
0x3F	63	0011 1111	0x7F	127	0111 1111	0xBF	191	1011 1111	0xFF	255	1111 1111

SEMTECH INTERNATIONAL AG
WIRELESS AND SENSING PRODUCTS

Contact Information

Taiwan	Tel: 886-2-2748-3380 Fax: 886-2-2748-3390	Switzerland	Tel: 41-32-729-4000 Fax: 41-32-729-4001
Korea	Tel: 82-2-527-4377 Fax: 82-2-527-4376	United Kingdom	Tel: 44-1794-527-600 Fax: 44-1794-527-601
Shanghai	Tel: 86-21-6391-0830 Fax: 86-21-6391-0831	France	Tel: 33-(0)169-28-22-00 Fax: 33-(0)169-28-12-98
Japan	Tel: 81-3-6408-0950 Fax: 81-3-6408-0951	Germany	Tel: 49-(0)8161-140-123 Fax: 49-(0)8161-140-124

Semtech International AG is a wholly-owned subsidiary of Semtech Corporation, which has its headquarters in the U.S.A