TEXAS INSTRUMENTS

Nonlinear Improvement of the MSP430 14-Bit ADC Characteristic

Application Report

June 1999

Mixed Signal Products

SLAA050

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty or endorsement thereof.

Contents

List of Figures

List of Tables

Nonlinear Improvement of the MSP430 14-Bit ADC Characteristic

Lutz Bierl

ABSTRACT

This application report shows nonlinear methods—with quadratic and cubic equations—to improve the accuracy of the 14-bit analog-to-digital converter (ADC) of the MSP430 family. The methods used differ in RAM and ROM requirements, calculation speed, achievable improvement, and complexity. The influence of the restricted calculation accuracy for 8-bit coefficients is compared to the accuracy of floating-point calculations. Finally, a comparison of all improvement methods is given. The *References* section at the end of the report lists related application reports in the MSP430 14-bit ADC series.

1 Introduction

The application report *Architecture and Function of the MSP430 14-Bit ADC*[1] gives a detailed overview of the architecture and function of the 14-bit analog-to-digital converter (ADC) of the MSP430 family. The principle of the ADC is explained and software examples are given. Also included are the explanation of the function of all hardware registers contained in the ADC.

The application report *Application Basics for the MSP430 14-Bit ADC*[2] shows several applications of the 14-bit ADC of the MSP430 family. Proven software examples and basic circuitry are shown and explained.

The application report *Additive Improvement of the MSP430 14-Bit ADC Characteristic*[3] explains the external hardware that is needed for the measurement of the characteristic of the MSP430's analog-to-digital converter. This report also demonstrates correction methods that use addition only: no multiplication is needed. This allows the application of these methods in real-time systems, where execution time can be critical.

The application report *Linear Improvement of the MSP430 14-Bit ADC Characteristic*[4] shows linear improvements using linear equations with border fit and correction by linear regression methods.

Figure 1 shows the block diagram of the 14-bit analog-to-digital converter of the MSP430 family.

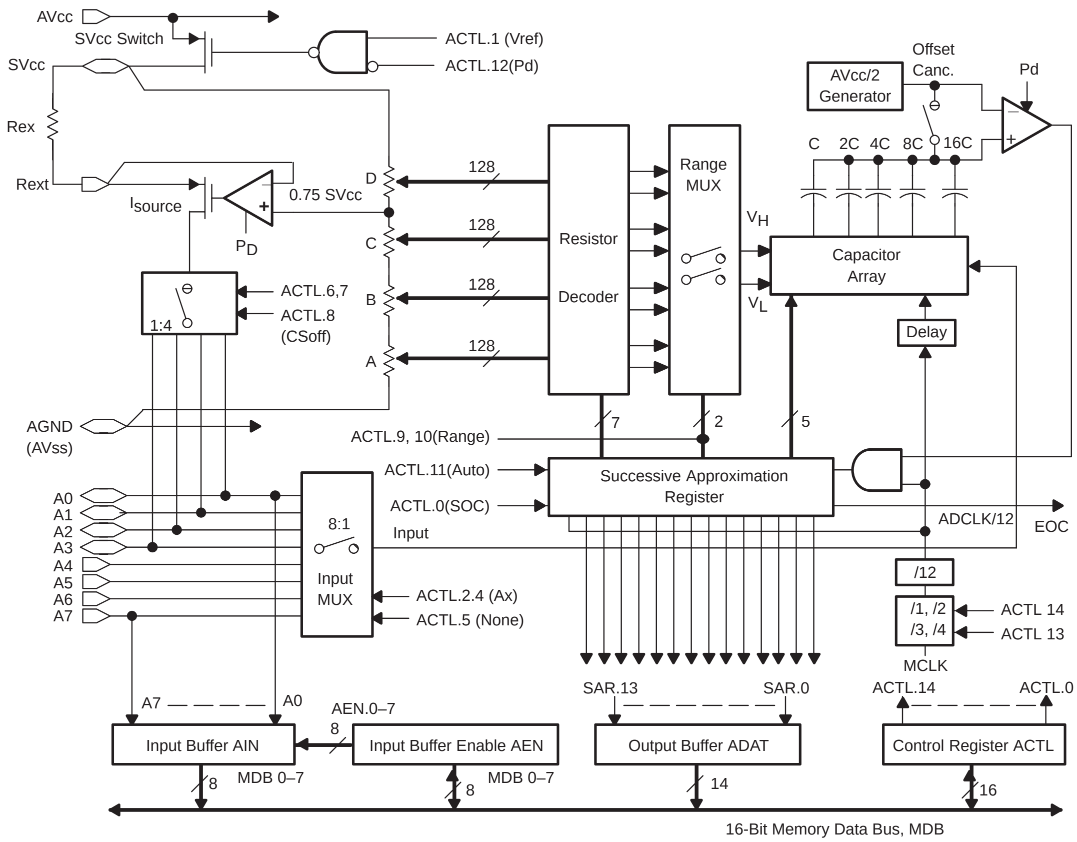


Figure 1. The Hardware of the 14-Bit Analog-to-Digital Converter

The methods for the improvement of the ADC discussed in this report are:

- Correction with nonlinear equations
- Correction with one quadratic equation per range
- Correction with one cubic equation per range

1.1 Correction With Quadratic Equations

The ADC is measured at three points within all four ranges. These points are used for a quadratic correction (one correction for each range). It is recommended to use more than one measurement for each one of these three important points. *Additive Improvement of the MSP430 14-Bit ADC Characteristic Application Report*[3] section 2.1 for details. Normally these three points are the two borders and the center of the actual range. This has two advantages:

- The ranges continue smoothly at the common borders.
- Only nine points of the ADC characteristic need to be measured for the full ADC range during the calibration. This is due to the common range border points.

But other arrangements are possible.

The improvement methods and their results for this report are demonstrated with the characteristic of device 1 due to its worst characteristic compared to the other three devices shown in *Architecture and Function of the MSP430 14-Bit ADC Application Report*.[1]

The formula used for each range with separate factors *ax* for 8-bit integer calculations is:

$$Nicorr = Ni + \left(\left(\left(\frac{Ni}{4096} - n\right) \times 256\right)^2 \times a2 + \left(\frac{Ni}{4096} - n\right) \times 256 \times a1 + a0\right)$$

Where:	Nicorr	Corrected ADC sample	[Steps]
	Ni	Measured ADC sample (non-corrected)	[Steps]
	N	Subdivision representing the ADC sample (0...255)	
	n	Range number (0...3 for ranges A...D)	
	a2	Quadratic coefficient of the correction	[Steps^{-1}]
	a1	Linear coefficient of the correction	
	a0	Offset of the correction	[Steps]
	i	Nominal ADC step of the ADC input (DAC output)	[Steps]

The term $N = \left(\frac{Ni}{4096} - n\right) \times 256$ of the equation above is the adaptation of a complete section—here a full range—to 256 subdivisions. The calculation of the term is made by simple shifts rather than division and a multiplication. Rounding is used to achieve better accuracy. See the initialization part of the software example.

The formula above uses the subdivisions (0 to 255) inside of an ADC range (0 to 4095 steps) instead of the full 14-bit position (0 to 16383 steps) of an ADC point. This is to maintain the accuracy of the calculation with limited coefficient length (here for 8-bit coefficients). The above formula is used with the 8-bit calculation.

If floating point calculation or 16-bit arithmetic is used, the higher resolution makes the range correction unnecessary: the full 14-bit result may be used for the calculations.

$$Nicorr = Ni + \left(Ni^2 \times a2 + Ni \times a1 + a0\right)$$

The software example given for the cubic correction in section 1.3—which is written in floating point notation—may be adapted easily to quadratic correction: the unused cubic part is simply left out and the address calculation for the coefficients is modified to three coefficients (a2..a0) instead of the four (a3..a0).

To save multiplications, the so-called *Horner* scheme is used. This scheme is applicable for all given examples. The formula using the 8-bit arithmetic now becomes:

$$Nicorr = Ni + \left(\left(\left(\frac{Ni}{4096} - n\right) \times 256 \times a2 + a1\right) \times \left(\frac{Ni}{4096} - n\right) \times 256 \times a0\right)$$

The 16-bit formula and the FPP formula now require only two multiplications instead of three.

$$Nicorr = Ni + ((Ni \times a2 + a1)Ni + a0)$$

Figure 2 shows the principle of the correction with four quadratic equations: the used correction parabolas are drawn together with the non-corrected ADC characteristic. As with all principle figures in this report, the black straight line indicates the correction value, the scribbled black line indicates the non-corrected ADC characteristic and the white line shows the corrected ADC characteristic. The small circles indicate the measured ADC points.

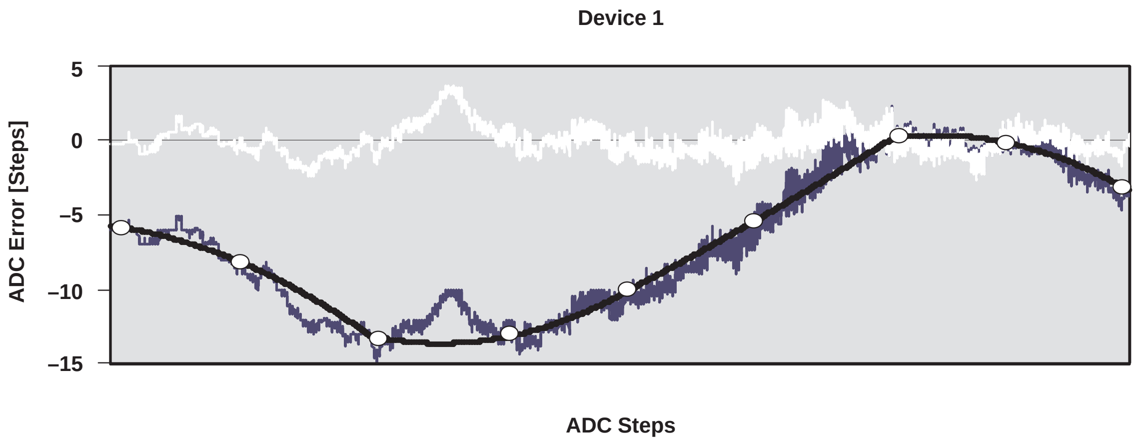


Figure 2. Principle of the Error Correction With Four Quadratic Equations

The statistical results for the quadratic correction are (single measurement for each one of the nine ADC steps used for the calculation of the correction coefficients):

	Full range	**Ranges A and B only**
Mean Value:	–0.08 Steps	0.24 Steps
Range:	6.78 Steps	5.86 Steps
Standard Deviation:	1.05 Steps	1.10 Steps
Variance:	1.11 Steps	1.21 Steps

If each of the nine ADC steps used for the calculation of the correction coefficients is measured in a slightly modified way, then the statistical results change also. Now the mean value of seven measured ADC steps is taken for the calculation. The seven ADC steps are:

Nn–12, Nn–8, Nn–4, Nn, Nn+4, Nn+8 and Nn+12, where Nn is the ADC step used in the calculation formula. Now the statistical results are:

	Full range	**Ranges A and B only**
Mean Value:	–0.07 Steps	0.11 Steps
Range:	6.47 Steps	6.02 Steps
Standard Deviation:	1.00 Steps	1.12 Steps
Variance:	1.00 Steps	1.24 Steps

Figure 3 shows the resulting errors of both methods in a graph (differences cannot be seen):

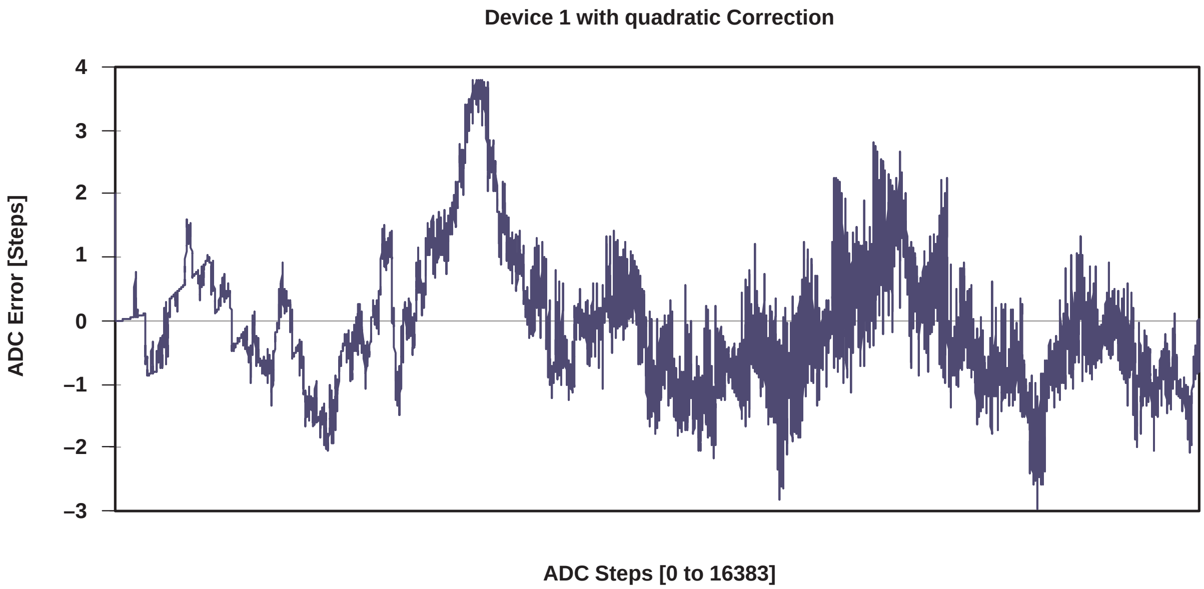


Figure 3. Error Correction With Four Quadratic Equations

Advantages: Only nine ADC measurements are necessary
No gaps at the range borders: perfect continuation
The MSP430 can calculate the correction coefficients a2 to a0

Disadvantages: Two multiplications are necessary (with Horner scheme)

1.2 Coefficients Estimation

With the maximum possible ADC error (±10 steps contained in a band of ±20 steps like shown in Figure 6) the maximum values for the coefficients a2 to a0 are shown in Table 1. Also given are the valences of the MSBs and the LSBs and the possible coefficient range. In Figure 6, the range C shows the worst case for a quadratic error curve. This curve is the basis for Table 1.

Table 1. Worst Case Coefficients for Quadratic Equations (8-Bit)

COEFFICIENT	MAXIMUM COEFFICIENT	VALENCE OF MSB (BIT 6)	VALENCE OF LSB (BIT 0)	COEFFICIENT RANGE
Quadratic coefficient a2	±6.103515E–4	2^{-11}	2^{-17}	±9.7E–4
Linear coefficient a1	±1.171875E–1	2^{-4}	2^{-10}	±1.25E–1
Constant coefficient a0	±2.00000E+1	2^{+4}	2^{-2}	±3.2000E+1

The integer calculation operates with signed 8-bit coefficients and ax ADC result rounded to 8 bits. The floating point calculation uses the full ADC result (0 to 16383) and a 32-bit format for the calculations.

To give an idea concerning the actual magnitudes, the twelve calculated correction coefficients *ax* for device 1 are:

Table 2. 8-Bit Coefficients for the Four Quadratic Equations of Device 1

COEFFICIENT	RANGE A	RANGE B	RANGE C	RANGE D
a2	9.155273E–5	–1.831055E–4	–2.746582E–5	1.517946E–4
a1	4.687500E–3	3.281250E–2	–3.0859375E–2	–2.0992208E–2
a0	6.000000E+0	1.320000E+1	9.600000E+0	–1.000000E–1

The three equations to calculate the correction coefficients a2, a1 and a0 out of the three known errors e3, e2 and e1 at the ADC steps N3, N2 and N1 are:

$$a1 = -\frac{(e_2 - e_1) \times \left(N_3^2 - N_2^2\right) - (e_3 - e_2) \times \left(N_2^2 - N_1^2\right)}{(N_2 - N_1) \times \left(N_3^2 - N_2^2\right) - (N_3 - N_2) \times \left(N_2^2 - N_1^2\right)}$$

$$a2 = -\frac{(e_2 - e_1) - a_1 \times (N_2 - N_1)}{N_2^2 - N_1^2} \qquad a0 = -\left(e1 - a2 \times N_1^2 - a1 \times N1\right)$$

NOTE:

N_3, N_2, and N_1 can be expressed in ADC steps (0...16383), range steps (0...4095) or subdivisions of the range (0...255) for 8-bit calculations. In the following, N represents subdivisions.

As shown with the linear improvements, using more than one quadratic parabola per ADC range is also possible. It is only necessary to adapt the 256 subdivisions to the sections of the ranges, to calculate the new coefficients a2 to a0, and to modify the addressing of the coefficients.

The software part after each ADC measurement is as follows. The numbers at the right border—below *int.frct*—indicate the maximum integer bits and the actual number of fraction bits for the result (integer.fraction). The Horner scheme is used for the calculation.

```
; Quadratic error correction with a single equation per range.
; 8–bit arithmetic. Cycles needed:
; Subdivision N =   0:  85 cycles
; Subdivision N > 7Fh: 206 cycles
;
;                                                              int.frct

   MOV    &ADAT,R5      ; ADC result Ni to R5                   14.0
   MOV    R5,R6         ; Address info for correction           14.0
   RRA    R5            ; Calculate subdivision 0...FFh         13.0
   RRA    R5            ; Prepare N = (Ni/4096–n)x256           12.0
   RRA    R5            ; 8 bit ADC info to low byte            11.0
   RRA    R5            ;                                       10.0
   ADC.B  R5            ; Round subdivision 0...FFh              8.0
   JNC    L$1           ; If result overflows to 100h:
   DEC.B  R5            ; Limit subdivision to FFh               8.0
;
L$1 SWPB  R6            ; Calculate coefficient address          6.0
   RRA.B  R6            ; 0...1Fh                                5.0
   RRA.B  R6            ; 0...0Fh                                4.0
   RRA.B  R6            ; 0...07h                                3.0
   BIC    #1h,R6        ; 0...06h                                3.0
```

```
   PUSH   R6            ; Save 0...6h                              3.0
   RRA.B  R6            ; 0...03h                                  2.0
   ADD    @SP+,R6       ; 0...9h (3n) pointer to a2                4.0
;
   MOV.B  R5,IROP1      ; ADC info to MPY register                 8.0
   MOV.B  TAB2(R6),IROP2L
                        ; Quadr. slope a2                          0.17
   CALL   #MPYS8        ; N x a2                                 +–0.17
   RLA    IRACL         ; To a1 format                           +–0.18
   ADD    #80h,IRACL    ; Round result                           +–0.18
   SWPB   IRACL         ;                                        +–0.10
   MOV.B  IRACL,IROP2L  ; To MPY register                        +–0.10
;
   MOV.B  R5,IROP1      ; Subdivision to MPY register              8.0
   ADD.B  TAB1(R6),IROP2L
                        ; Linear slope a1 added                  +–0.10
   CALL   #MPYS8        ; ((N x a2) + a1) x N                    +–5.10
;
   ADD    #80h,IRACL    ; Round result                           +–5.10
   SWPB   IRACL         ; To a0 format                           +–5.2
   ADD.B  TAB0(R6),IRACL
                        ; Add a0                                 +–5.2
   SXT    IRACL         ; Correction to 16 bit                   +–5.2
   RRA    IRACL         ; (((N x a2) + a1) x N) + a0             +–5.1
   RRA    IRACL         ; Carry is used for rounding             +–5.0
   ADDC   &ADAT,IRACL   ; Corrected result Nicorr                 14.0
   ...                  ; Use Nicorr in IRACL
;
; The 12 RAM bytes starting at label TAB2 contain the
; correction coefficients a2, a1 and a0 for the four ranges.
; The bytes are loaded during the initialization
;
   .bss   TAB2,1        ; Range A a2: quadr. coeff.              +–0.17
   .bss   TAB1,1        ;         a1: lin. coefficient           +–0.10
   .bss   TAB0,1        ;         a0: constant coeff.            +–5.2
   .bss   TABx,9        ; Ranges B, C, D: a2...a0.
```

EXAMPLE: The ADC is measured at the two borders and the middle of ADC range B (n = 1). The measured errors—device 1 is used—are shown below. The three correction coefficients a2, a1, and a0 for the range B are calculated with the formulas given before. The correction coefficients for the other three ranges may be calculated the same way; only the appertaining border and center errors need to be used. Twelve measurements were made for each ADC step, and the two extremes were discarded: this leads to one decimal fraction digit.

ADC Step	4096	6144	8192
Subdivision N	0	128	256
Error e [Steps]	–13.2	–13.4	–9.6

Error coefficients for the range B:

$$a2 = -0.000183106$$

$$a1 = +0.0328125$$

$$a0 = +13.2$$

For better legibility $N = \left(\frac{Ni}{4096} - n\right) \times 256$ is used in the following.

Correction:

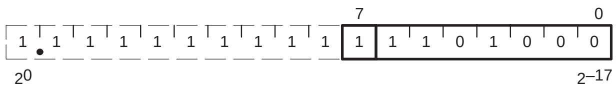
$$((N \times a2 + a1) \times N + a0) = ((N \times (-0.000183106) + 0.0328125) \times N + 13.2)$$

The correction for the ADC step 7000—located in range B—is calculated:

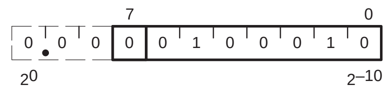
$$\left(\left(\left(\frac{7000}{4096}-1\right)\times 256\times(-0.000183106)+0.0328125\right)\times\left(\frac{7000}{4096}-1\right)\times 256+13.2\right)=+13.3$$

Corrected ADC sample: $Nicorr = Ni + 13.3$. Valid for ADC step 7000

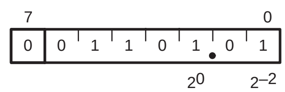
Format: a2: ±0.17 $-0.000183106/2^{-17} = -24 = \text{E8h}$



a1: ±0.10 $+0.0328125/2^{-10} = +33.6 \approx 22\text{h}$



a0: ±5.2 $+13.2/2^{-2} = +52.8 \approx 35\text{h}$



1.3 Correction With Cubic Equations

The ADC is measured at the two borders of each range (common to two ranges) and at one third and two thirds of each range, which results in 13 measurements: e.g., N = 20, 1366, 2731, and 4096 for range A. The errors of these 13 measurements are used for the calculation of four cubic equations, one for each ADC range. It is recommended to use more than one measurement for each of these thirteen points. The resulting correction coefficients a3, a2, a1, and a0 for each ADC range are stored in the RAM or EEPROM.

The above method has two advantages:

- The ranges continue smoothly at the common borders.
- Only thirteen points of the ADC characteristic need to be measured for the full ADC range during the calibration. This is due to the common range border points.

The used formula for each range with separate factors ax for an 8-bit integer calculation is:

$$Nicorr = Ni + \left(N^3 \times a3 + N^2 \times a2 + N \times a1 + a0\right)$$

Where: $N = \left(\frac{Ni}{4096} - n\right) \times 256$, the ADC result of a range adapted to the subdivisions 0...255.

Where:	Nicorr	Corrected ADC sample	[Steps]
	Ni	Measured ADC sample (non-corrected)	[Steps]
	N	Subdivision representing the ADC sample (0...255)	
	n	Range number (0...3 for ranges A...D)	
	a3	Cubic coefficient of the correction	[Steps^{-2}]
	a2	Quadratic coefficient of the correction	[Steps^{-1}]
	a1	Linear coefficient of the correction	
	a0	Offset of the correction	[Steps]
	i	Nominal ADC step of the ADC input (DAC output)	[Steps]

The integer formula above uses the subdivision (0..255) inside of an ADC range (0 to 4095) instead of the full 14-bit position (0 to 16383) of an ADC point. This is to increase the accuracy of the calculation also with limited coefficient length, e.g., for 8-bit coefficients.

If floating point calculation is used, the high resolution of the 24-bit mantissa makes the range correction unnecessary. The equation simplifies to:

$$Nicorr = Ni + \left(Ni^3 \times a3 + Ni^2 \times a2 + Ni \times a1 + a0\right)$$

To save multiplications the Horner scheme is used again. This reduces the number of multiplications from six to only three. The formula using the 8-bit arithmetic now becomes (N represents the actual subdivision 0...255. See above):

$$Nicorr = Ni + (((N \times a3) + a2) \times N + a1) \times N + a0$$

The formula for 16-bit and floating point calculations now becomes:

$$Nicorr = Ni + (((Ni \times a3) + a2) \times Ni + a1) \times Ni + a0$$

Figure 4 shows the principle of the correction with four cubic equations: the correction parabolas actually used are printed together with the corrected and non-corrected ADC characteristic. The circles indicate the measured ADC points.

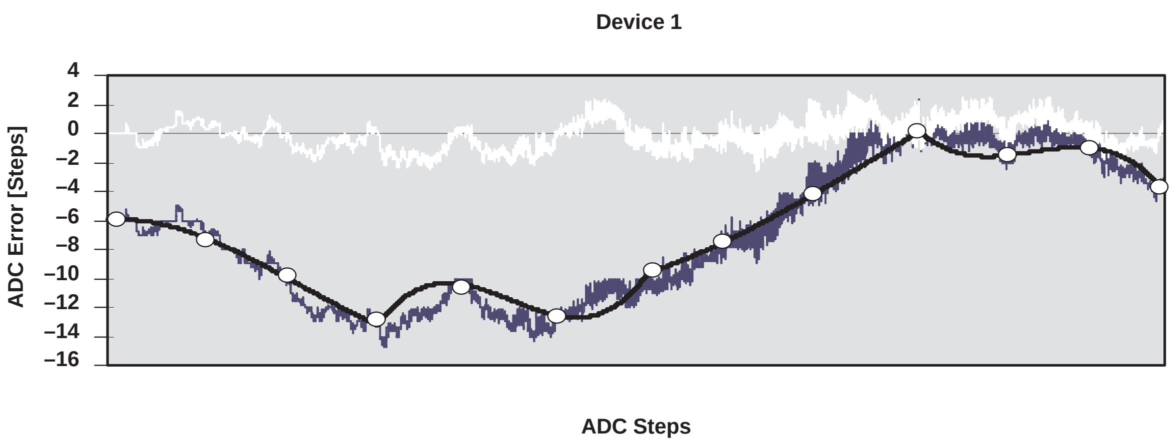


Figure 4. Principle of the Error Correction With Four Cubic Equations

The statistical results for the cubic correction method are:

	Full range	**Ranges A and B only**
Mean Value:	–0.10 Steps	–0.28 Steps
Range:	5.47 Steps	4.97 Steps
Standard Deviation:	0.93 Steps	0.97 Steps
Variance:	0.87 Steps	0.94 Steps

Figure 5 shows the resulting error correction in a graph:

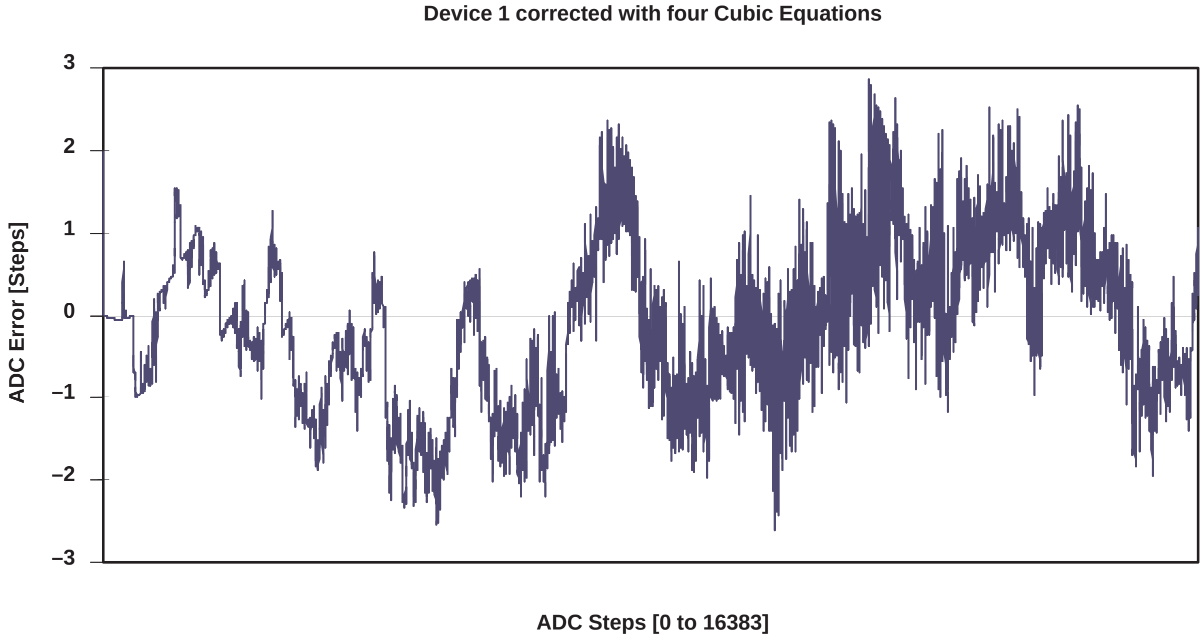


Figure 5. Error Correction With Four Cubic Equations

Advantages:

Good adaptation to worst case ADC characteristics
Low storage needs: 16 bytes RAM or EEPROM (integer calculation)
Monotonicity is ensured due to common samples at the range borders
Only thirteen ADC measurements are necessary for the calibration

Disadvantages:

Three multiplications are necessary (with HORNER scheme)
Host is necessary for the calculation of the correction coefficients ax

1.4 Coefficients Estimation

With the maximum possible ADC error (±10 steps contained in a band of ±20 steps like shown in Figure 6) the maximum values for the coefficients a3 to a0 are shown in Table 3 (8-bit arithmetic). Also given are the valences of the MSB and the LSB and the possible coefficient range. In Figure 6, the range D shows the worst case of a cubic error curve. This curve is the basis for Table 3.

Table 3. Worst Case Cubic Coefficients (8-Bit Arithmetic)

COEFFICIENT	MAXIMUM COEFFICIENT	VALENCE OF MSB (BIT 6)	VALENCE OF LSB (BIT 0)	COEFFICIENT RANGE
Cubic coefficient a3	±6.357828E–6	2^{-18}	2^{-24}	±7.57E–6
Quadratic coefficient a2	±2.441406E–3	2^{-9}	2^{-15}	±3.88E–3
Linear coefficient a1	±2.473958E–1	2^{-3}	2^{-9}	±2.48E–1
Constant coefficient a0	±2.00000E+1	2^{+4}	2^{-2}	±3.200E+1

The integer calculation operates with signed 8-bit coefficients and an ADC result rounded to 8 bits (256 subdivisions).

The floating point calculation uses the full ADC result (0 to 16383) and a 32-bit format for the calculations. To give an example, for device 1 the calculated sixteen correction factors a3 to a0 are (12-bit ADC info is used):

Table 4. Correction Coefficients for the Cubic Equations of Device 1

COEFFICIENT	RANGE A	RANGE B	RANGE C	RANGE D
a3	–1.18E–10	–6.483598E–10	3.286326E–11	5.17398E–10
a2	1.02E–06	3.943417E–06	–3.497543E–07	–2.655711E–06
a1	–4.37E–04	–6.153471E–03	–1.486924E–03	3.83333E–03
a0	6.00E+00	1.320000E+01	9.600000E+00	–1.000000E–01

The algorithm to calculate the four correction coefficients a3 to a0 out of the four measured errors e4, e3, e2 and e1 at the ADC steps N4, N3, N2 and N1 is very complex. It is recommended to use a mathematical support software running on a host computer for this task. A simple calculation software routine is available from Texas Instruments on request.

For the cubic correction an example using the MSP430 Floating Point Package FPP4 is given below. This software example can be adapted easily to linear and quadratic correction:

- The parts not used are deleted (e.g., the parts handling the coefficients a3 and a2 if a linear correction is needed)
- The calculation of the start address of the correction coefficients (address of a3 in the example) out of the ADC result is modified slightly.

```
; Cubic error correction with a single equation per range.
; Floating point arithmetic. Cycles needed: 800 to 2400
;
DOUBLE .EQU   0            ; Use .FLOAT format (32 bit)
;
       MOV    #xxx,&ACTL   ; Define ADC measurement
       CALL   #MEASR       ; Measure. Result Ni to ADAT
       CALL   #FLT_SAV     ; Save registers R5 to R12
       SUB    #4,SP        ; Allocate stack for FP result
       MOV    #ADAT,RPARG  ; Load address of ADC buffer
       CALL   #CNV_BIN16U  ; Convert ADC result Ni to FP
       SUB    #4,SP        ; New working space for calc.
;
       MOV    &ADAT,R15    ; Calc. address of coeff. a3
       SWPB   R15
       AND    #0030h,R15   ; Range x 16: rel. address a3
       ADD    #a3,R15      ; Start address of coeff. block
       MOV    R15,RPARG    ; Points to actual a3
       CALL   #FLT_MUL     ; a3 x Ni
       ADD    #a2–a3,R15   ; Address of a2
       MOV    R15,RPARG    ; Points to actual a2
       CALL   #FLT_ADD     ; a3 x Ni + a2
       ADD    #4,RPARG     ; To Ni
       CALL   #FLT_MUL     ; (a3 x Ni+a2)Ni
       ADD    #a2–a3,R15   ; Address of a1
       MOV    R15,RPARG    ; Points to actual a1
       CALL   #FLT_ADD     ; ((a3 x Ni)+a2)Ni + a1
       ADD    #4,RPARG     ; To Ni
       CALL   #FLT_MUL     ; (((a3 x Ni) + a2)Ni + a1)Ni
       ADD    #a2–a3,R15   ; To actual a0
       MOV    R15,RPARG
       CALL   #FLT_ADD     ; (((a3 x Ni)+a2)Ni+a1)Ni+a0
       ADD    #4,RPARG     ; To Ni
       CALL   #FLT_ADD     ; Nicorr = Ni + correction
;
```

```
       POP    2(SP)         ; Result to top of stack
       POP    2(SP)         ; POPs correct the stack
       ...                  ; Continue calc. with Nicorr
       CALL   #FLT_REC      ; Restore registers R12 to R5
       ...                  ; Normal program continues
;
; Correction coefficients are loaded from EEPROM during init.
;
       .bss   a3,4          ; Range A: Cubic coefficient a3
       .bss   a2,4          ; Quadratic coefficient a2
       .bss   a1,4          ; Linear coefficient a1
       .bss   a0,4          ; Constant coefficient a0
;
       .bss   ax,48         ; Ranges B, C and D: a3...a0
```

The assembler software part after each ADC measurement is as follows. The numbers at the right border—below int.frct—indicate the maximum integer bits and the maximum number of fraction bits (integer.fraction). The Horner scheme is used for the calculation.

```
; Cubic error correction with a single equation per range.
; 8–bit arithmetic. Cycles needed:
; Subdivision N =  0:  108 cycles; Subdivision N > 7Fh: 283 cycles
;
;                                                              int.frct
       MOV    &ADAT,R5         ; ADC result Ni to R5            14.0
       MOV    R5,R6            ; Address info for correction    14.0
       RRA    R5               ; Calculate subdivision 0...FFh  13.0
       RRA    R5               ; Prepare N = (Ni/4096–n)x256    12.0
       RRA    R5               ; 8 bit ADC info to low byte     11.0
       RRA    R5               ;                                10.0
       ADC.B  R5               ; Round subdivision 0...FFh       8.0
       JNC    L$1              ; If result overflows to 100h:
       DEC.B  R5               ; Limit subdivision to FFh        8.0
;
L$1    SWPB   R6               ; Calculate coeff. address a3     6.0
       BIC.B  #0Fh,R6          ; 0...30h                         6.0
       RRA.B  R6               ; 0...18h                         5.0
       RRA.B  R6               ; 0...0Ch: address of slope a3    4.0
;
       MOV.B  R5,IROP1         ; Subdivision to MPY register     8.0
       MOV.B  TAB3(R6),IROP2L  ; Cubic slope a3                  0.24
       CALL   #MPYS8           ; N x a3                        +–0.24
       SWPB   IRACL            ;                               +–0.16
       RRA.B  IRACL            ; To a2 format                  +–0.15
       ADC.B  IRACL            ; Round result                  +–0.15
       MOV.B  IRACL,IROP2L     ; To MPY register               +–0.15
;
       MOV.B  R5,IROP1         ; Subdivision to MPY register    8.0
       ADD.B  TAB2(R6),IROP2L  ; Quadr. slope a2 added         +–0.15
       CALL   #MPYS8           ; ((N x a3 + a2) x N            +–0.15
       RLA    IRACL            ; To a1 format                  +–0.16
       RLA    IRACL            ;                               +–0.17
       ADD    #080h,IRACL      ; Round result                  +–0.17
       SWPB   IRACL            ;                               +–0.9
       MOV.B  IRACL,IROP2L     ; To MPY register               +–0.9
;
       MOV.B  R5,IROP1         ; Subdivision to MPY register     8.0
       ADD.B  TAB1(R6),IROP2L  ; Linear slope a1 added           0.9
       CALL   #MPYS8           ; ((N x a3 + a2) x N + a1) x N  +–5.9
;
       RLA    IRACL            ; To a0 format                  +–5.10
       ADD    #80h,IRACL       ; Round result                  +–5.10
       SWPB   IRACL            ; To a0 format                  +–5.2
       ADD.B  TAB0(R6),IRACL   ; Add a0                        +–5.2
```

```
        SXT    IRACL             ; Correction to 16 bit           +–5.2
        RRA    IRACL             ;                                +–5.1
        RRA    IRACL             ; Carry is used for rounding     +–5.0
        ADDC   &ADAT,IRACL       ; Corrected result Nicorr         14.0
        ...                      ; Use Nicorr in IRACL
;
; The 16 RAM bytes starting at label TAB3 contain the
; correction coefficients a3, a2, a1 and a0. The bytes are
; loaded during the initialization
;
        .bss   TAB3,1            ; Range A a3: cubic coeff.       +–0.24
        .bss   TAB2,1            ;         a2: quadr. coeff.      +–0.15
        .bss   TAB1,1            ;         a1: lin. coefficient   +–0.9
        .bss   TAB0,1            ;         a0: constant coeff.    +–5.2
        .bss   TABx,12           ; Ranges B, C, D: a3...a0
```

As shown with the linear improvements, it is also possible to use more than one cubic parabola per ADC range. It is only necessary to adapt the 256 subdivisions to the sections of the ranges, to calculate the new coefficients, and to modify the addressing of the coefficients.

EXAMPLE: The ADC is measured at the borders and at one third and two thirds of ADC range D (n = 3). The measured errors—device 1 is used—are shown below. The four cubic correction coefficients a3 to a0 for the range D are calculated with a math package running on a PC. The correction coefficients for the other three ranges may be calculated the same way using the appertaining errors of each range. Twelve measurements were made for each ADC step, the two extremes were discarded: this leads to one decimal fraction digit.

ADC Step	1228	13653	15019	16350
Subdivision N	0	85.33	170.67	253.9
Error e [Steps]	0.1	–1.5	–1.1	–4.0

Error coefficients for the range D:

$$a3 = +0.00000146501$$

$$a2 = -0.000512371$$

$$a1 = +0.0518045$$

$$a0 = -0.10$$

For better legibility $N = \left(\frac{Ni}{4096} - n\right) \times 256$ is used in the following.

Correction: $(((N \times a3) + a2) \times N + a1) \times N + a0$

$$= (((N \times 0.00000146501) - 0.000512371) \times N + 0.0518045) \times N - 0.10$$

The correction for the ADC step 15000—located in range D—is calculated:

$$N = \left(\frac{15000}{4096} - 3\right) \times 256 = 169.5 \approx 170$$

$$(((170 \times 0.00000146501) - 0.000512371) \times 170 + 0.0518045) \times 170 - 0.10 = +1.1$$

Corrected ADC sample: $Nicorr = Ni + 1.1$. Valid for ADC step15000

Format: a3: ±0.24 $+0.000146501/2^{-24} \approx +24.58 = 19h$

2^0																	7							0 (2^{-24})
0.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1

a2: ±0.15 $-0.000512371/2^{-15} = -16.79 \approx EFh$

2^0								7							0 (2^{-15})
1.	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1

a1: ±0.9 $+0.0518045/2^{-9} = +26.52 \approx 1Bh$

2^0		7							0 (2^{-9})
0.	0	0	0	0	1	1	0	1	1

a0: ±5.2 $-0.1/2^{-2} = -0.4 \approx 00h$

7					2^0		0 (2^{-2})
0	0	0	0	0	0.	0	0

2 Considerations to the Integer Calculations

The calculations for this application report were made with a floating point package. If the 14-bit ADC is used within a real time system the floating point calculation time is normally too long. Therefore the necessary loss of accuracy needs to be known if integer calculations with their restricted bit length are used. The most time consuming parts are the multiplication subroutines, so they are shown first.

2.1 Multiplication Subroutines

To reduce the multiplication time as much as possible, two multiplication subroutines, which terminate immediately after the operand IROP1 becomes zero are shown; this means that the operand with leading zeroes should be in the register IROP1—here the subdivision representing the ADC result.[1]

2.1.1 8-Bit Multiplication Subroutine

If the operands of the multiplication subroutine are normally shorter than 8 bits, then the multiplication subroutine below saves time due to its run time optimization: the multiplication terminates immediately after IROP1 gets zero due to the right shifts during the processing.

```
; Run time optimized 8–bit Multiplication Subroutines
; Definitions
;
IROP1     .EQU   R14          ; Unsigned subdivision (00h...FFh)
IROP2L    .EQU   R13          ; Signed coefficient   (80h...7Fh)
IRACL     .EQU   R12          ; Result word
;
; Cycles for specific registers contents without CALL:
; TASK        MACU8  MACS8  MPYS8  IROP1  IROP2  Result (MPYS8)
;-------------------------------------------------------------
; MINIMUM      9     12     13     000h x 000h = 0000h
; MEDIUM      34     37     38     00Fh x 00Fh = 00E1h
; MAXIMUM     66     70     71     0FFh x 0FFh = FF01h
;
; Used registers IROP1, IROP2L, IRACL
;
; Signed multiply subroutine: IROP1 x IROP2L –> IRACL
;
MPYS8  CLR    IRACL           ; 0 –> 16 bit RESULT
;
; Signed multiply–and–accumulate subroutine:
; (IROP1 x IROP2L) + IRACL –> IRACL
;
MACS8  TST.B  IROP2L          ; Sign of factor
       JGE    MACU8           ; Positive sign: proceed
       SWPB   IROP1           ; Negative sign: correction nec.
       SUB    IROP1,IRACL     ; Correct result word
       SWPB   IROP1
;
; Unsigned multiply–and–accumulate subroutine (MAC):
; (IROP1 x IROP2L) + IRACL –> IRACL
;
MACU8  BIT.B  #1,IROP1        ; Test actual bit (LSB)
       JZ     L$01            ; If 0: do nothing
       ADD    IROP2L,IRACL    ; If 1: add multiplier to resultL$01
L$01   RLA    IROP2L          ; Double multiplier IROP2
       RRC.B  IROP1           ; Next bit of IROP1 to LSB
       JNZ    MACU8           ; If IROP1 = 0: finished
       RET
```

[1] The idea for these subroutines initially came from Leslie Mable of TIL.

2.1.1.1 16-Bit Multiplication Subroutine

This multiplication subroutine is used if the 8-bit version is not accurate enough. Like the 8-bit version, the multiplication terminates immediately after IROP1 becomes zero due to the right shifts during the processing. All of the shown ADC improvement methods may be adapted to the 16-bit multiplication subroutine.

```
; Run time optimized 16–bit Multiplication Subroutines
;
IROP1  .EQU   R11           ; Unsigned ADC result (0000h...3FFFh)
IROP2L .EQU   R12           ; Signed coefficient  (8000h...7FFFh)
IROP2M .EQU   R13           ; High word of signed factor (0)
IRACL  .EQU   R14           ; Result word low
IRACM  .EQU   R15           ; Result word high
;
; Cycles for specific register contents without CALL:
; TASK        MACU   MACS   MPYS   IROP1  IROP2  Result (MPYS)
;–––––––––––––––––––––––––––––––––––––––––––––––––––––––––––––
; MINIMUM      11     14     16    0000h x 0000h = 00000000h
; MEDIUM       83     86     88    00FFh x 00FFh = 0000FE01h
; MAXIMUM     143    147    149    3FFFh x FFFFh = FFFFC001h
;
; Used registers: all of the above ones
;
; Signed multiply subroutine: IROP1 x IROP2L –> IRACM|IRACL
;
MPYS   CLR    IRACL         ; 0 –> result word low
       CLR    IRACM         ; 0 –> result word high
MACS   TST    IROP2L        ; Sign of factor a1
       JGE    MACU          ; Positive sign: proceed
       SUB    IROP1,IRACM     ; Correct result
MACU   CLR    IROP2M        ; Clear MSBs of multiplier
L$002  BIT    #1,IROP1      ; Test actual bit (LSB)
       JZ     L$01          ; If 0: do nothing
       ADD    IROP2L,IRACL    ; If 1: add multiplier to result
       ADDC   IROP2M,IRACM
L$01   RLA    IROP2L        ; Double multiplier IROP2
       RLC    IROP2M        ;
;
       RRC    IROP1         ; Next bit of IROP1 to LSB
       JNZ    L$002         ; If IROP1 = 0: finished
       RET
```

2.2 Maximum Magnitude of the 8-Bit Coefficients

To get the maximum accuracy with the limited 8-bit format used for the correction coefficients, it is necessary to calculate the worst case magnitude for each one of these coefficients. The basis for this calculation is the maximum error of the 14-bit ADC:

±10 steps within a band of ±20 steps

Figure 6 shows examples for the worst case errors of the 14-bit ADC:

- Range A shows the maximum error band of the ADC: ±20 steps; within this error band all errors of different devices are contained. The four dark boxes indicate four possible error ranges of ±10 steps: they are examples for single devices, within such an error band the errors of a single device are contained.
- Range B gives an example for the maximum linear error: within one range (4096 steps) the error changes by 20 steps.
- Range C is an example for a maximum quadratic error.
- Range D is an example for a maximum cubic error. This means within an ADC range the ADC characteristic moves from an error of +10 steps at the lower

range border to +2.5 then back to +7.5 and back to 0 steps at the upper range border.

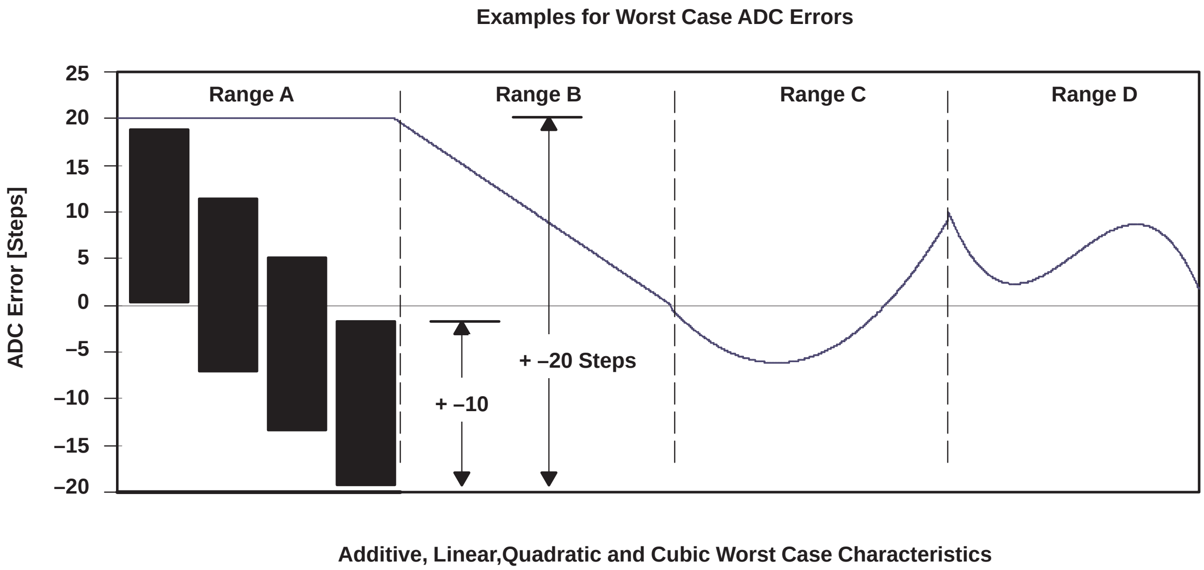


Figure 6. Worst Case ADC Error With Different Improvement Methods

Table 5 shows the worst case values—the largest possible values—for the 8-bit correction coefficients that were calculated with the following assumptions:

- The ADC characteristic uses the full error band of ±10 steps.
- The ADC characteristic changes its direction as often as the order of the correction formula, e.g., twice for a quadratic correction.
- The correction is made for each range individually; this means the ADC result bits 13 and 12—the bits defining the ADC range—are cleared.
- The relative ADC result within each range (12 bits) is rounded to eight bits (0 to FFh) for the calculations (8-bit arithmetic).
- The linear coefficient a1 of each method must allow a ±10 step correction.
- The constant coefficient a0 of each method must allow a ±20 step correction.

The maximum correction coefficients calculated with the above assumptions are listed in Table 5. (NA means not applicable):

Table 5. Worst Case Correction Coefficients (8-Bit Arithmetic)

	CUBIC CORRECTION	QUADDRATIC CORRECTION	LINEAR CORRECTION	ADDITIVE CORRECTION
a3	±6.357828E–6	NA	NA	NA
a2	±2.441406E–3	±6.103515E–4	NA	NA
a1	±2.473958E–1	±1.171875E–1	±1.562500E–1	NA
a0	±2.000000E+1	±2.000000E+1	±2.000000E+1	±2.000000E+1

2.3 Number Formats of the 8-Bit Coefficients

The format chosen for the correction data is byte format due to its low storage needs and the speed advantages for multiplications. Figure 7 shows a signed 8-bit number with three fractional bits (±4.3). The range of this number format is –16.00 to +15.875.

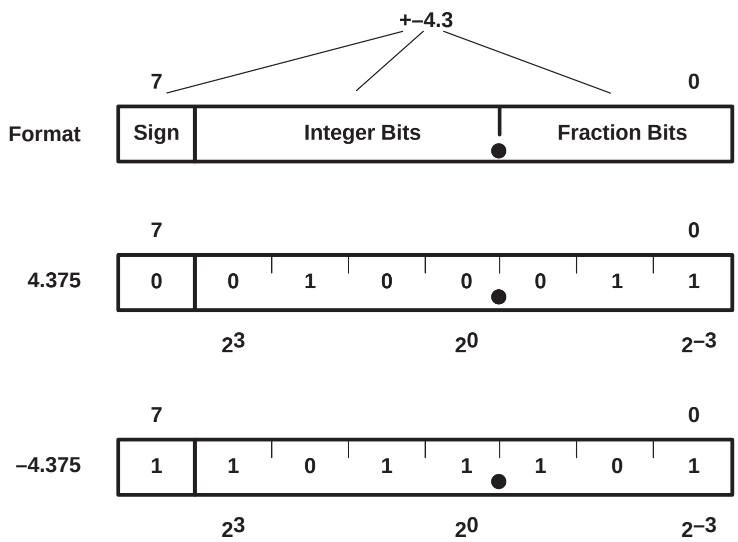


Figure 7. Number Format With Integers for 8-Bit Calculations

For the coefficients a3 and a2 no integer parts exist, due to the small values of the resulting numbers. This makes a different format necessary, but the philosophy is the same, to pack very small numbers with many leading zeros or ones into a single byte. Figure 8 shows the number format of the quadratic coefficient a2 used in a cubic correction. All bits of the extended sign have the same value as the sign bit (bit 7).

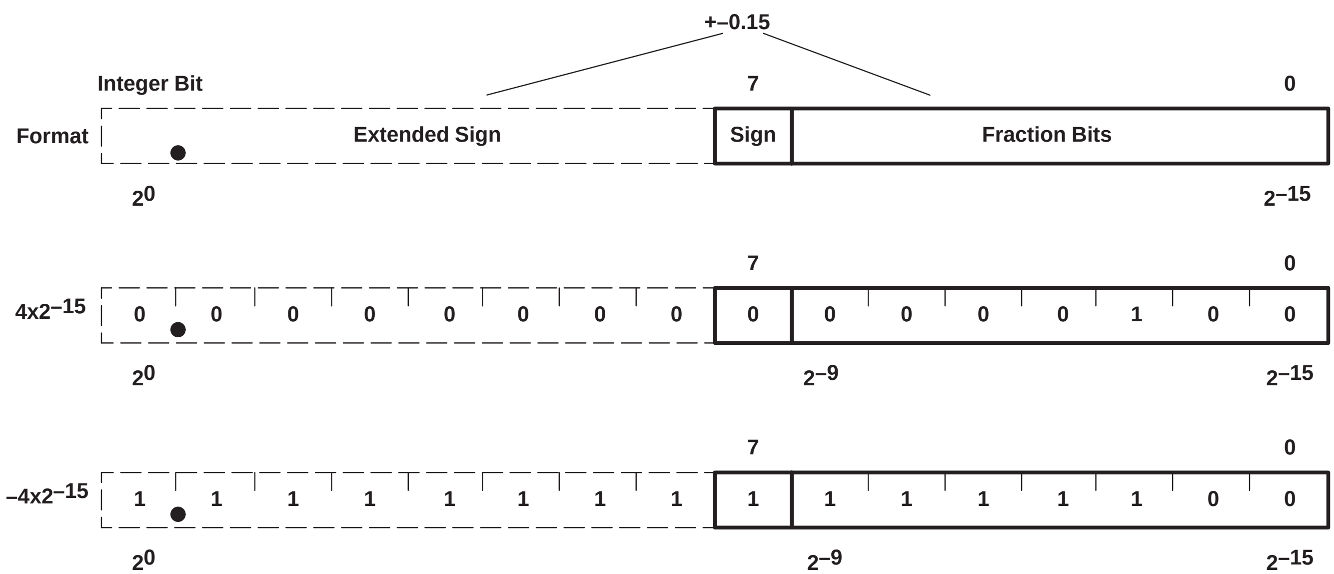


Figure 8. Number Format With Fraction Part Only for 8-Bit Calculations

2.4 Calculation of the 8-Bit Coefficients

To get the subdivision N out of the ADC value—ranging from 0 to 3FFFh—a short calculation is necessary:

$$N = \left(\frac{Ni}{4096} - n\right) \times 128 \quad \text{ranging from 0 to 128 for linear correction}$$

$N = \left(\frac{Ni}{4096} - n\right) \times 256$ ranging from 0 to 256 for quadratic and cubic correction

With two, three, or four subdivisions N—dependent on the used correction method—the coefficients ax are calculated. See the appropriate sections.

1. To start, it is necessary to find the minimum valence—a power of 2—of bit 6 (MSB) of the 8-bit number that is sufficient for the worst case value of the coefficient ax. The formula for this calculation is:

$$V_{MSB} \geq log2|ax|–1$$

Where:	V_{MSB}	Valence for the MSB (bit 6) of the 8-bit number
	V_{LSB}	Valence for the LSB (bit 0) of the 8-bit number
	ax	Decimal correction coefficient (a3 to a0)

The above formula ensures that the worst case value of the coefficient ax fits into an 8-bit twos complement number.

2. The valence V_{LSB} of the LSB is for 8-bit arithmetic

$$V_{LSB} = V_{MSB} - 6$$

With this valence V_{LSB} the 8-bit coefficient ax8bit is calculated:

$$ax8bit = \frac{ax}{2^{V_{LSB}}}$$

3. The result ax8bit—which is also named ax in the following equations—is converted into a signed hexadecimal number using the twos complement format:

- A positive coefficient is simply converted.
- A negative coefficient is converted and negated afterwards (complemented and incremented).

EXAMPLE: The worst case value for the cubic correction coefficient a3 is ±6.357828E–6 (see Table 5). To find the minimum valence of the MSB of the number format the equation above is used:

$$V_{MSB} \geq log2|ax| - 1 = log2\ 6.357828E - 6\ - 1 = -17.263 - 1 = -18.263$$

$$V_{MSB} \geq -18.263 \rightarrow V_{MSB} = -18$$

This result means that bit 6—the MSB—of the 8-bit coefficient a3 must have a minimum valence of $2^{–18}$. For the LSB the valence V_{LSB} becomes:

$$V_{LSB} = V_{MSB} - 6 = -18–6 = -24$$

This means, a3 can cover the number range from $–128 \times 2^{–24}$ to $+127 \times 2^{–24}$ (–7.57E–6 to +7.63E–6) in steps of $2^{–24}$ (5.96E–8).

Calculation: a3: ±0.24 $+6.357828E–6/2^{–24} = +106.67 \approx$ 6Bh

This means the worst case of the a3 value results in 107 steps out of 127 steps: good resolution and enough reserve are given. The number format—shown for the negative worst case value of a3 (–107 = 95h)—is:

																	7							0
1.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1	0	1
2^0																								2^{-24}

The bits 2^0 to 2^{-17} for the above example always have the same value: they contain the extended sign: the same value as the sign bit in bit 7 of the 8–bit value (2s complement arithmetic):

- Zero for a positive coefficient
- One for a negative coefficient

Information is contained only in the bits 7 to 0 (2^{-18} to 2^{-24} for the above example). This is possible due to the known maximum value of these coefficients.

2.5 Accuracy With the 8-Bit Integer Routines

To show the loss of accuracy when moving from floating point to integer calculations with 8-bit coefficients, the results of the linear and the cubic correction are given.

2.5.1 Accuracy for the Linear Correction

The linear correction—which is not sensitive to coefficient truncation due to the simple algorithm—is shown with both calculation methods. The correction coefficients a1 and a0 of the linear equation shown in section 1.2.1.1, single linear equation per range (with border fit) of *Linear Improvement of the MSP430 14-Bit ADC Characteristic*, SLAA048, [4], were recalculated to fit into signed 8-bit constants with their restricted resolution. With these 8-bit coefficients the calculations were repeated. The statistictical results in comparison to the floating point results are (full range, 8-bit results after rounding):

	32-Bit Floating Point	**8-Bit Integer Calculations**
Mean Value:	–0.32 Steps	–0.32 Steps
Range:	5.6 Steps	6.0 Steps
Standard Deviation:	0.94 Steps	0.98 Steps
Variance:	0.88 Steps	0.96 Steps

Figure 9 compares the corrected ADC characteristics by floating point calculation vs 8-bit arithmetic (integer result). The difference of the corrected characteristics (FPP result—8-bit result) is displayed also:

- The white, scribbled line indicates the result of the correction using floating point calculations
- The black, scribbled line indicates the result of the correction with 8–bit arithmetic
- The black line below the above two lines indicates the difference between the two corrections using the floating point and the 8-bit arithmetic. The offset is chosen as –4 steps, which means –4 steps represent the zero line of the difference

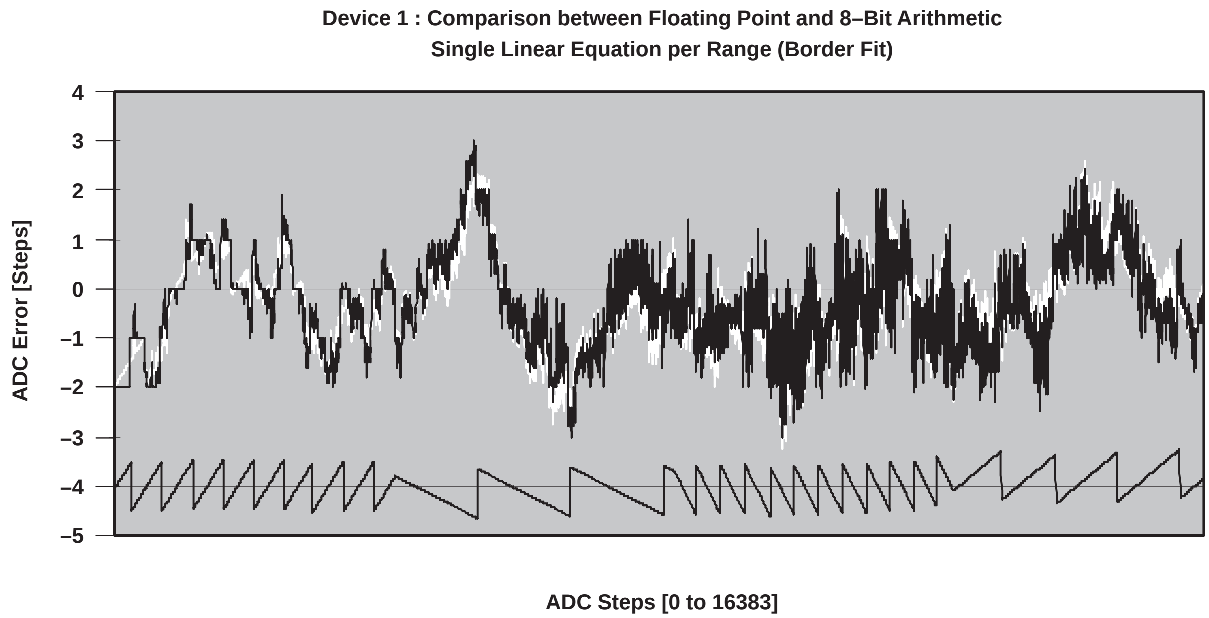


Figure 9. Comparison of Corrected ADC Characteristics. 8-Bit Results After Rounding

As can be seen, the loss of accuracy is not critical (max. ±0.5 steps), the statistical values are nearly identical. This proves that the 8-bit arithmetic is useful for this improvement method due to its speed and storage advantages.

The difference between the floating point and the 8-bit arithmetic looks even better if the 8-bit result is used before the rounding: the two fraction bits of the calculation are used as well. Figure 10 shows this:

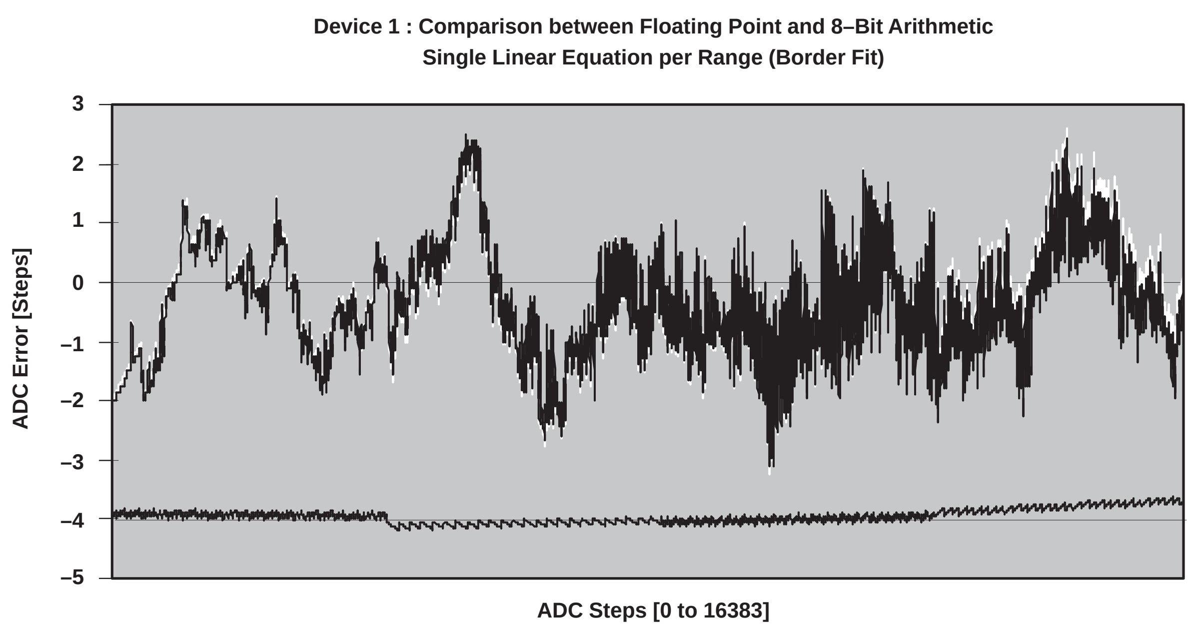


Figure 10. Comparison of Corrected ADC Characteristics. 8-Bit Results before Rounding

Nearly no difference now exists between the floating point and the 8-bit results.

2.5.2 Accuracy for the Cubic Correction

The cubic correction—which is the most sensitive one to coefficient length due to the third power multiplication—is also shown with both calculation methods.

The correction coefficients a3 to a0 of the cubic equations shown in Table 4 were recalculated to fit into signed 8-bit constants with their restricted resolution. With these 8-bit coefficients the calculations were repeated. The results in comparison to the floating point results are (full range):

	32-Bit Floating Point	**8-Bit Integer Calculations**
Mean Value:	–0.10 Steps	–0.17 Steps
Range:	5.47 Steps	6.25 Steps
Standard Deviation:	0.93 Steps	1.03 Steps
Variance:	0.87 Steps	1.06 Steps

Figure 11 compares the corrected ADC characteristics by floating point calculation vs 8-bit arithmetic. The difference of the corrected characteristics (FPP result—8-bit result) is displayed as well:

- The white, scribbled line indicates the result of the floating point calculations
- The black, scribbled line indicates the result of the 8-bit arithmetic
- The black line below indicates the difference between floating point and 8-bit arithmetic. The difference is shown before the rounding of the result (two binary digits). The offset is –6 steps, which means –6 steps represent the zero line of this difference

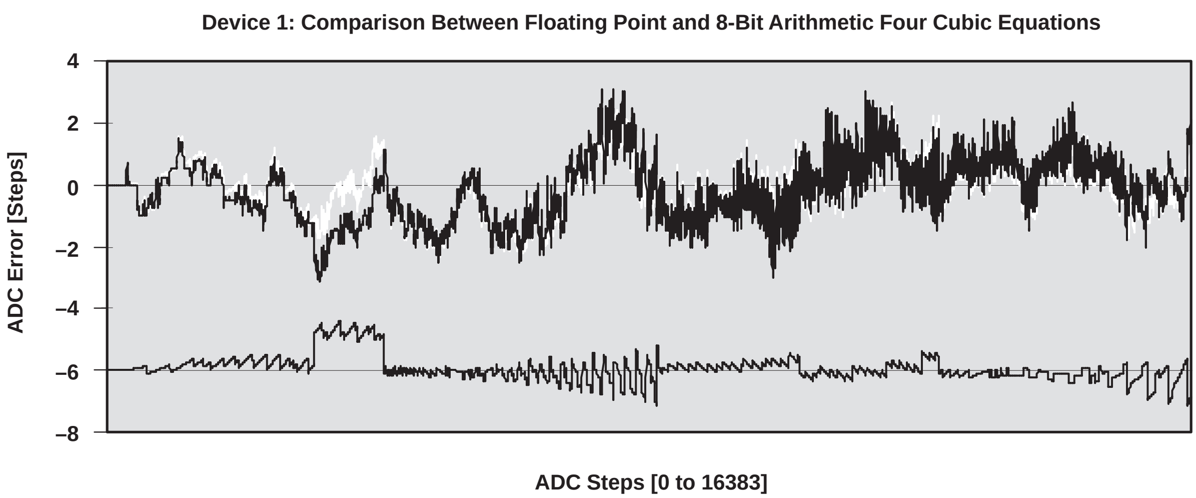


Figure 11. Comparison of Corrected ADC Characteristics. 8-Bit Results Before Rounding

The loss of accuracy is not critical (±1 LSB), but higher than with the linear improvement method. The statistical values are nearly identical to the floating point results.

3 Comparison of the Used Improvement Methods

This section gives an overview to the possible improvements including the effort that is needed to implement them (RAM, ROM, number of measurements during the calibration, calculation time).

3.1 Comparison Tables

Table 6 gives the statistical results for all of the previously described improvement methods. The definitions of the statistical values are given in the application report *Additive Improvement of the MSP430 14-Bit ADC Characteristic*.[3] The most important value of Table 6 is the range: it indicates the worst case value for the error of the ADC (here for device 1). For example, a range of 6.0 means, that the maximum difference of errors is 6.0. All values are ADC steps.

Table 6. Comparison Table for the Different Improvement Methods

CORRECTION METHOD	MEAN VALUE	RANGE	STANDARD DEVIATION	VARIANCE
Additive Corrections:				
Mean value of full range	–0.44	17.1	4.74	22.51
Mean value of 4 ranges	–0.31	13.5	2.49	6.20
Center of ranges	0.20	13.5	2.56	6.53
Multiple sections (8 sections)	–0.14	8.40	1.47	2.16
(16 sections)	–0.29	6.40	1.04	1.08
(32 sections)	0.14	5.20	0.77	0.59
(64 sections)	–0.08	4.60	0.64	0.41
Linear Equations with Border Fit:				
Single linear equation per range	–0.32	5.60	0.94	0.88
Two linear equations per range	–0.29	6.49	0.97	0.94
Four linear equations per range	–0.22	5.36	0.83	0.69
Linear Equations with Linear Regression:				
Single linear equation per range	0.03	5.09	0.94	0.88
Multiple linear equations per range (2)	–0.03	4.84	0.78	0.61
Correction with Quadratic Equations	–0.27	6.96	1.14	1.29
Correction with Cubic Equations:				
8-Bit calculation	–0.17	6.25	1.03	1.06
Floating point calculation	–0.10	5.47	0.93	0.87

Table 7 gives the memory (RAM, ROM, EEPROM) requirements and the necessary number of CPU cycles for all improvement methods. The meaning of the four columns is:

- **RAM/EEPROM:** The number of bytes in the RAM or an external EEPROM that are needed for the continuous storage of the correction coefficients if 8-bit arithmetic is used (full range). It indicates words, if 16-bit arithmetic is used for the calculations. If the correction coefficients are stored in an external memory (e.g., EEPROM), then only a part of this number (the coefficients actually used) need RAM space. If the current source is used, then only one half of the given number is needed (for the ranges A and B only).
- **ROM:** The number of ROM bytes needed for the correction algorithm. The multiplication subroutine is not included in this number.

- **Cycles:** The number of CPU cycles needed for the calculation of the correction.
- **Calibration Samples:** The number of ADC samples that are needed for the used improvement method. The number of actual measurements for each sample is not included in this number. See *Measurement Methods for the ADC Reference Samples* in the application report *Additive Improvement of the MSP430 14-Bit ADC Characteristic*[3] for examples.

Table 7. Comparison Table for the Different Improvement Methods

CORRECTION METHOD	RAM EEPROM BYTES	ROM BYTES	CYCLES	CALIBRATION SAMPLES
Additive Corrections:				
Mean value of full range	2	10	7	16...64
Mean value of 4 ranges	4	24	16	16...64
Center of ranges	4	24	16	4
Multiple sections (8 sections)	8	22	13	9
(16 sections)	16	20	12	17
(32 sections)	32	18	11	33
(64 sections)	64	18	11	65
Linear Equations with Border Fit:				
Single linear equation per range	8	60	51...100	5
Two linear equations per range	16	64	48...97	9
Four linear equations per range	32	56	49...101	17
Linear Equations with Linear Regression:				
Single linear equation per range				
8-Bit calculation	8	60	51...100	16...64
16-Bit calculation	16	44	47...178	16...64
Multiple linear equations per range (2)	16	54	48...97	32...128
Correction with Quadratic Equations	12	84	85...206	9
Correction with Cubic Equations:				
8-Bit calculation	16	102	108...283	13
Floating point calculation	64	88	800...2400	13

3.2 Comparison Graph

To give an impression of how the discussed improvement methods perform, Figure 12 shows the original (non-corrected) characteristic of device 1 and the best improvement in one figure: it is the additive correction with 64 sections described in the application report *Additive Improvement of the MSP430 14-Bit ADC Characteristic*:[3] 64 sections of the ADC range are corrected by the addition of individual constants.

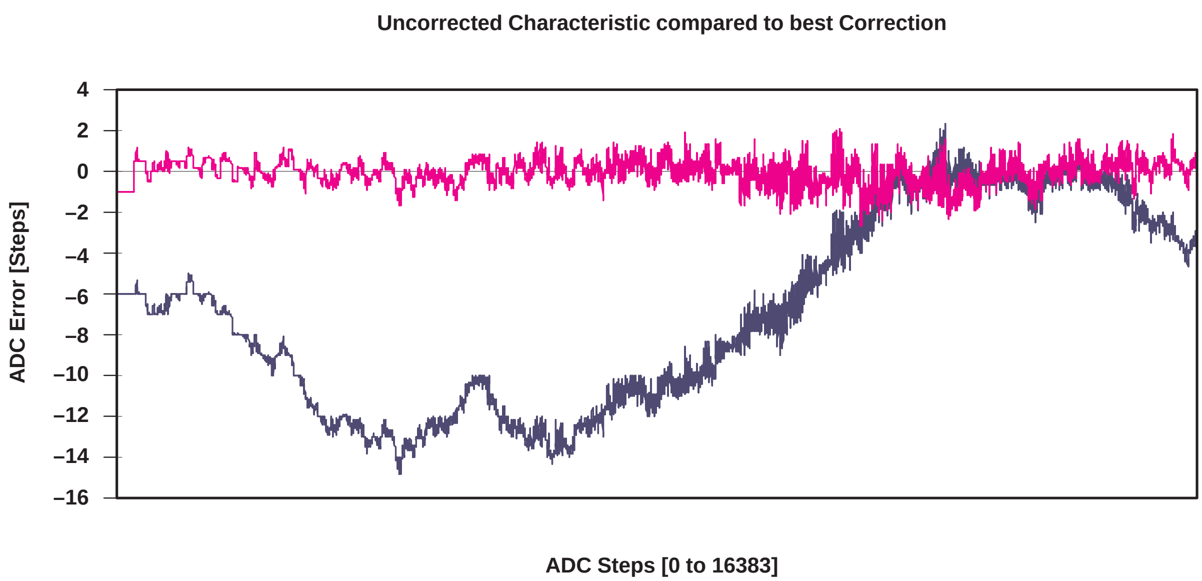


Figure 12. Comparison of the Non-Corrected ADC Characteristic and the Best Improvement

As can be seen, the non-corrected range (17 steps) reduces to a range of less than 5 steps. The statistical results are (full range):

- Best correction: Additive correction of 64 sections for the full ADC range
- Second best correction: Linear regression with two equations per ADC range

	Non-Corrected Device 1	Best Correction	2nd Best Corrected
Mean Value:	–6.95 Steps	–0.08 Steps	–0.03 Steps
Range:	17 Steps	4.60 Steps	4.84 Steps
Standard Deviation:	4.74 Steps	0.64 Steps	0.78 Steps
Variance:	22.51 Steps	0.41 Steps	0.61 Steps

4 Selection Guide

To quickly find the best improvement method for the 14-bit ADC, Table 8 gives a hint to which method is best for a given application. The selection criteria are:

- Accuracy: The range is used.
- RAM critical: The RAM needed for the coefficients is ≤ 8 bytes.
- Time critical: The calculation time takes ≤ 60 cycles on an average.

This table is taken from the results of device 1. But the table may be usable for other MSP430 devices too. With a more regular ADC characteristic than device 1, the more complex methods will show better results than the simpler ones.

Table 8. Selection for the Improvement Methods

NEEDED ACCURACY	RAM CRITICAL TIME CRITICAL	RAM CRITICAL	TIME CRITICAL	RAM AND TIME NON-CRITICAL
High (±2.5 Steps)	Not possible	Single linear equation/range (linear regression)	Multiple sections (64 sections)	Two linear equations/range (linear regression)
Medium (±3.5 Steps)	Not possible	Single linear equation/range (border fit)	Multiple sections (16 and 32 sections)	All others not named
Low (±7 Steps)	Mean Value/Range Center of Ranges Multiple Sections (8 Sections)			
Very Low (±10 Steps)	Mean value of full range			

5 Summary

The five application reports in this series show many simple-to-realize improvements for the accuracy of the 14-bit analog-to-digital converter of the MSP430. From a non-corrected error range of 17 steps, it was possible to reduce the range to less than ±2.5 steps. Even better, the standard deviation improved from 4.74 steps to 0.65 steps. With a larger effort, the results can be even better—for example if sophisticated statistical methods are applied. Solutions are possible for real-time systems also, e.g., the additive method with its simple and fast algorithm. It was also shown, that relatively simple correction methods can deliver the best results.

6 References

1. *Architecture and Function of the MSP430 14-Bit ADC Application Report*, 1999, Literature #SLAA045
2. *Application Basics for the MSP430 14-Bit ADC Application Report*, 1999, Literature #SLAA046
3. *Additive Improvement of the MSP430 14-Bit ADC Characteristic Application Report*, 1999, Literature #SLAA047
4. *Linear Improvement of the MSP430 14-Bit ADC Characteristic Application Report*, 1999, Literature #SLAA048
5. *MSP430 Metering Application Report*, 1997, Literature #SLAAE10B
6. Data Sheet MSP430C325, MSP430P323, 1998, Literature #SLASE06A
7. *MSP430 Family Architecture Guide and Module Library*, 1996, Literature #SLAUE10B
8. *MSP430 Application Report*, 1998, Literature #SLAAE10C