Welcome to E-XFL.COM

Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance

Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	Sensing Machine
Core Processor	Coolrisc816®
Program Memory Type	FLASH (22kB)
Controller Series	XE8000
RAM Size	512 x 8
Interface	UART, USRT
Number of I/O	24
Voltage - Supply	2.4V ~ 5.5V
Operating Temperature	-40°C ~ 85°C
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	64-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/semtech/xe8805ami028lf

Dec reg	C, V, Z, a	a := reg-1; if a=hFF then C := 0 else C := 1; reg := a
Dec reg, eaddr	C, V, Z, a	a := DM(eaddr)-1; if a=hFF then C := 0 else C := 1; reg := a
Decc reg1, reg2	C, V, Z, a	a := reg2-(1-C); if a=hFF and C=0 then C := 0 else C := 1; reg1 := a
Decc reg	C, V, Z, a	a := reg-(1-C); if a=hFF and C=0 then C := 0 else C := 1; reg := a
Decc reg, eaddr	C, V, Z, a	a := DM(eaddr)-(1-C); if a=hFF and C=0 then C := 0 else C := 1; reg := a
And reg,#data[7:0]	-,-, Z, a	a := reg and data[7:0]; reg := a
And reg1, reg2, reg3	-,-, Z, a	a := reg2 and reg3; reg1 := a
And reg1, reg2	-,-, Z, a	a := reg1 and reg2; reg1 := a
And reg, eaddr	-,-, Z, a	a := reg and DM(eaddr); reg := a
Or reg,#data[7:0]	-,-, Z, a	a := reg or data[7:0]; reg := a
Or reg1, reg2, reg3	-,-, Z, a	a := reg2 or reg3; reg1 := a
Or reg1, reg2	-,-, Z, a	a := reg1 or reg2; reg1 := a
Or reg, eaddr	-,-, Z, a	a := reg or DM(eaddr); reg := a
Xor reg,#data[7:0]	-,-, Z, a	a := reg xor data[7:0]; reg := a
Xor reg1, reg2, reg3	-,-, Z, a	a := reg2 xor reg3; reg1 := a
Xor reg1, reg2	-,-, Z, a	a := reg1 xor reg2; reg1 := a
Xor reg, eaddr	-,-, Z, a	a := reg or DM(eaddr); reg := a
Add reg,#data[7:0]	C, V, Z, a	a := reg+data[7:0]; if overflow then C:=1 else C := 0; reg := a
Add reg1, reg2, reg3	C, V, Z, a	a := reg2+reg3; if overflow then C:=1 else C := 0; reg1 := a
Add reg1, reg2	C, V, Z, a	a := reg1+reg2; if overflow then C:=1 else C := 0; reg1 := a
Add reg, eaddr	C, V, Z, a	a := reg+DM(eaddr); if overflow then C:=1 else C := 0; reg := a
Addc reg,#data[7:0]	C, V, Z, a	a := reg+data[7:0]+C; if overflow then C:=1 else C := 0; reg := a
Addc reg1, reg2, reg3	C, V, Z, a	a := reg2+reg3+C; if overflow then C:=1 else C := 0; reg1 := a
Addc reg1, reg2	C, V, Z, a	a := reg1+reg2+C; if overflow then C:=1 else C := 0; reg1 := a
Addc reg, eaddr	C, V, Z, a	a := reg+DM(eaddr)+C; if overflow then C:=1 else C := 0; reg := a
Subd reg,#data[7:0]	C, V, Z, a	a := data[7:0]-reg; if underflow then C := 0 else C := 1; reg := a
Subd reg1, reg2, reg3	C, V, Z, a	a := reg2-reg3; if underflow then C := 0 else C := 1; reg1 := a
Subd reg1, reg2	C, V, Z, a	a := reg2-reg1; if underflow then C := 0 else C := 1; reg1 := a
Subd reg, eaddr	C, V, Z, a	a := DM(eaddr)-reg; if underflow then C := 0 else C := 1; reg := a
Subdc reg,#data[7:0]	C, V, Z, a	a := data[7:0]-reg-(1-C); if underflow then C := 0 else C := 1; reg := a
Subdc reg1, reg2, reg3	C, V, Z, a	a := reg2-reg3-(1-C); if underflow then C := 0 else C := 1; reg1 := a
Subdc reg1, reg2	C, V, Z, a	a := reg2-reg1-(1-C); if underflow then C := 0 else C := 1; reg1 := a
Subdc reg, eaddr	C, V, Z, a	a := DM(eaddr)-reg-(1-C); if underflow then C := 0 else C := 1; reg := a
Subs reg,#data[7:0]	C, V, Z, a	a := reg-data[7:0]; if underflow then C := 0 else C := 1; reg := a
Subs reg1, reg2, reg3	C, V, Z, a	a := reg3-reg2; if underflow then C := 0 else C := 1; reg1 := a
Subs reg1, reg2	C, V, Z, a	a := reg1-reg2; if underflow then C := 0 else C := 1; reg1 := a
Subs reg, eaddr	C, V, Z, a	a := reg-DM(eaddr); if underflow then C := 0 else C := 1; reg := a
Subsc reg,#data[7:0]	C, V, Z, a	a := reg-data[7:0]-(1-C); if underflow then C := 0 else C := 1; reg := a
Subsc reg1, reg2, reg3	C, V, Z, a	a := reg3-reg2-(1-C); if underflow then C := 0 else C := 1; reg1 := a
Subsc reg1, reg2	C, V, Z, a	a := reg1-reg2-(1-C); if underflow then C := 0 else C := 1; reg1 := a
Subsc reg, eaddr	C, V, Z, a	a := reg-DM(eaddr)-(1-C); if underflow then C := 0 else C := 1; reg := a
Mul reg,#data[7:0]	u, u, u, a	a := (data[7:0]*reg)[7:0]; reg := (data[7:0]*reg)[15:8]
Mul reg1, reg2, reg3	u, u, u, a	a := (reg2*reg3)[7:0]; reg1 := (reg2*reg3)[15:8]
Mul reg1, reg2	u, u, u, a	a := (reg2*reg1)[7:0]; reg1 := (reg2*reg1)[15:8]
Mul reg, eaddr	u, u, u, a	a := (DM(eaddr)*reg)[7:0]; reg := (DM(eaddr)*reg)[15:8]
Mula reg,#data[7:0]	u, u, u, a	a := (data[7:0]*reg)[7:0]; reg := (data[7:0]*reg)[15:8]
Mula reg1, reg2, reg3	u, u, u, a	a := (reg2*reg3)[7:0]; reg1 := (reg2*reg3)[15:8]
Mula reg1, reg2	u, u, u, a	a := (reg2*reg1)[7:0]; reg1 := (reg2*reg1)[15:8]
Mula reg, eaddr	u, u, u, a	a := (DM(eaddr)*reg)[7:0]; reg := (DM(eaddr)*reg)[15:8]
Mshl reg,#shift[2:0]	u, u, u, a	a := (reg*2^{shift})[7:0]; reg := (reg*2^{shift})[15:8]
Mshr reg,#shift[2:0]	u, u, u, a	a := (reg*$2^{(8-shift)}$)[7:0]; reg := (reg*$2^{(8-shift)}$)[15:8]
Mshra reg,#shift[2:0]	u, u, u, a*	a := (reg*$2^{(8-shift)}$)[7:0]; reg := (reg*$2^{(8-shift)}$)[15:8]
Cmp reg,#data[7:0]	C, V, Z, a	a := data[7:0]-reg; if underflow then C :=0 else C:=1; V := C and (not Z)
Cmp reg1, reg2	C, V, Z, a	a := reg2-reg1; if underflow then C :=0 else C:=1; V := C and (not Z)
Cmp reg, eaddr	C, V, Z, a	a := DM(eaddr)-reg; if underflow then C :=0 else C:=1; V := C and (not Z)
Cmpa reg,#data[7:0]	C, V, Z, a	a := data[7:0]-reg; if underflow then C :=0 else C:=1; V := C and (not Z)
Cmpa reg1, reg2	C, V, Z, a	a := reg2-reg1; if underflow then C :=0 else C:=1; V := C and (not Z)
Cmpa reg, eaddr	C, V, Z, a	a := DM(eaddr)-reg; if underflow then C :=0 else C:=1; V := C and (not Z)
Tstb reg,#bit[2:0]	-, -, Z, a	a[bit] := reg[bit]; other bits in a are 0
Setb reg,#bit[2:0]	-, -, Z, a	reg[bit] := 1; other bits unchanged; a := reg
Clrb reg,#bit[2:0]	-, -, Z, a	reg[bit] := 0; other bits unchanged; a := reg
Invb reg,#bit[2:0]	-, -, Z, a	reg[bit] := not reg[bit]; other bits unchanged; a := reg

5 System Block

6.1 Features

- Power On Reset (POR)
- External reset from the RESET pin
- Programmable Watchdog timer reset
- Programmable BusError reset
- Sleep mode management
- Programmable Port A input combination reset

6.2 Overview

The reset block is the reset manager. It handles the different reset sources and distributes them through the system. It also controls the sleep mode of the circuit.

6.3 Register map

Pos.	RegSysCtrl	Rw	Reset	Function
7	SleepEn	r w	0 resetcold	enables Sleep mode 0: sleep mode is disabled 1: sleep mode is enabled
6	EnResPConf	r w	0 resetcold	enables the resetpconf signal when the resetglobal is active 0: resetpconf is disabled 1: resetpconf is enabled
5	EnBusError	r w	0 resetcold	enables reset from BusError 0: BusError reset source is disabled 1: BusError reset source is enabled
4	EnResWD	r w	0 resetcold	enables reset from Watchdog 0: Watchdog reset source is disabled 1: Watchdog reset source is enabled this bit can not be set to 0 by SW
3 – 0	-	r	0000	unused

Table 6-1. **RegSysCtrl** register.

Pos.	RegSysReset	Rw	Reset	Function
7	Sleep	rw	0 resetsystem	Sleep mode control (reads always 0)
6	-	r	0	unused
5	ResetBusError	r c	0 resetcold	reset source was BusError
4	ResetWD	r c	0 resetcold	reset source was Watchdog
3	ResetfromportA	r c	0 resetcold	reset source was Port A combination
2	ResPad	r c	0 resetcold	reset source was reset pad
1	ResPadSleep	r c	0 resetcold	reset source was reset pad in sleep mode
0	-	r	0	unused

Table 6-2. **RegSysReset** register

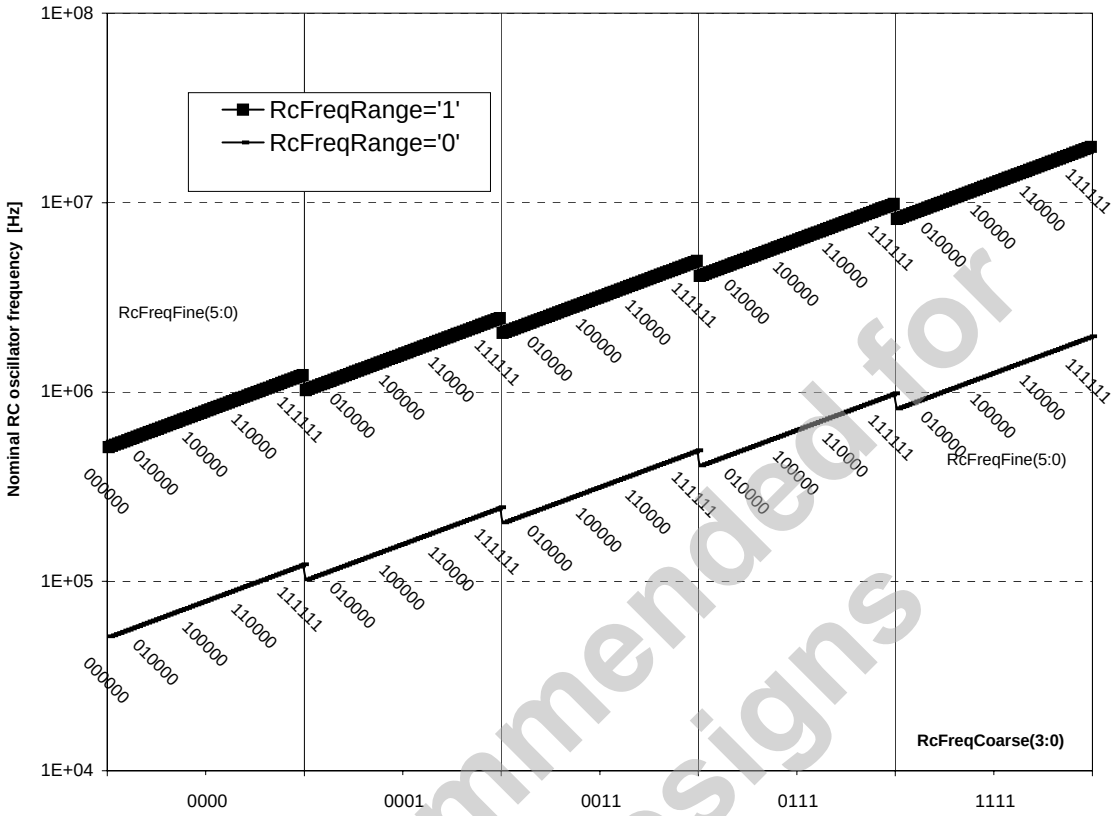


Figure 7-2. RC oscillator nominal frequency tuning.

7.5.1.3 RC oscillator specifications

symbol	description	min	typ	max	unit	Comments
f_{RCmin}	Lowest RC frequency	40	80	120	kHz	Note 1
RcFreqFine	fine tuning step		1.4	2.0	%	
RC_su	startup time		30	50	us	**BiasRc**=0
			3	5	us	**BiasRc**=1
PSRR @ DC	Supply voltage dependence		TBD		%/V	Note 2
			TBD		%/V	Note 3
$\Delta f/\Delta T$	Temperature dependence		0.1		%/°C	

Table 7-7. RC oscillator specifications

Note 1: this is the frequency tolerance when all trimming codes are 0.

Note 2: frequency shift as a function of VBAT with normal regulator function.

Note 3: frequency shift as a function of VBAT while the regulator is short-circuited to VBAT.

The tolerances on the minimal frequency and the drift with supply or temperature can be cancelled using the software DFLL (digital frequency locked loop) which uses the crystal oscillator as a reference frequency.

10 Low Power RAM

10.1 Features

- Low power RAM locations.

10.2 Overview

In order to save power consumption, 8 8-bit registers are provided in page 0. These memory locations should be reserved for often-updated variables. Accessing these register locations requires much less power than the other general purpose RAM locations.

10.3 Register map

pos.	Reg00	rw	reset	function
7-0	Reg00	rw	XXXXXXXX	low-power data memory

Table 10-1: Reg00

pos.	Reg01	rw	reset	function
7-0	Reg01	rw	XXXXXXXX	low-power data memory

Table 10-2: Reg01

pos.	Reg02	rw	reset	function
7-0	Reg02	rw	XXXXXXXX	low-power data memory

Table 10-3: Reg02

pos.	Reg03	rw	reset	function
7-0	Reg03	rw	XXXXXXXX	low-power data memory

Table 10-4: Reg03

pos.	Reg04	rw	reset	function
7-0	Reg04	rw	XXXXXXXX	low-power data memory

Table 10-5: Reg04

pos.	Reg05	rw	reset	function
7-0	Reg05	rw	XXXXXXXX	low-power data memory

Table 10-6: Reg05

pos.	Reg06	rw	reset	function
7-0	Reg06	rw	XXXXXXXX	low-power data memory

Table 10-7: Reg06

pos.	Reg07	rw	reset	function
7-0	Reg07	rw	XXXXXXXX	low-power data memory

Table 10-8: Reg07

11 Port A

pos.	RegUartTxSta	rw	reset	description
7-2	-	r	000000	Unused
1	UartTxBusy	r	0 resetsystem	Uart busy transmitting
0	UartTxFull	r	0 resetsystem	**RegUartTx** full Set by writing to **RegUartTx** Cleared when transferring **RegUartTx** into internal shift register

Table 14-4: RegUartTxSta

pos.	RegUartRx	rw	reset	description
7-0	UartRx	r	00000000 resetsystem	Received data

Table 14-5: RegUartRx

pos.	RegUartRxSta	rw	Reset	description
7-6	-	r	00	Unused
5	UartRxSErr	r	0 resetsystem	Start error
4	UartRxPErr	r	0 resetsystem	Parity error
3	UartRxFErr	r	0 resetsystem	Frame error
2	UartRxOErr	rc	0 resetsystem	Overrun error Cleared by writing **RegUartRxSta**
1	UartRxBusy	r	0 resetsystem	Uart busy receiving
0	UartRxFull	r	0 resetsystem	**RegUartRx** full Cleared by reading **RegUartRx**

Table 14-6: RegUartRxSta

14.4 Interrupts map

interrupt source	default mapping in the interrupt manager
Irq_uart_Tx	**IrqHig(1)**
Irq_uart_Rx	**IrqHig(0)**

Table 14-7: Interrupts map

14.5 Uart baud rate selection

In order to have correct baud rates, the Uart interface has to be fed with a stable and trimmed clock source. The clock source can be the RC oscillator or the crystal oscillator. The precision of the baud rate will depend on the precision of the selected clock source.

14.5.1 Uart on the RC oscillator

To select the RC oscillator for the Uart, the bit **SelXtal** in **RegUartCmd** has to be 0.

In order to obtain a correct baud rate, the RC oscillator frequency has to be set to one of the frequencies given in the table on the next page. The precision of the obtained baud rate is directly proportional to the frequency deviation with respect to the values in the table.

pos.	RegAcCfg3	rw	reset	description
7	PGA1_GAIN	r w	0 resetsystem	PGA1 stage gain selection
6:0	PGA3_GAIN[6:0]	r w	0000000 resetsystem	PGA3 stage gain selection

Table 16-7: RegAcCfg3

pos.	RegAcCfg4	rw	reset	description
7	reserved	r	0	Unused
6:0	PGA3_OFFSET[6:0]	r w	0000000 resetsystem	PGA3 stage offset selection

Table 16-8: RegAcCfg4

pos.	RegAcCfg5	rw	reset	description
7	BUSY	r	0 resetsystem	Activity flag
6	DEF	w r0	0	Selects default configuration
5:1	AMUX[4:0]	r w	00000 resetsystem	Input channel configuration selector
0	VMUX	r w	0 resetsystem	Reference channel selector

Table 16-9: RegAcCfg5

16.4 ZoomingADC™ Description

Figure 16-2 gives a more detailed description of the acquisition chain.

16.4.1 Acquisition Chain

Figure 16-1 shows the general block diagram of the acquisition chain (AC). A control block (not shown in Figure 16-1) manages all communications with the CoolRisc™ microcontroller.

Analog inputs can be selected among eight input channels, while reference input is selected between two differential channels.

The core of the zooming section is made of three differential programmable amplifiers (PGA). After selection of a combination of input and reference signals V_{IN} and V_{REF}, the input voltage is modulated and amplified through stages 1 to 3. Fine gain programming up to 1'000V/V is possible. In addition, the last two stages provide programmable offset. Each amplifier can be bypassed if needed.

The output of the PGA stages is directly fed to the analog-to-digital converter (ADC), which converts the signal $V_{IN,ADC}$ into digital.

Like most ADCs intended for instrumentation or sensing applications, the ZoomingADC™ is an over-sampled converter (See Note[1]). The ADC is a so-called incremental converter, with bipolar operation (the ADC accepts both positive and negative input voltages). In first approximation, the ADC output result relative to full-scale (*FS*) delivers the quantity:

[1] Note: Over-sampled converters are operated with a sampling frequency f_S much higher than the input signal's Nyquist rate (typically f_S is 20-1'000 times the input signal bandwidth). The sampling frequency to throughput ratio is large (typically 10-500). These converters include digital decimation filtering. They are mainly used for high resolution, and/or low-to-medium speed applications.

- BUSY: (r) set to 1 if a conversion is running. Note that the flag is set at the effective start of the conversion. Since the ADC is generally synchronized on a lower frequency clock than the CPU, there might be a small delay (max. 1 cycle of the ADC sampling frequency) between the writing of the START or CONT bits and the appearance of BUSY flag.
- DEF: (w) sets all values to their defaults (PGA disabled, max speed, nominal modulator bias current, 2 elementary conversions, over-sampling rate of 32) and starts a new conversion without waiting the end of the preceding one.
- AMUX(4:0): (rw) AMUX[4] sets the mode (0 → 4 differential inputs, 1 → 7 inputs with A(0) = common reference) AMUX(3) sets the sign (0 → straight, 1→ cross) AMUX[2:0] sets the channel.
- VMUX: (rw) sets the differential reference channel (0 → R(1) and R(0), 1 → R(3) and R(2)).

(r = read; w = write; rw = read & write)

VMUX (RegAcCfg5[0])	V_{REFP}	V_{REFN}
0	AC_R(1)	AC_R(0)
1	AC_R(3)	AC_R(2)

Table 16-12. Analog Reference input selection

16.6 Programmable Gain Amplifiers

As seen in Figure 16-1, the zooming function is implemented with three programmable gain amplifiers (PGA). These are:

- PGA1: coarse gain tuning
- PGA2: medium gain and offset tuning
- PGA3: fine gain and offset tuning

All gain and offset settings are realized with ratios of capacitors. The user has control over each PGA activation and gain, as well as the offset of stages 2 and 3. These functions are examined hereafter.

ENABLE[3:0]	Block
xxx0 xxx1	ADC disabled ADC enabled
xx0x xx1x	PGA1 disabled PGA1 enabled
x0xx x1xx	PGA2 disabled PGA2 enabled
0xxx 1xxx	PGA3 disabled PGA3 enabled

Table 16-13. ADC & PGA enabling

PGA1_GAIN	PGA1 Gain GD_1 (V/V)
0	1
1	10

Table 16-14. PGA1 Gain Settings

PGA2_GAIN[1:0]	PGA2 Gain GD_2 (V/V)
00	1
01	2
10	5
11	10

Table 16-15. PGA2 gain settings

PGA2_OFFSET[3:0]	PGA2 Offset $GDoff_2$ (V/V)
0000	0
0001	+0.2
0010	+0.4
0011	+0.6
0100	+0.8
0101	+1
1001	-0.2
1010	-0.4
1011	-0.6
1100	-0.8
1101	-1

Table 16-16. PGA2 offset settings

PGA3_GAIN[6:0]	PGA3 Gain GD_3 (V/V)
0000000	0
0000001	1/12(=0.083)
...	...
0000110	6/12
...	...
0001100	12/12
0010000	16/12
...	
0100000	32/12
...	
1000000	64/12
...	
1111111	127/12(=10.58)

Table 16-17. PGA3 gain settings

PGA3_OFFSET[6:0]	PGA3 Offset $GDoff_3$ (V/V)
0000000	0
0000001	+1/12(=+0.083)
0000010	+2/12
...	...
0010000	+16/12
...	...
0100000	+32/12
...	...
0111111	+63/12(=+5.25)
1000000	0
1000001	-1/12(=-0.083)
1000010	-2/12
...	...
1010000	-16/12
...	...
1100000	-32/12
...	...
1111111	-63/12(=-5.25)

Table 16-18. PGA3 offset settings

Finally, combining equations Eq. 5 to Eq. 7 for the three PGA stages, the input voltage $V_{IN,ADC}$ of the ADC is related to V_{IN} by:

$$V_{IN,ADC} = GD_{TOT} \cdot V_{IN} - GDoff_{TOT} \cdot V_{REF} \text{ (V)} \qquad \text{(Eq. 9)}$$

where the total PGA gain is defined as:

$$GD_{TOT} = GD_3 \cdot GD_2 \cdot GD_1 \qquad \text{(V/V)} \qquad \text{(Eq. 10)}$$

and the total PGA offset is:

$$GDoff_{TOT} = GDoff_3 + GD_3 \cdot GDoff_2 \qquad \text{(V/V)} \qquad \text{(Eq. 11)}$$

16.7 ADC Characteristics

The main performance characteristics of the ADC (resolution, conversion time, etc.) are determined by three programmable parameters:

- sampling frequency f_S,
- over-sampling ratio *OSR*, and
- number of elementary conversions N_{ELCONV}.

The setting of these parameters and the resulting performances are described hereafter.

16.7.1 Conversion Sequence

A conversion is started each time the bit **START** or the bit **DEF** is set. As depicted in Figure 16-5, a complete analog-to-digital conversion sequence is made of a set of N_{ELCONV} elementary incremental conversions and a final quantization step. Each elementary conversion is made of (*OSR*+1) sampling periods T_S=1/f_S, i.e.:

$$T_{ELCONV} = (OSR+1)/f_S \text{ (s)} \qquad \text{(Eq. 12)}$$

The result is the mean of the elementary conversion results. An important feature is that the elementary conversions are alternatively performed with the offset of the internal amplifiers contributing in one direction and the other to the output code. Thus, converter internal offset is eliminated if at least two elementary sequences are performed (i.e. if $N_{ELCONV} \geq 2$). A few additional clock cycles are also required to initiate and end the conversion properly.

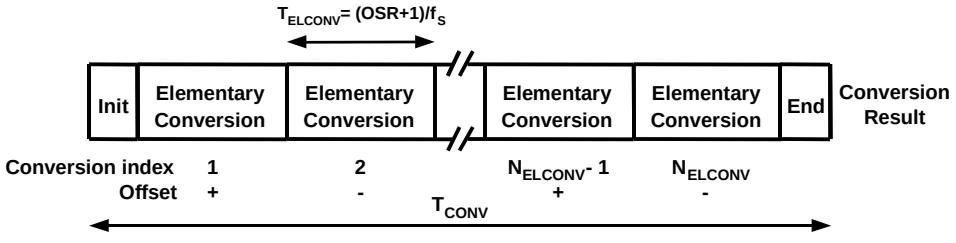


Figure 16-5. Analog-to-digital conversion sequence

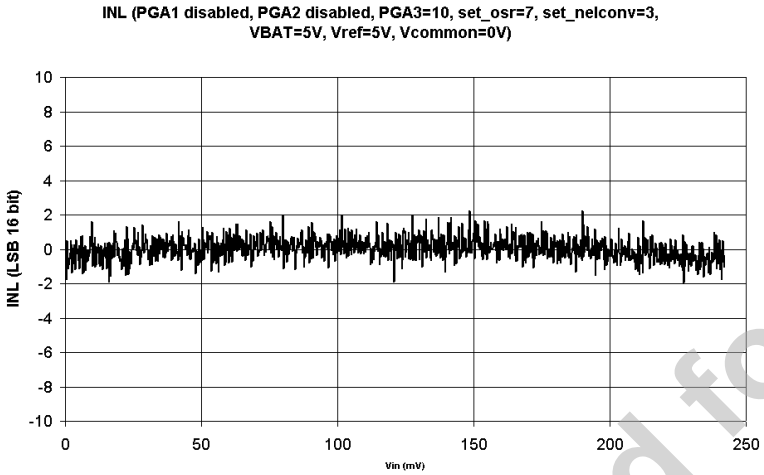


Figure 16-12. Integral non-linearity of the ADC and gain of 10 (PGA1 and PGA2 disabled, PGA3=10, reference voltage of 5V)

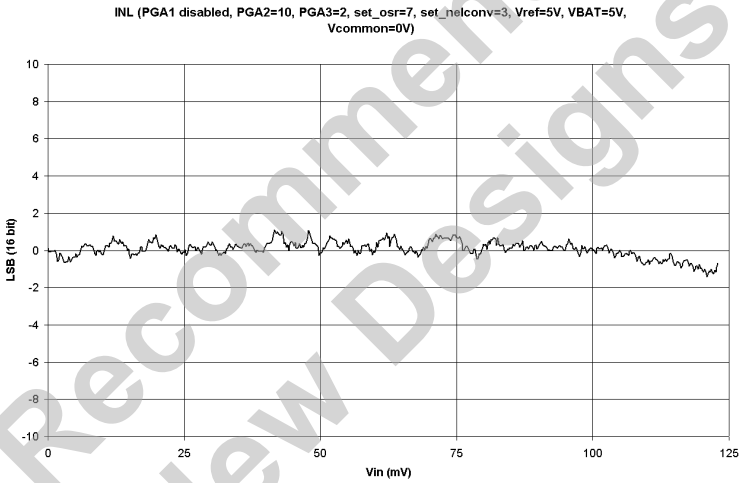


Figure 16-13. Integral non-linearity of the ADC and gain of 20 (PGA1 and PGA2=10, PGA3=2, reference voltage of 5V)

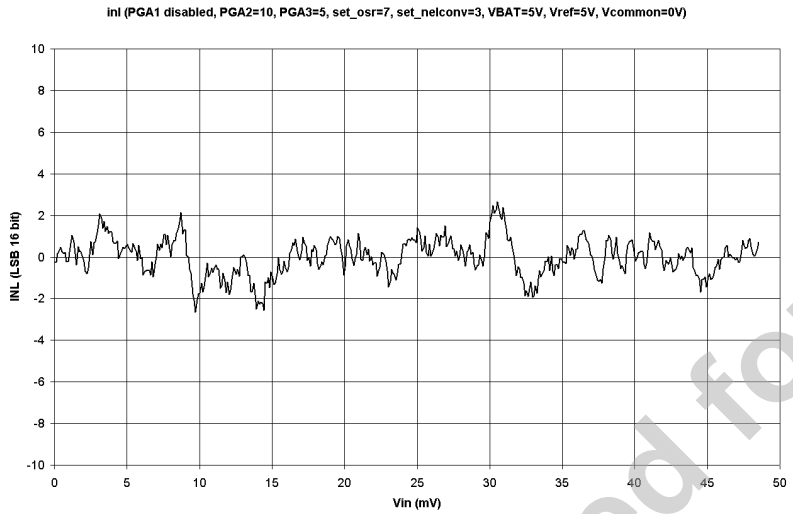


Figure 16-14. Integral non-linearity of the ADC and gain of 50 (PGA1 disabled, PGA2=10, PGA3=5, reference voltage of 5V)

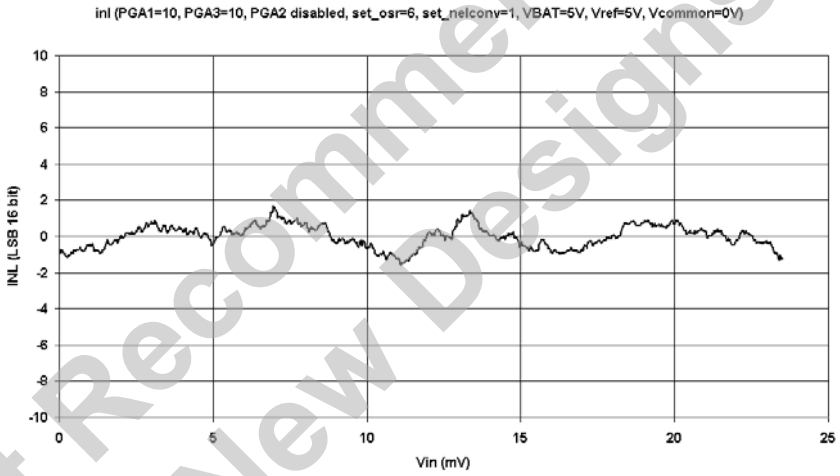


Figure 16-15. Integral non-linearity of the ADC and gain of 100 (PGA1=10 and PGA3=10, PGA2 disabled, reference voltage of 5V)

16.8.3.2 Differential non-linearity

The differential non-linearity is generated by the ADC. The PGA does not add differential non-linearity. Figure 16-16 shows the differential non-linearity.

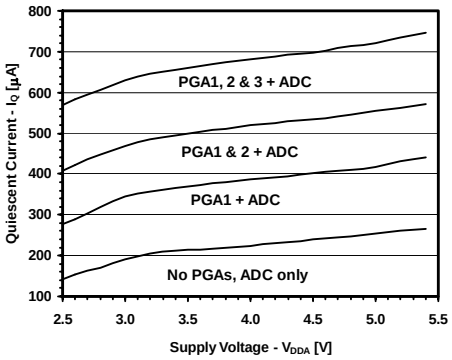


Figure 16-21. Quiescent current consumption vs. supply voltage

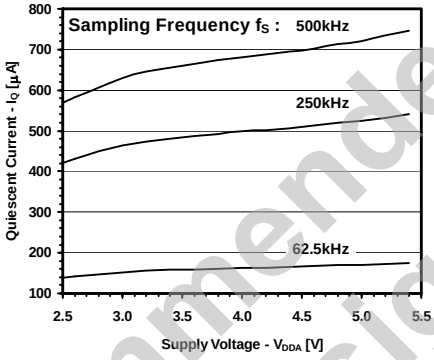
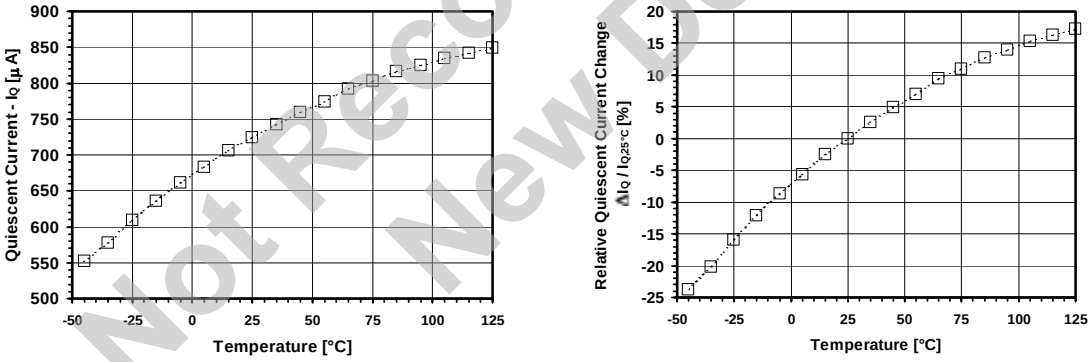


Figure 16-22. Quiescent current consumption vs. supply voltage for different sampling frequencies



(a) **(b)**

Figure 16-23. (a) Absolute and (b) relative change inquiescent current consumption vs. temperature

Supply	GAIN = 1	GAIN =5	GAIN = 10	GAIN = 20	GAIN =100	Unit
V_{DD} = 5V	79	78	100	99	97	dB
V_{DD} = 3V	72	79	90	90	86	dB

Table 16-29. PSRR (n = 16 bits, V_{IN} = V_{REF} = 2.5V, f_S = 500kHz)

16.9 Application Hints

16.9.1 Input Impedance

The PGAs of the acquisition chain employ switched-capacitor techniques. For this reason, while a conversion is done, the input impedance on the selected channel of the PGAs is inversely proportional to the sampling frequency f_S and to stage gain as given in equation 22.

$$Z_{in} \geq \frac{768 \cdot 10^9 \Omega Hz}{f_s \cdot gain} \quad \text{(Eq. 22)}$$

The input impedance observed is the input impedance of the first PGA stage that is enabled or the input impedance of the ADC if all three stages are disabled.

PGA1 (with a gain of 10), PGA2 (with a gain of 10) and PGA3 (with a gain of 10) each have a minimum input impedance of 150kΩ at f_S = 512kHz (see Specification Table). Larger input impedance can be obtained by reducing the gain and/or by reducing the sampling frequency. Therefor, with a gain of 1 and a sampling frequency of 100kHz, Z_{in} > 7.6MΩ.

The input impedance on channels that are not selected is very high (>100MΩ).

16.9.2 PGA Settling or Input Channel Modifications

PGAs are reset after each writing operation to registers **RegAcCfg1-5**. Similarly, input channels are switched after modifications of **AMUX[4:0]** or **VMUX**. To ensure precise conversion, the ADC must be started after a PGA or inputs common-mode stabilization delay. This is done by writing bit **START** several cycles after PGA settings modification or channel switching. Delay between PGA start or input channel switching and ADC start should be equivalent to *OSR* (between 8 and 1024) number of cycles. This delay does not apply to conversions made without the PGAs.

If the ADC is not settled within the specified period, there is most probably an input impedance problem (see previous section).

16.9.3 PGA Gain & Offset, Linearity and Noise

Hereafter are a few design guidelines that should be taken into account when using the ZoomingADC™:

1) Keep in mind that increasing the overall PGA gain, or "zooming" coefficient, improves linearity but degrades noise performance.
2) Use the minimum number of PGA stages necessary to produce the desired gain ("zooming") and offset. Bypass unnecessary PGAs.
3) For high gains (>50), use PGA stage 1. For low gains (<50) use stages 2 and 3.
4) For the lowest noise, set the highest possible gain on the first (front) PGA stage used in the chain. For example, in an application where a gain of 20 is needed, set the gain of PGA2 to 10, set the gain of PGA3 to 2.

17.1 Features

- Generates a voltage that is higher or equal to the supply voltage.
- Can be easily enabled or disabled

17.2 Overview

The Vmult block generates a voltage (called “Vmult”) that is higher or equal to the supply voltage. This output voltage is used in the acquisition chain.

The voltage multiplier should be on (bit **ENABLE** in **RegVmultCfg0**) when using the acquisition chain or analog properties of the Port B while VBAT is below 3V. If the multiplier is enabled, the external capacitor on the pin VMULT is mandatory.

The source clock of Vmult is selected by **FIN[1:0]** in **RegVmultCfg0**. It is strongly recommended to use the same settings as in the ADC.

17.3 Control register

There is only one register in the Vmult. Table 177-1 describes the bits in the register.

Pos.	RegVmultCfg0	rw	Reset	Function
2	Enable	rw	0 resetsystem	enable of the vmult ‘1’ : enabled ‘0’ : disabled
1-0	Fin	rw	0 resetsystem	system clock division factor ‘00’ : 1/2, ‘01’ : 1/4, ‘10’ : 1/16, ‘11’ : 1/64

Table 177-1. **RegVmultCfg0**

17.4 External component

When the multiplier is enabled, a capacitor has to be connected to the VMULT pin. If the multiplier is disabled, the pin may remain floating.

	Min.	Max.		Note
Capacitor on VMULT	1.0	3.0	nF	

18. Signal D/A (DAS)

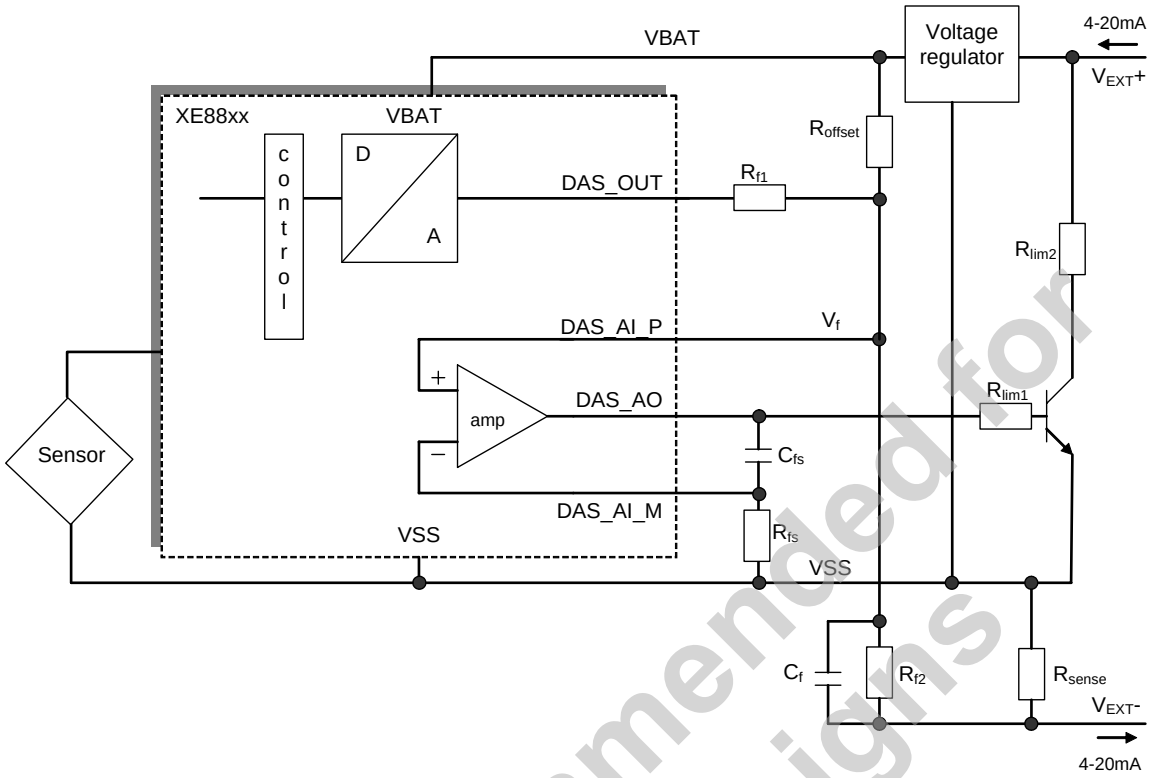


Figure 18-6. 2-wire 4-20mA with second order filter and increased stability