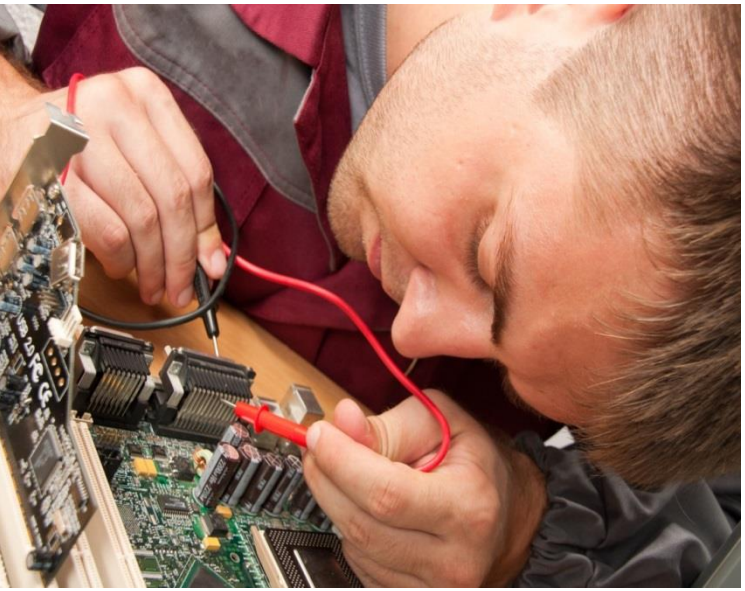
TECHNISCHE HOCHSCHULE DEGGENDORF



Microcontroller Programming (8)

Gerald Kupris, 03.12.2013

Lectures Microcontroller Programming WS2013/14

08.10.2013 Microcontroller, Programming and Debuging Interfaces
15.10.2013 Reading and Writing of Registers
22.10.2013 I/O-Pins, Reading and Writing of Single Bits
29.10.2013 Clock Generation, CPU und Computing Power
05.11.2013 Interrupts
12.11.2013 No lecture !
19.11.2013 Memory
26.11.2013 Timer and PWM, Watchdog Timer
03.12.2013 Analog to Digital Converter
10.12.2013 Additional Explanation of the Freescale Cup Cars
17.12.2013 Serial Interfaces: SPI, IIC and UART
14.01.2014 Current Consumption and Low Power Modi
21.01.2014 Freescale Cup Semester Final Race at 14:00 p.m.

Hands-On Workshops Microcontroller Programming

08.10.2013 Workshop 1: Preparation of the Work Place
15.10.2013 Workshop 2: Loading and Debugging of Programs
22.10.2013 Workshop 3: Using the GPIO Pins
29.10.2013 Workshop 4: Clock Generation and Calculations
05.11.2013 Workshop 5: Interrupts
12.11.2013 No Workshop !
19.11.2013 Workshop 6: Using the Flash Memory
26.11.2013 Workshop 7: Timer and Pulse Width Modulation (PWM)
03.12.2013 Workshop 8: Analog to Digital Conversion
10.12.2013 Project Work on the Freescale Cup Cars
17.12.2013 Project Work on the Freescale Cup Cars
14.01.2014 Project Work on the Freescale Cup Cars
21.01.2014 Freescale Cup Semester Final Race at 14:00 p.m.

Participation on all workshops is required for admittance to the final race!

Workshop Start Time Tuesday: 15:45 p.m.
Workshop Start Time Thursday: 14:45 p.m.

Freescale Cup Rules of Participation

All Teams will receive a pre-assembled Freescale Cup Car.
This car can be / should be modified (HW and SW) as wanted, as long as it complies to the **2014 EMEA Challenge Rules** (published on V: drive).

All teams have to submit a technical report of their vehicle.

The Semester Final Race will take place 21.01.2014 at 14:00 in ITC1.

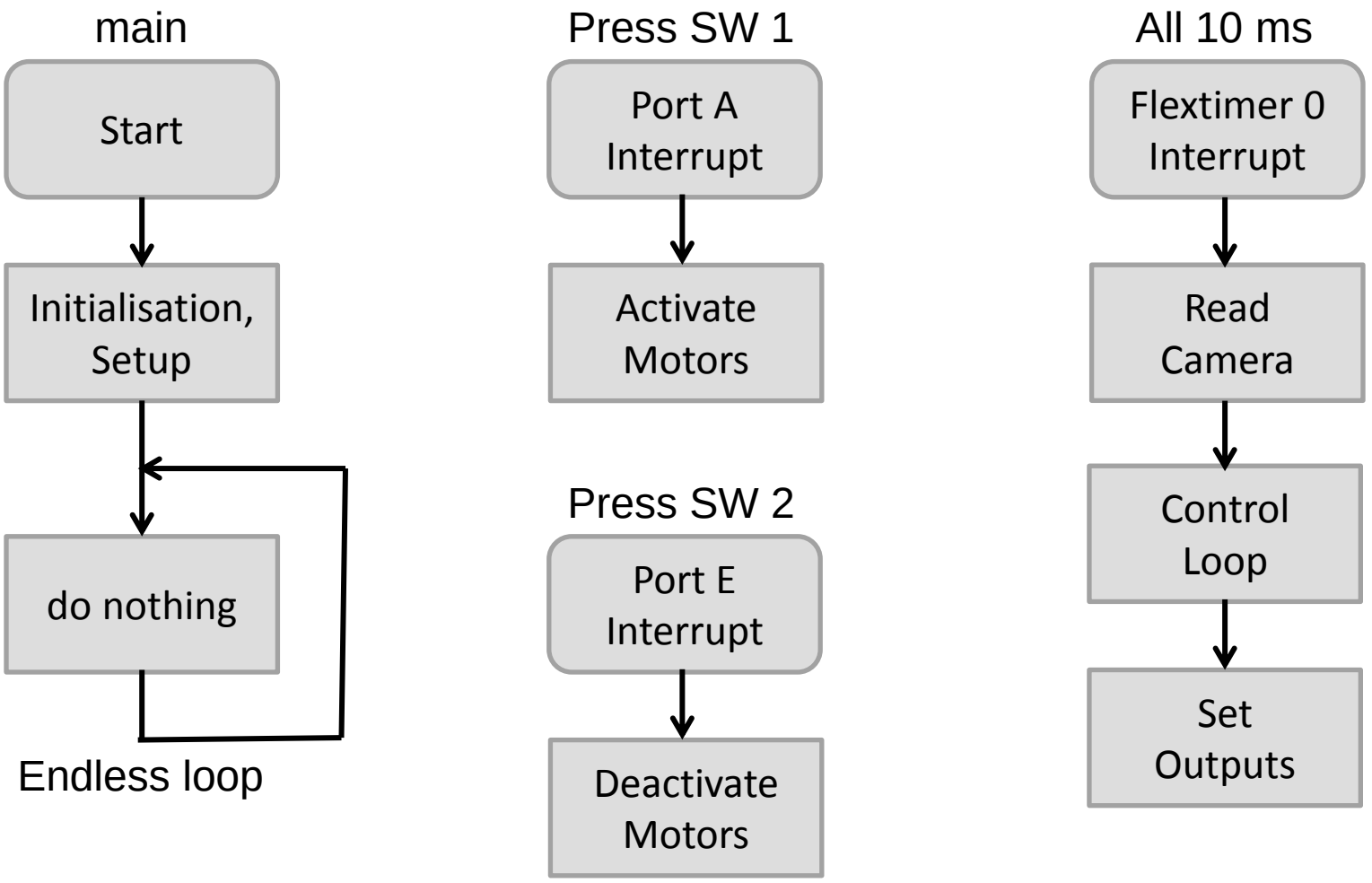
The grade of the course will depend on the results of the semester finals and on the quality of the technical report.

The best teams will participate in the German Qualification Race:
Tuesday, 18 Mar 2014: Munich, Germany at the University of Applied Sciences Munich - attendees of all teams in Germany and UK.

Test drives at the Embedded World 25. - 27. Feb. 2014 in Nuremberg!

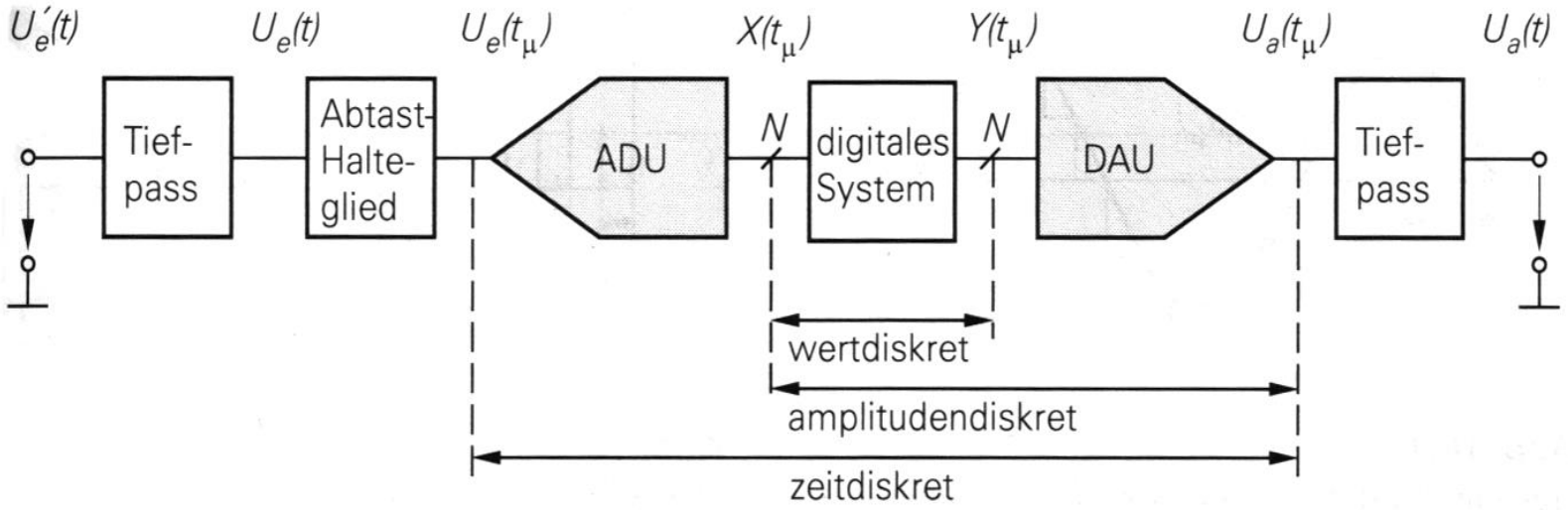
Freescale Cup Basic Software Structure

Example project for K60N512, but can be changed to any other MCU



Digital Signal Processing

In the real-world we have analog signals. The first step is usually to convert the signal from an analog to a digital form, by sampling and then digitizing it using an analog-to-digital converter (ADC), which turns the analog signal into a stream of numbers. However, often, the required output signal is another analog output signal, which requires a digital-to-analog converter (DAC). Even if this process is more complex than analog processing and has a discrete value range, the application of computational power to digital signal processing allows for many advantages over analog processing in many applications, such as error detection and correction in transmission as well as data compression.



Analog to Digital Converter

An analog-to-digital converter (abbreviated ADC, A/D or A to D) is a device that converts a continuous physical quantity (usually voltage) to a digital number that represents the quantity's amplitude.

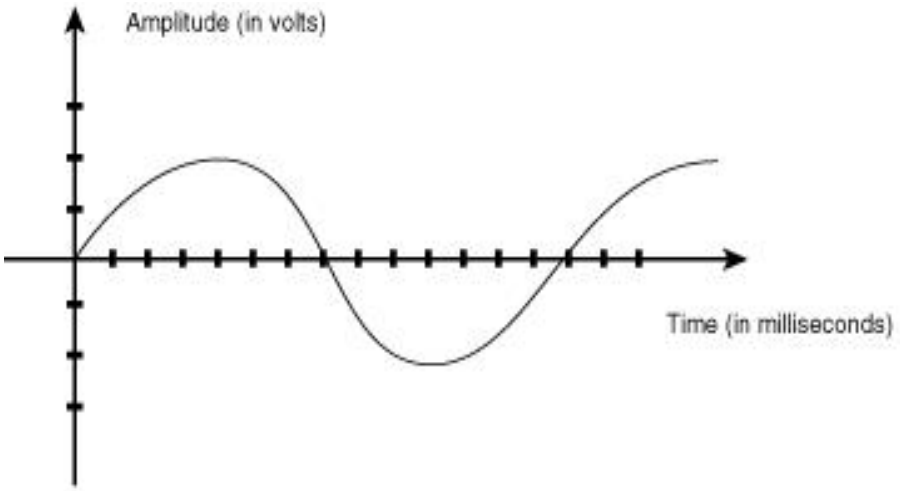
The conversion involves quantization of the input, so it necessarily introduces a small amount of error. Instead of doing a single conversion, an ADC often performs the conversions ("samples" the input) periodically. The result is a sequence of digital values that have converted a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal.

An ADC is defined by its bandwidth (the range of frequencies it can measure) and its signal to noise ratio (how accurately it can measure a signal relative to the noise it introduces). The actual bandwidth of an ADC is characterized primarily by its sampling rate, and to a lesser extent by how it handles errors such as aliasing. The dynamic range of an ADC is influenced by many factors, including the resolution, linearity and accuracy.

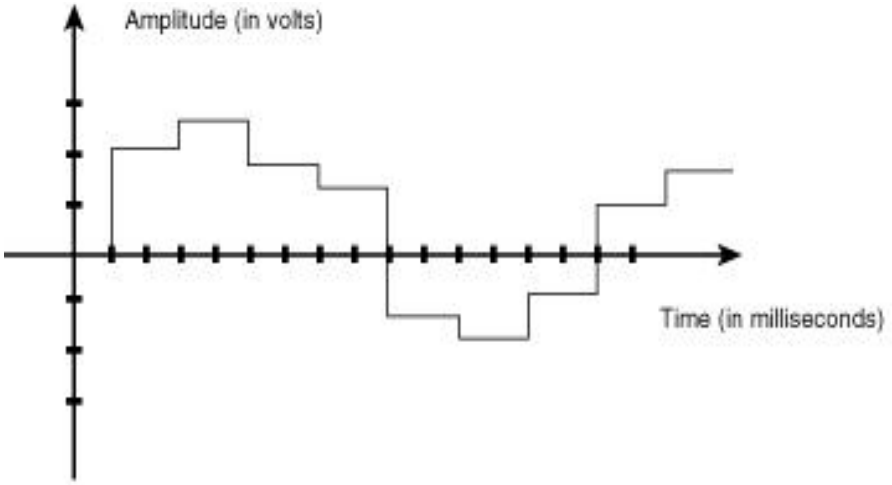
Quantization

Quantization, in mathematics and digital signal processing, is the process of mapping a large set of input values to a smaller set – such as rounding values to some unit of precision. A device or algorithmic function that performs quantization is called a quantizer. The round-off error introduced by quantization is referred to as quantization error.

Analog Signal



Digital Signal

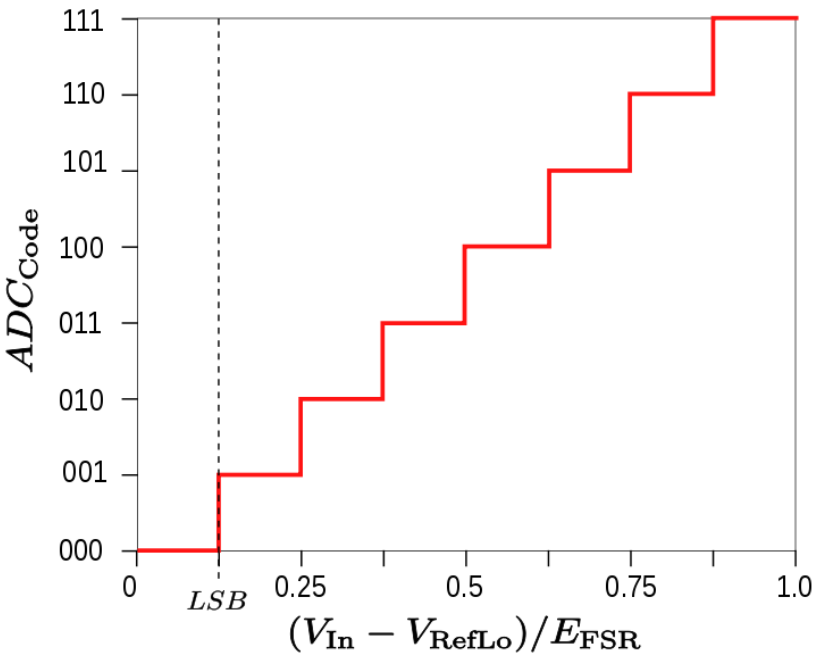


Resolution

The resolution of the converter indicates the number of discrete values it can produce over the range of analog values. The resolution determines the magnitude of the quantization error and therefore determines the maximum possible average signal to noise ratio for an ideal ADC without the use of oversampling. The values are usually stored electronically in binary form, so the resolution is usually expressed in bits.

In consequence, the number of discrete values available, or "levels", is assumed to be a power of two.

For example, an ADC with a resolution of 8 bits can encode an analog input to one in 256 different levels, since $2^8 = 256$. The values can represent the ranges from 0 to 255 (i.e. unsigned integer) or from −128 to 127 (i.e. signed integer), depending on the application.

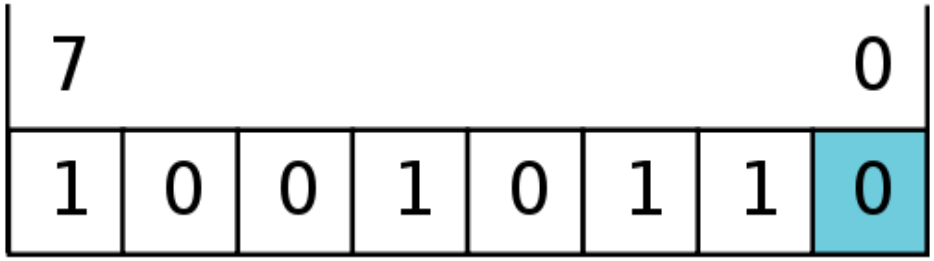


Least Significant Bit

Resolution can also be defined electrically, and expressed in volts. The minimum change in voltage required to guarantee a change in the output code level is called the least significant bit (LSB) voltage. The resolution Q of the ADC is equal to the LSB voltage.

An ADC has several sources of errors. Quantization error and (assuming the ADC is intended to be linear) non-linearity are intrinsic to any analog-to-digital conversion.

These errors are measured in a unit called the least significant bit (LSB). In the above example of an eight-bit ADC, an error of one LSB is 1/256 of the full signal range, or about 0.4%.

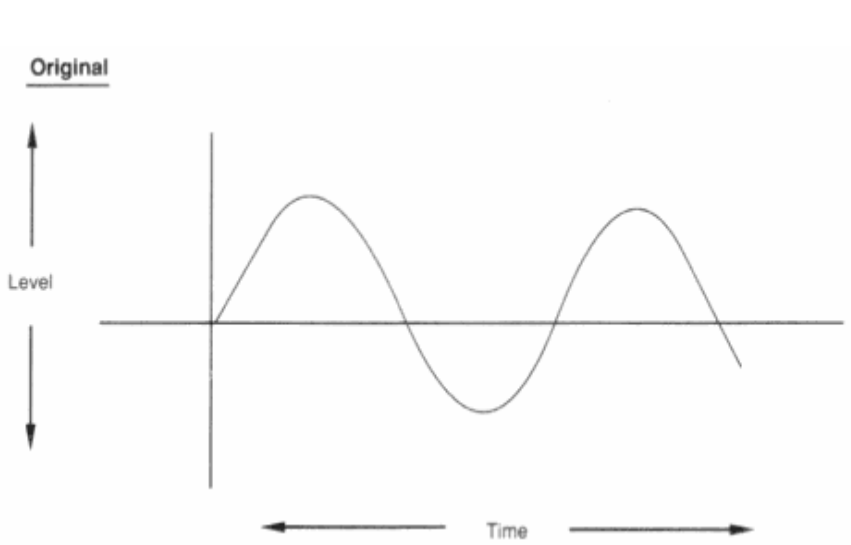


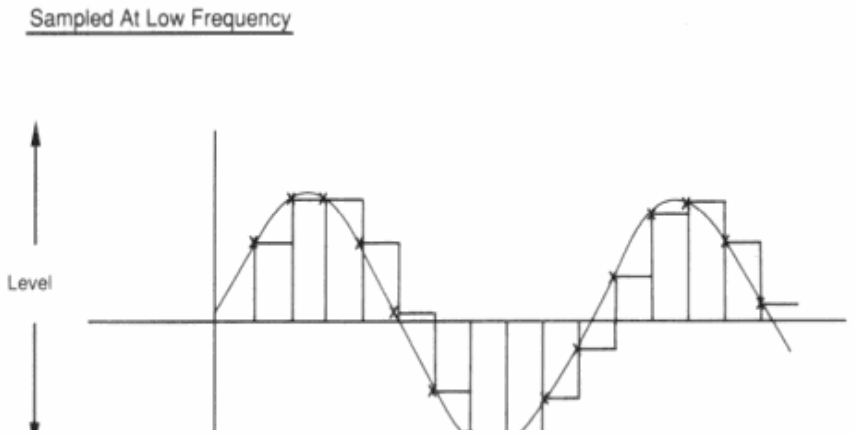
Effective Number of Bits

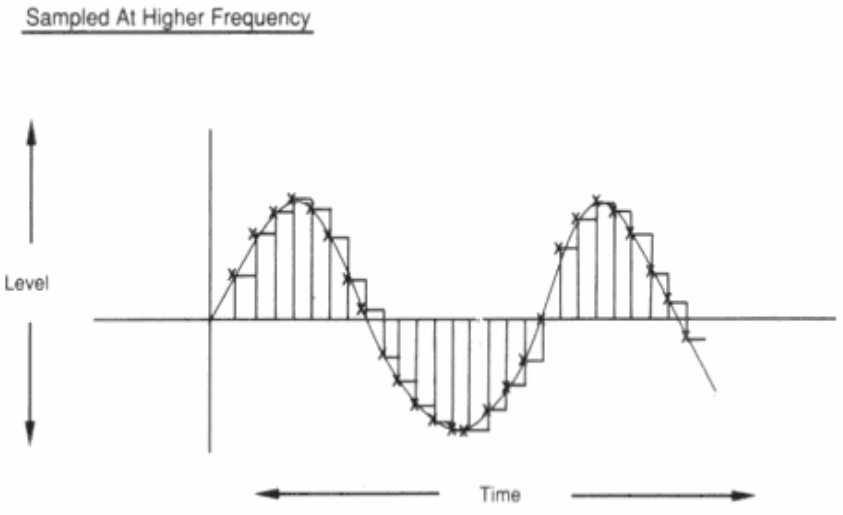
Effective number of bits (ENOB) is a measure of the dynamic performance of an analog-to-digital converter (ADC) and its associated circuitry. The resolution of an ADC is specified by the number of bits used to represent the analog value, in principle giving 2^N signal levels for an N-bit signal.

However, all real ADC circuits introduce noise and distortion. ENOB specifies the resolution of an ideal ADC circuit that would have the same resolution as the circuit under consideration.

Sampling Rate







The sampling rate, sample rate, or sampling frequency defines the number of samples per unit of time (usually seconds) taken from a continuous signal to make a discrete signal.

Example: Sampling Rate for CD: 44,1 kHz

Oversampling

Oversampling is the process of sampling a signal with a sampling frequency significantly higher than twice the bandwidth or highest frequency of the signal being sampled. Oversampling helps avoid aliasing, improves resolution and reduces noise.

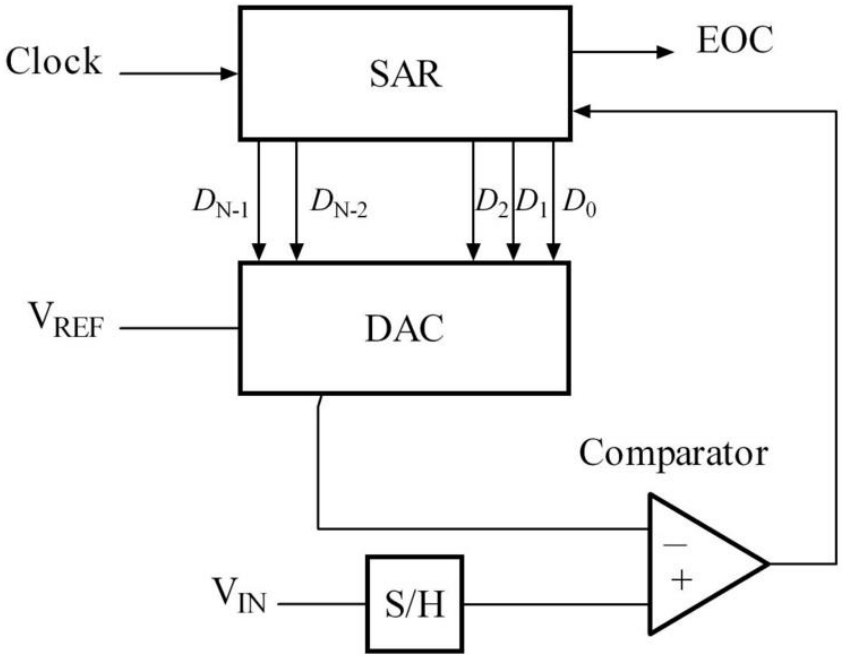
In practice, oversampling is implemented in order to achieve cheaper higher-resolution A/D and D/A conversion. For instance, to implement a 24-bit converter, it is sufficient to use a 20-bit converter that can run at 256 times the target sampling rate. Combining 256 consecutive 20-bit samples can increase the signal-to-noise ratio at the voltage level by a factor of 16 (the square root of the number of samples averaged), adding 4 bits to the resolution and producing a single sample with 24-bit resolution.

Successive Approximation ADC

The successive approximation ADC circuit typically consists of four chief subcircuits:

A sample and hold circuit to acquire the input voltage (Vin).

An analog voltage comparator that compares Vin to the output of the internal DAC and outputs the result of the comparison to the successive approximation register (SAR).



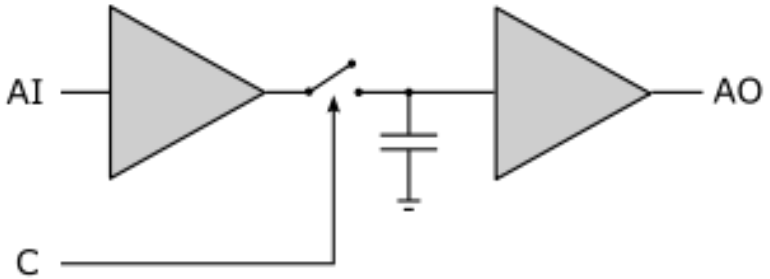
A successive approximation register subcircuit designed to supply an approximate digital code of Vin to the internal DAC.

An internal reference DAC that supplies the comparator with an analog voltage for comparison with Vin.

Sample and Hold

A sample and hold (S/H, also "follow-and-hold") circuit is an analog device that samples (captures, grabs) the voltage of a continuously varying analog signal and holds (locks, freezes) its value at a constant level for a specified minimum period of time. Sample and hold circuits and related peak detectors are the elementary analog memory devices. They are typically used in analog-to-digital converters to eliminate variations in input signal that can corrupt the conversion process.

A typical sample and hold circuit stores electric charge in a capacitor and contains at least one fast FET switch and at least one operational amplifier. To sample the input signal the switch connects the capacitor to the output of a buffer amplifier. The buffer amplifier charges or discharges the capacitor so that the voltage across the capacitor is practically equal, or proportional to, input voltage. In hold mode the switch disconnects the capacitor from the buffer.



Successive Approximation ADC Function

The sample and hold circuit is used to acquire the input voltage (Vin).

The successive approximation register is initialized so that the most significant bit (MSB) is equal to a digital 1. This code is fed into the DAC, which then supplies the analog equivalent of this digital code (Vref/2) into the comparator circuit for comparison with the sampled input voltage.

If this analog voltage exceeds Vin the comparator causes the SAR to reset this bit; otherwise, the bit is left a 1. Then the next bit is set to 1 and the same test is done, continuing this binary search until every bit in the SAR has been tested. The resulting code is the digital approximation of the sampled input voltage and is finally output by the DAC at the end of the conversion (EOC).

Important consequence: an A/D conversion takes some clock cycles (time)!

Other types of A/D Converters

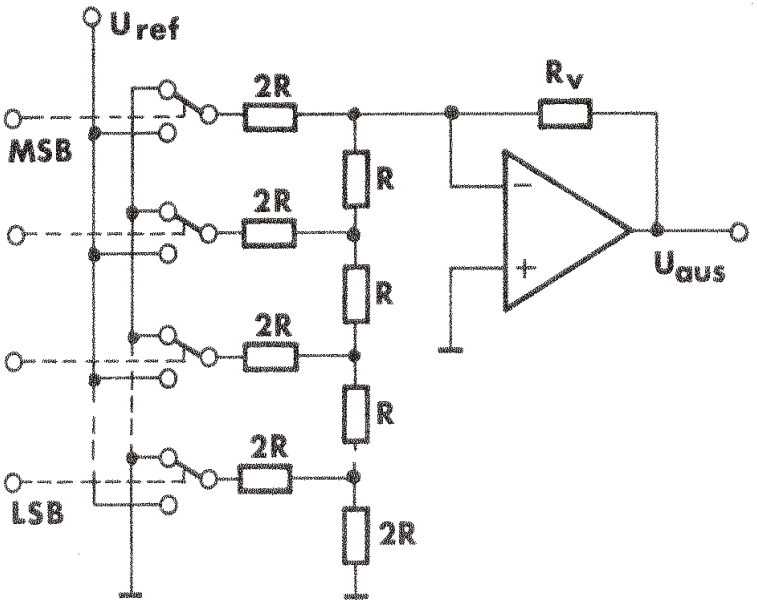
A **direct-conversion ADC** or **flash ADC** has a bank of comparators sampling the input signal in parallel, each firing for their decoded voltage range. The comparator bank feeds a logic circuit that generates a code for each voltage range. Direct conversion is very fast, capable of gigahertz sampling rates, but usually has only 8 bits of resolution or fewer.

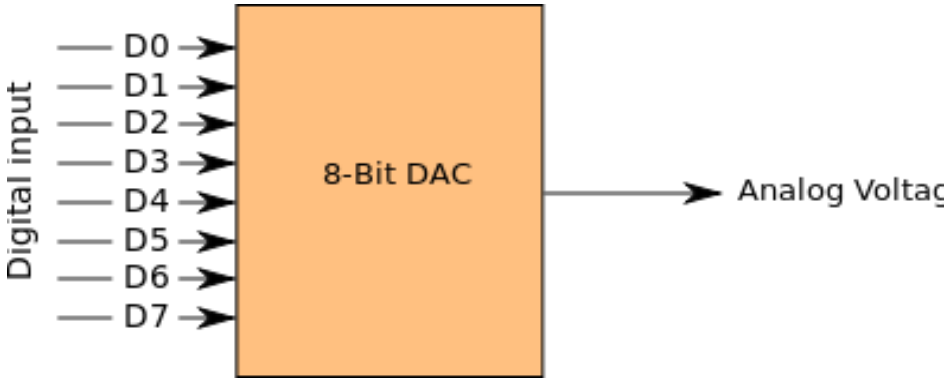
An **integrating ADC** (also **dual-slope** or multi-slope ADC) applies the unknown input voltage to the input of an integrator and allows the voltage to ramp for a fixed time period (the run-up period). Then a known reference voltage of opposite polarity is applied to the integrator and is allowed to ramp until the integrator output returns to zero (the run-down period). The input voltage is computed as a function of the reference voltage, the constant run-up time period, and the measured run-down time period. The run-down time measurement is usually made in units of the converter's clock, so longer integration times allow for higher resolutions. Likewise, the speed of the converter can be improved by sacrificing resolution.

Digital to Analog Converter

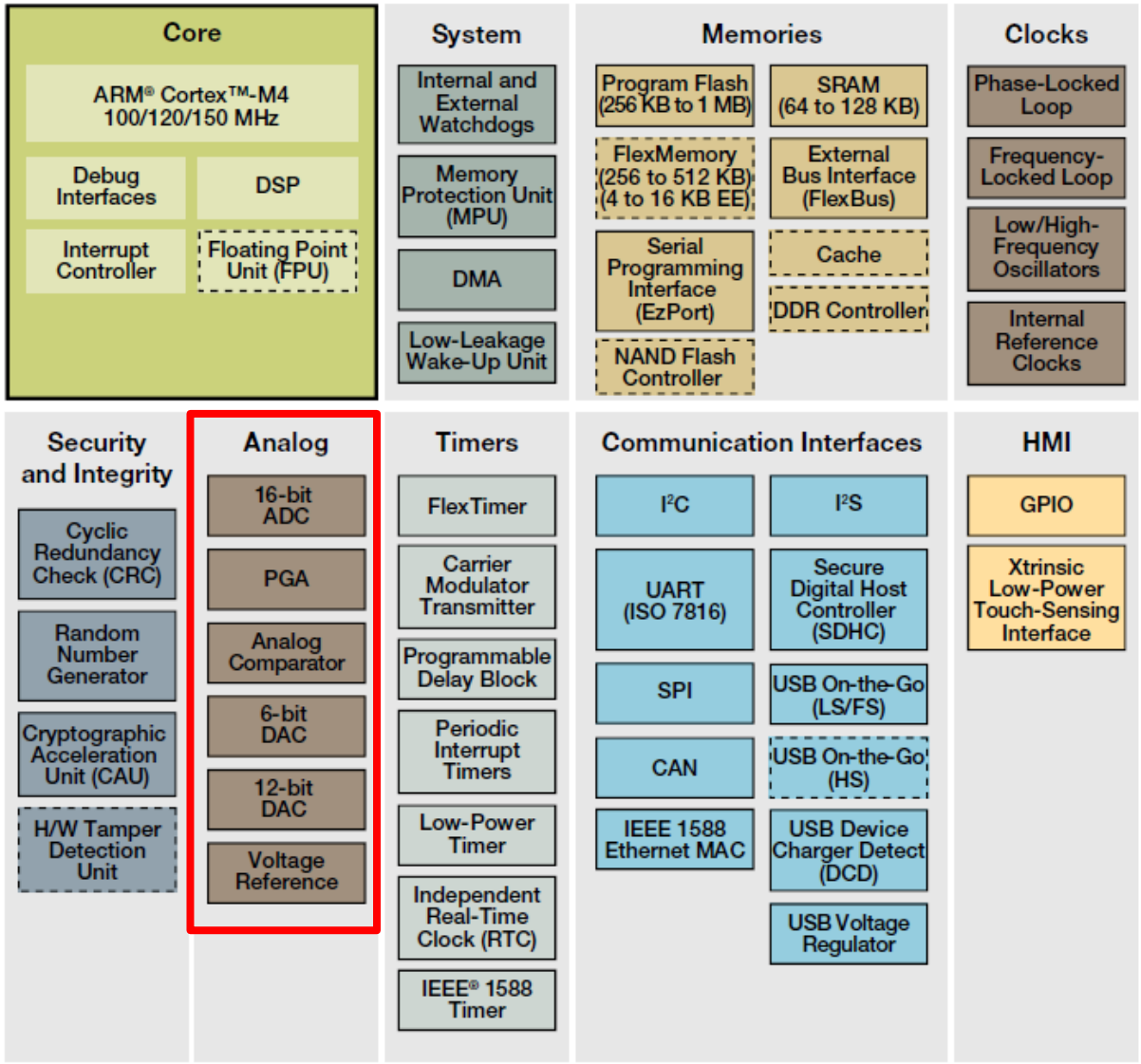
A digital-to-analog converter (DAC or D-to-A) is a function that converts digital data (usually binary) into an analog signal (current, voltage, or electric charge).

A DAC converts an abstract finite-precision number (usually a fixed-point binary number) into a physical quantity (e.g., a voltage or a pressure). In particular, DACs are often used to convert finite-precision time series data to a continually varying physical signal.





Analog Modules on the Kinetis K60



Analog-to-Digital Converter (ADC)

The Kinetis K60 device contains two ADCs: ADC0 and ADC1.

Features

- 16-bit Successive Approximation (SAR)
- 1.2V minimum reference
- Differential or Single Ended
- Averaging by 1, 4, 8, 16, or 32
- Automatic Compare Function
- Triggering synchronization w/ DAC
- Configurable sample time, speed/power
- Up to 20 input channels per converter
- 1µs continuous conversions (12-bit mode)
- PGA front-end (ch2) with x64 gain

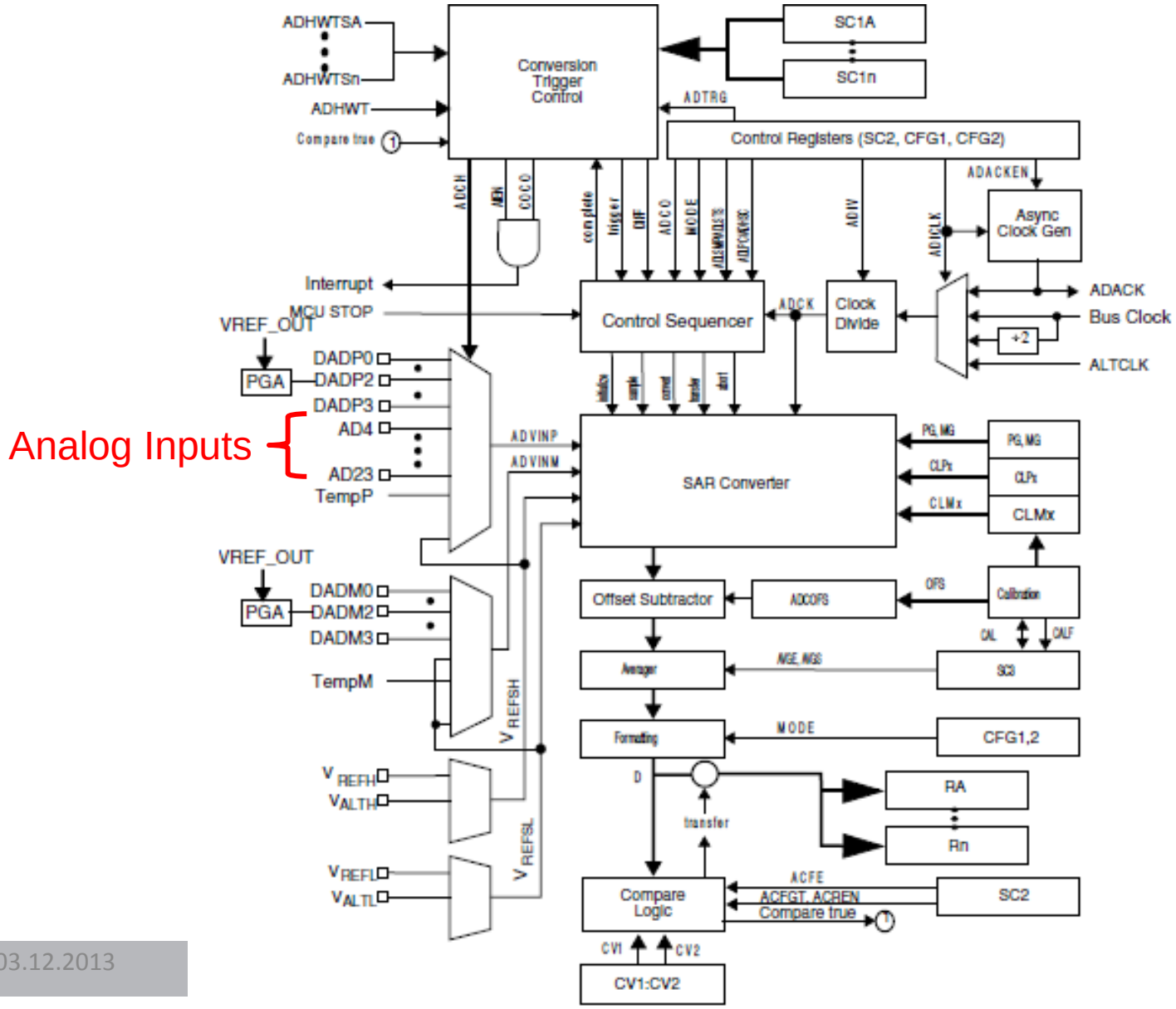
Performance

- Effective Number of Bits (ENOB)
  - AVG=32:(min)14.3b(typ)**14.5b***(4MHz, >2.7V)*
  - AVG=16:(min)13.8b(typ)14.0b*(4MHz, >2.7V)*
  - AVG=32:(min)12.8b(typ)14.2b*(all conditions)*
  - AVG=16:(min)12.7b(typ)13.8b*(all conditions)*
- Conversion Time
  - 16b, s.e., 8MHz, avg=1:3.12µs
  - 16b, s.e., 4MHz, avg=16:100µs
  - 16b, diff., 4MHz, avg=32:272µs

Key Points

- Successive Approximation is **much faster than Sigma Delta** → Allows monitoring rapidly changing events, channel switching/interleaving, very low power conversions
- With averaging, accuracy as good as SD → Allows accurate measurement of slower or settled signals even while interleaved or switching channels
- **Differential input allows much better noise rejection** and accuracy than single ended with reduced external components
- **Result is as accurate as SD but much faster meaning lower measurement time**

ADC Block Diagram



ADC Signal Descriptions

Signal	Description	I/O
DADP[3:0]	Differential analog channel inputs	I
DADM[3:0]	Differential analog channel inputs	I
AD[23:4]	Single-ended analog channel inputs	I
V_{REFSH}	Voltage reference select high	I
V_{REFSL}	Voltage reference select low	I
V_{DDA}	Analog power supply	I
V_{SSA}	Analog ground	I

Analog channel inputs (ADx)
The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the ADCH channel select bits when the DIFF bit in the SC1n register is low.

Differential analog channel inputs (DADx)
The ADC module supports up to 4 differential analog channel inputs. Each differential analog input is a pair of external pins (DADPx and DADMx) referenced to each other to provide the most accurate analog to digital readings.

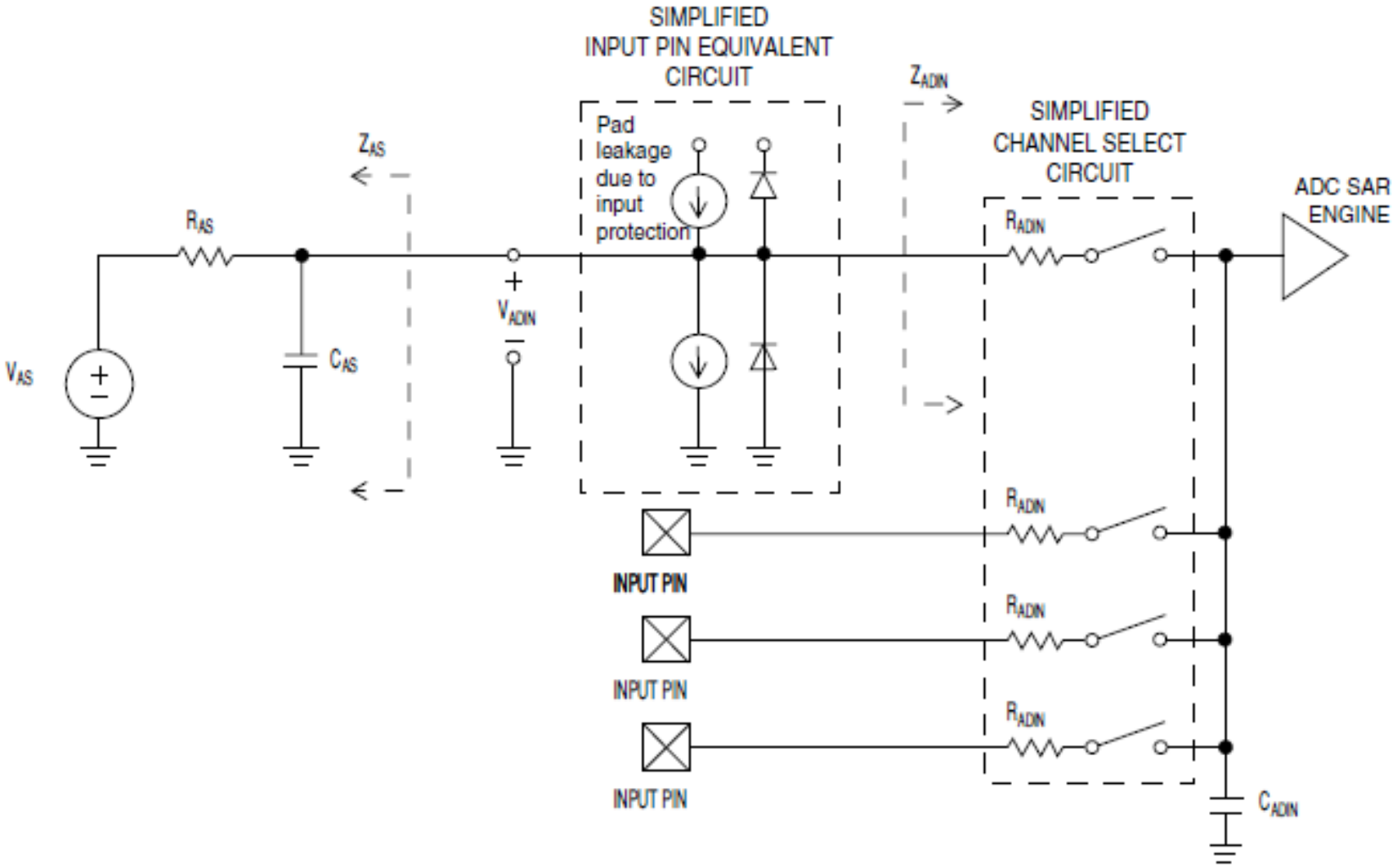
Analog Power, Ground, Reference

Analog Power (VDDA)
The ADC analog portion uses VDDA as its power connection. In some packages, VDDA is connected internally to VDD. If externally available, connect the VDDA pin to the same voltage potential as VDD. External filtering may be necessary to ensure clean VDDA for good results.

Analog Ground (VSSA)
The ADC analog portion uses VSSA as its ground connection. In some packages, VSSA is connected internally to VSS. If externally available, connect the VSSA pin to the same voltage potential as VSS.

Voltage Reference Select (VREFSH and VREFSL)
VREFSH and VREFSL are the high and low reference voltages for the converter. The ADC can be configured to accept one of two voltage reference pairs for VREFSH and VREFSL. Each pair contains a positive reference that must be between the minimum Ref Voltage High and VDDA, and a ground reference that must be at the same potential as VSSA.

ADC Input Impedance Equivalency Diagram



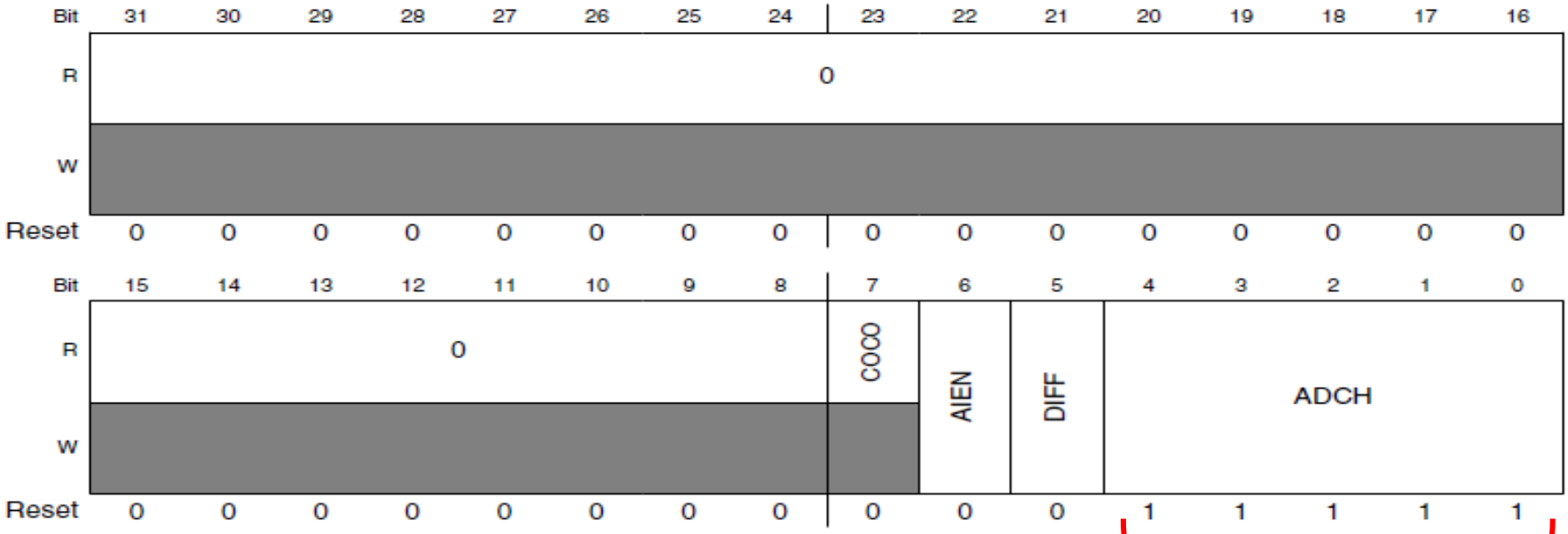
ADC Memory Map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4003_B000	ADC status and control registers 1 (ADC0_SC1A)	32	R/W	0000_001Fh
4003_B004	ADC status and control registers 1 (ADC0_SC1B)	32	R/W	0000_001Fh
4003_B008	ADC configuration register 1 (ADC0_CFG1)	32	R/W	0000_0000h

... alltogether 56 Registers ...

400B_B068	ADC minus-side general calibration value register (ADC1_CLM1)	32	R/W	0000_0040h
400B_B06C	ADC minus-side general calibration value register (ADC1_CLM0)	32	R/W	0000_0020h

ADCx Status and Control Register



Conversion complete flag

ADC interrupt enable

Differential mode enable

Input channel select

ADCx Status and Control Register

Field	Description
6 AIEN	Interrupt enable AIEN enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt disabled. 1 Conversion complete interrupt enabled.
5 DIFF	Differential mode enable DIFF configures the ADC to operate in differential mode. When enabled, this mode automatically selects from the differential channels, and changes the conversion algorithm and the number of cycles to complete a conversion. 0 Single-ended conversions and input channels are selected. 1 Differential conversions and input channels are selected.
4–0 ADCH	Input channel select The ADCH bits form a 5-bit field that selects one of the input channels. The input channel decode depends on the value of the DIFF bit. DAD0-DAD3 are associated with the input pin pairs DADPx and DADMx. The successive approximation converter subsystem is turned off when the channel select bits are all set (ADCH = 11111). This feature allows for explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional single conversion from being performed. It is not necessary to set the channel select bits to all ones to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes.

ADC Channel Assignment

ADC Channel (SC1n[ADCH])	Channel	Input signal (SC1n[DIFF]= 1)	Input signal (SC1n[DIFF]= 0)
00000	DAD0	ADC0_DP0 and ADC0_DM0[1]	ADC0_DP0[2]
00001	DAD1	ADC0_DP1 and ADC0_DM1	ADC0_DP1
00010	DAD2	PGA0_DP and PGA0_DM	PGA0_DP
00011	DAD3	ADC0_DP3 and ADC0_DM3[3]	ADC0_DP3[4]
00100[5]	AD4a	Reserved	Reserved
00101[5]	AD5a	Reserved	Reserved
00110[5]	AD6a	Reserved	Reserved
00111[5]	AD7a	Reserved	Reserved
00100[5]	AD4b	Reserved	ADC0_SE4b
00101[5]	AD5b	Reserved	ADC0_SE5b
00110[5]	AD6b	Reserved	ADC0_SE6b
00111[5]	AD7b	Reserved	ADC0_SE7b
11001	AD25	Reserved	Reserved
11010	AD26	Temperature Sensor (Diff)	Temperature Sensor (S.E)
11011	AD27	Bandgap (Diff)[9]	Bandgap (S.E)[9]
11100	AD28	Reserved	Reserved
11101	AD29	-VREFH (Diff)	VREFH (S.E)
11110	AD30	Reserved	VREFL
11111	AD31	Module Disabled	Module Disabled

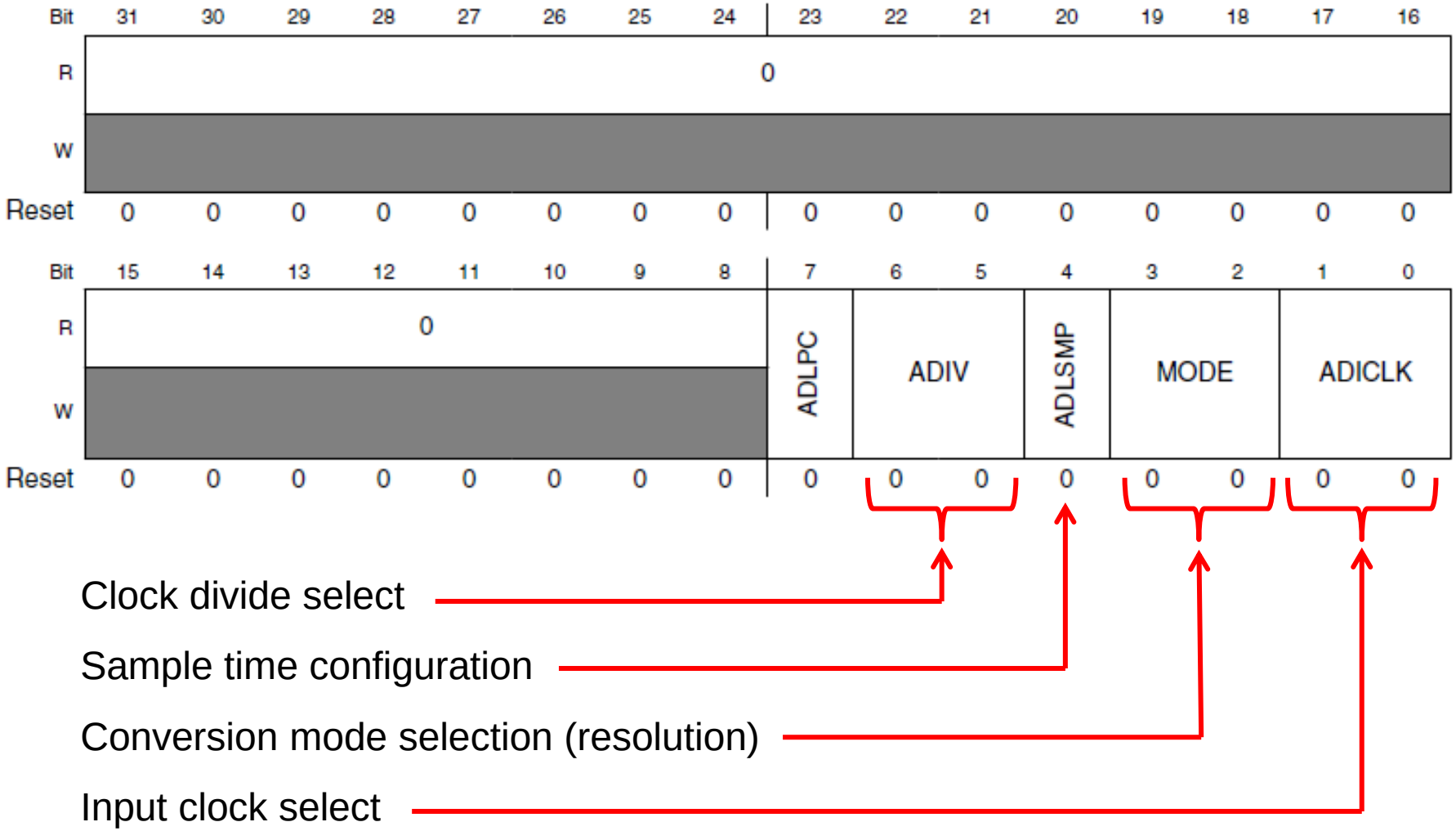
Define A/D Channel!

ADC1
SC1n[DIFF]= 0
Single ended mode!

SC1n[ADCH]= 10100
ADC1, Channel 20

```
ADC1_SC1A = 20;                                  // ADC1 Channel 20
printf("Port A Interrupt A/D Value: ");  // print
ad_value = ADC1_RA;                              // read A/D value
printf("%i\n", ad_value);                        // print A/D value
```

ADCx Configuration Register



ADCx Configuration Register

4 ADLSMP	Sample time configuration ADLSMP selects between different sample times based on the conversion mode selected. This bit adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption if continuous conversions are enabled and high conversion rates are not required. When ADLSMP=1, the long sample time select bits, (ADLSTS[1:0]), can select the extent of the long sample time. 0 Short sample time. 1 Long sample time.
3–2 MODE	Conversion mode selection MODE bits are used to select the ADC resolution mode. 00 When DIFF=0: It is single-ended 8-bit conversion; when DIFF=1, it is differential 9-bit conversion with 2's complement output. 01 When DIFF=0: It is single-ended 12-bit conversion; when DIFF=1, it is differential 13-bit conversion with 2's complement output. 10 When DIFF=0: It is single-ended 10-bit conversion; when DIFF=1, it is differential 11-bit conversion with 2's complement output. 11 When DIFF=0: It is single-ended 16-bit conversion; when DIFF=1, it is differential 16-bit conversion with 2's complement output.
1–0 ADICLK	Input clock select ADICLK bits select the input clock source to generate the internal clock, ADCK. Note that when the

ADC Application Information

The external analog inputs are typically shared with digital I/O pins on MCU devices. Empirical data shows that capacitors on the analog inputs improve performance in The presence of noise or when the source impedance is high.

Use of 0.01 µF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used they must be placed as near as possible to the package pins and be referenced to VSSA.

For proper conversion, **the input voltage must fall between VREFH and VREFL.** If the input is equal to or exceeds VREFH, the converter circuit converts the signal to 0xFFFF (full scale 16-bit representation), 0xFFF (full scale 12-bit representation), 0x3FF (full scale 10-bit representation) or 0xFF (full scale 8-bit representation).

If the input is equal to or less than VREFL, the converter circuit converts it to 0x000. Input voltages between VREFH and VREFL are straight-line linear conversions. There is a brief current associated with VREFL when the sampling capacitor is charging. For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins should not be transitioning during conversions.

ADC Data Result Register

Addresses: ADC0_RA is 4003_B000h base + 10h offset = 4003_B010h
ADC0_RB is 4003_B000h base + 14h offset = 4003_B014h
ADC1_RA is 400B_B000h base + 10h offset = 400B_B010h
ADC1_RB is 400B_B000h base + 14h offset = 400B_B014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																D															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ADCx_Rn field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 D	Data result

Data Result Register Description

Conversion mode	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Format
16-bit differential	S	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Signed 2's complement
16-bit single-ended	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
13-bit differential	S	S	S	S	D	D	D	D	D	D	D	D	D	D	D	D	Sign extended 2's complement
12-bit single-ended	0	0	0	0	D	D	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
11-bit differential	S	S	S	S	S	S	D	D	D	D	D	D	D	D	D	D	Sign extended 2's complement
10-bit single-ended	0	0	0	0	0	0	D	D	D	D	D	D	D	D	D	D	Unsigned right justified
9-bit differential	S	S	S	S	S	S	S	S	D	D	D	D	D	D	D	D	Sign extended 2's complement
8-bit single-ended	0	0	0	0	0	0	0	0	D	D	D	D	D	D	D	D	Unsigned right justified

16-bit ADC Operating Conditions

Symbol	Description	Conditions	Min.	Typ.[1]	Max.	Unit
V_{DDA}	Supply voltage	Absolute	1.71	—	3.6	V
V_{REFH}	ADC reference voltage high		1.13	V_{DDA}	V_{DDA}	V
V_{REFL}	Reference voltage low		V_{SSA}	V_{SSA}	V_{SSA}	V
V_{ADIN}	Input voltage		V_{REFL}	—	V_{REFH}	V
C_{ADIN}	Input capacitance	• 16 bit modes • 8/10/12 bit modes	— —	8 4	10 5	pF
R_{ADIN}	Input resistance		—	2	5	kΩ
f_{ADCK}	ADC conversion clock frequency	≤ 13 bit modes	1.0	—	18.0	MHz
f_{ADCK}	ADC conversion clock frequency	16 bit modes	2.0	—	12.0	MHz

Base Conversion Time (BCT)

Mode	Base conversion time (BCT)
8b s.e.	17 ADCK cycles
9b diff	27 ADCK cycles
10b s.e.	20 ADCK cycles
11b diff	30 ADCK cycles
12b s.e.	20 ADCK cycles
13b diff	30 ADCK cycles
16b s.e.	25 ADCK cycles
16b diff	34 ADCK cycles

f_{ADCK}	ADC conversion clock frequency	≤ 13 bit modes	1.0	—	18.0	MHz
f_{ADCK}	ADC conversion clock frequency	16 bit modes	2.0	—	12.0	MHz

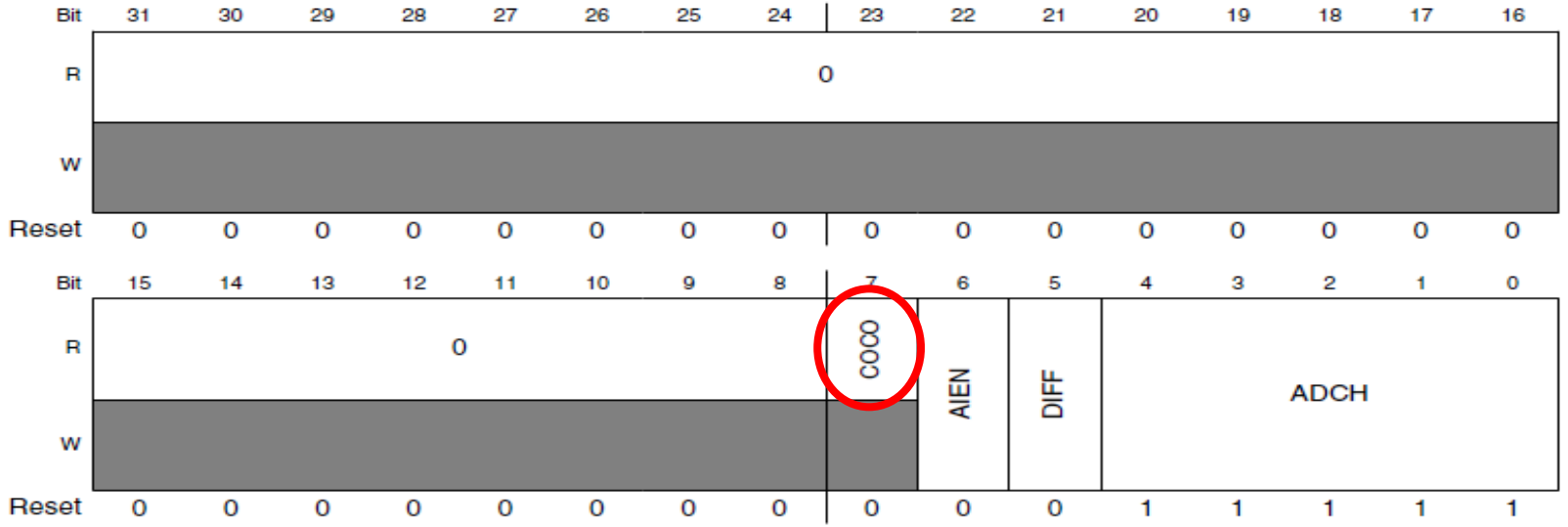
Consider Conversion Time!

```
ADC1_SC1A = 20;                                  // ADC1 Channel 20
printf("Port A Interrupt A/D Value: ");  // print
ad_value = ADC1_RA;                              // read A/D value
printf("%i\n", ad_value);                        // print A/D value
```

Better (Freescale Cup Car):

```
// Read ADC Channel
unsigned char ReadADCChannel(unsigned char Channel)
 {
   ADC1_SC1A  =  Channel;                              // set ADC chann
   while((ADC1_SC1A & ADC_SC1_COCO_MASK) == 0); // wait until AD
   return     ADC1_RA;                                 // return value
 }
```

COCO Flag in the ADC Status and Control Register



Field	Description
31–8 Reserved	This read-only field is reserved and always has the value zero.
7 COCO	Conversion complete flag The COCO flag is a read-only bit that is set each time a conversion is completed when the compare function is disabled (ACFE=0) and the hardware average function is disabled (AVGE=0). When the compare function is enabled (ACFE=1), the COCO flag is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled (AVGE=1), the COCO flag is set upon completion of the selected number of conversions (determined by the AVGS bits). The COCO flag in SC1A is also set at the completion of a Calibration sequence. The COCO bit is cleared when the respective SC1n register is written or when the respective Rn register is read. 0 Conversion not completed. 1 Conversion completed.

16-bit ADC Operating Conditions

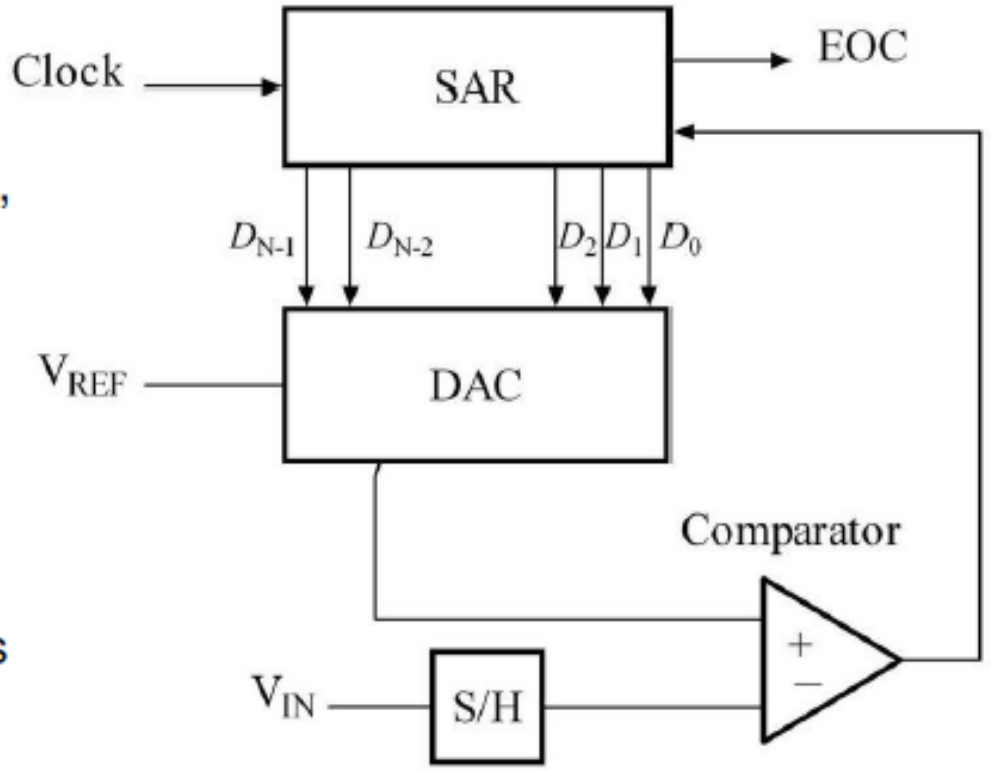
Symbol	Description	Conditions	Min.	Typ.[1]	Max.	Unit
C_{rate}	ADC conversion rate	≤ 13 bit modes No ADC hardware averaging Continuous conversions enabled, subsequent conversion time	20.000	—	818.330	Ksps
C_{rate}	ADC conversion rate	16 bit modes No ADC hardware averaging Continuous conversions enabled, subsequent conversion time	37.037	—	461.467	Ksps

16-bit ADC Operating Errors

Symbol	Description	Conditions[1]	Min.	Typ.[2]	Max.	Unit
TUE	Total unadjusted error	• 12 bit modes	—	±4	±6.8	LSB[4]
		• <12 bit modes	—	±1.4	±2.1	
DNL	Differential non-linearity	• 12 bit modes	—	±0.7	-1.1 to +1.9	LSB[4]
		• <12 bit modes	—	±0.2	-0.3 to 0.5	
INL	Integral non-linearity	• 12 bit modes	—	±1.0	-2.7 to +1.9	LSB[4]
		• <12 bit modes	—	±0.5	-0.7 to +0.5	
E_{FS}	Full-scale error	• 12 bit modes	—	-4	-5.4	LSB[4]
		• <12 bit modes	—	-1.4	-1.8	
E_Q	Quantization error	• 16 bit modes	—	-1 to 0	—	LSB[4]
		• ≤13 bit modes	—	—	±0.5	

ADC16 Calibration: SAR ADC Block Diagram

- ADC16 is a Successive Approximation Register (SAR) architecture.
- Vin is sampled and compared to output voltages of a precision DAC, controlled by the SAR state machine logic.
- The ADC digital result is the DAC setting that most closely approximates the Vin level.
- Generally, an SAR ADC is only as good as the DAC within it. It is the DAC we are calibrating to achieve highest ADC performance!



Automatic Compare Function

The compare function can be configured to check if the result is less than or greater-thanor-equal-to a single compare value, or if the result falls within or outside a range determined by two compare values. The compare mode is determined by ACFGT, ACREN, and the values in the compare value registers (CV1 and CV2).

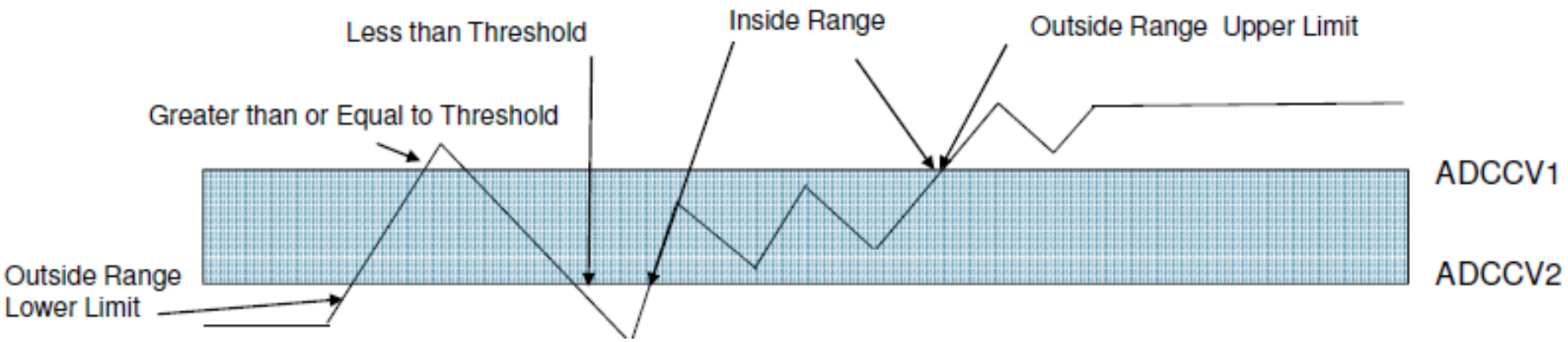
ACFGT	ACREN	ADCCV1 relative to ADCCV2	Function	Compare mode description
0	0	—	Less than threshold	Compare true if the result is less than the CV1 registers.
1	0	—	Greater than or equal to threshold	Compare true if the result is greater than or equal to CV1 registers.
0	1	Less than or equal	Outside range, not inclusive	Compare true if the result is less than CV1 **Or** the result is greater than CV2.
0	1	Greater than	Inside range, not inclusive	Compare true if the result is less than CV1 **And** the result is greater than CV2.
1	1	Less than or equal	Inside range, inclusive	Compare true if the result is greater than or equal to CV1 **And** the result is less than or equal to CV2.
1	1	Greater than	Outside range, inclusive	Compare true if the result is greater than or equal to CV1 **Or** the result is less than or equal to CV2.

Compare Modes

Inside and Outside range capabilities allow applications to do comparator functions without the need for added hardware.

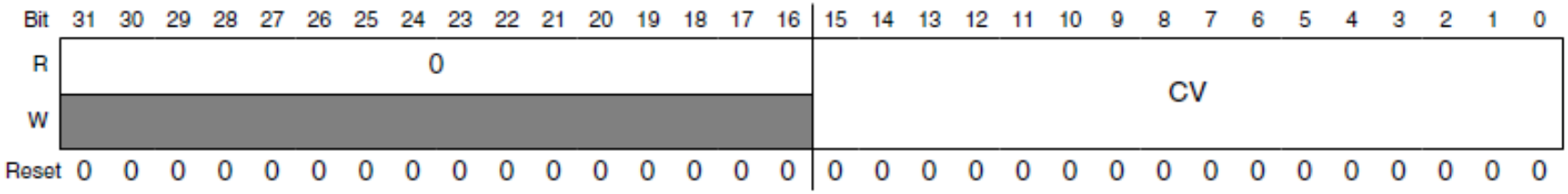
Continuous conversions can be used so that the ADC will not interrupt until the compare condition is met.

Along with the IRTC, the compare functionality can be very useful in stop mode operations.



Compare value registers (ADC*x*_CV*n*)

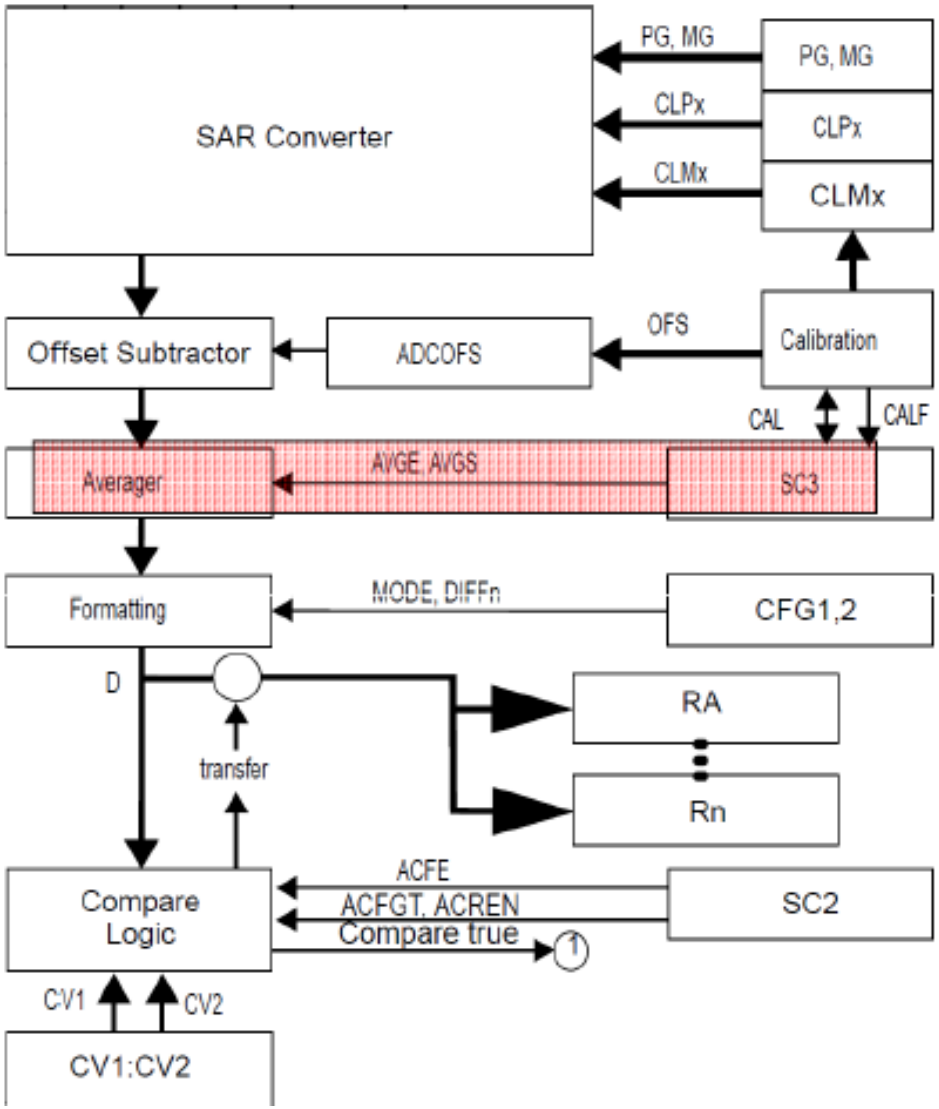
The compare value registers (CV1 and CV2) contain a compare value used to com-Pare with the conversion result when the compare function is enabled (ACFE=1). This register is formatted the same for both bit position definition and value format (unsigned or signextended 2's complement) as the data result registers (Rn) in the different modes of operation. Therefore, the compare function only uses the compare value register bits that are related to the ADC mode of operation.



ADC*x*_CV*n* field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value zero.
15–0 CV	Compare value

Hardware Average

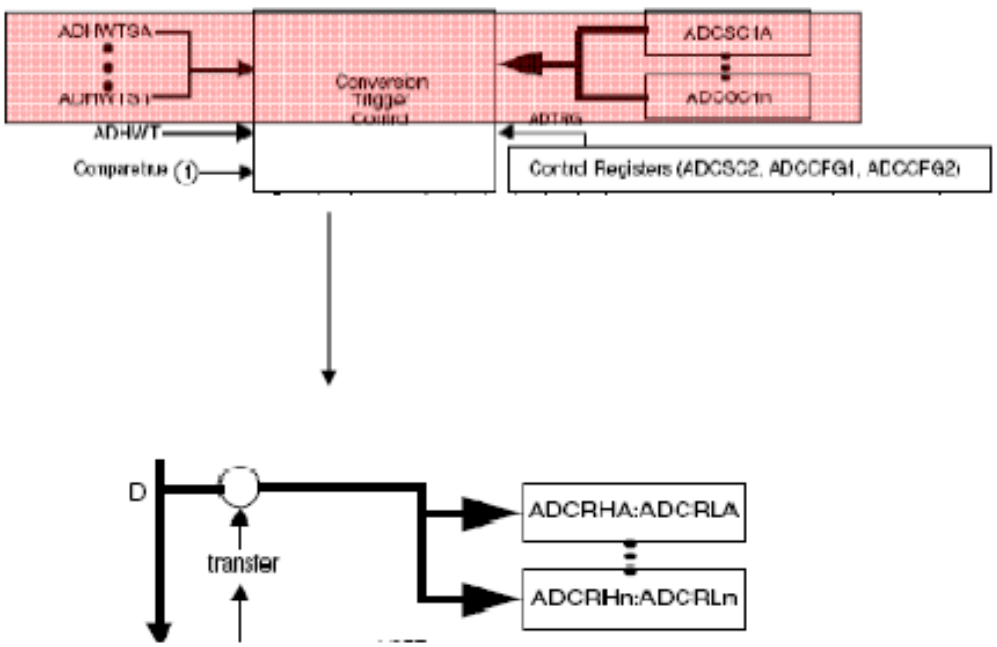


Depending on the AVGS bits 4, 8, 16 or 32 conversions can be averaged using the ADC hardware

Conversions occur back to back and an interrupt is generated at the end of the selected average mode reducing SPS and load core

Averaging can be used to reduce the impact of random noise in the system and is strongly encouraged to achieve the best 16-bit conversion results

Multiple Channel Select and Result Registers



Multiple ADCSC1n registers are used to select channels and conversion modes for the ADC.

Each ADCSC1n register contains its own interrupt enable and conversion complete flag to allow flexibility in the interrupt handling.

Using the Programmable Delay block hardware triggers can be sent to the ADC to initiate conversions at pre-set time intervals for detailed control of ADC conversion timing

Results for each ADCSC1 are stored in individual result registers

Temperature Sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs. The following equation provides an approximate transfer function of the temperature sensor.

$$\text{Temp} = 25 - \left(\left(V_{TEMP} - V_{TEMP25} \right) \div m\right)$$

where:

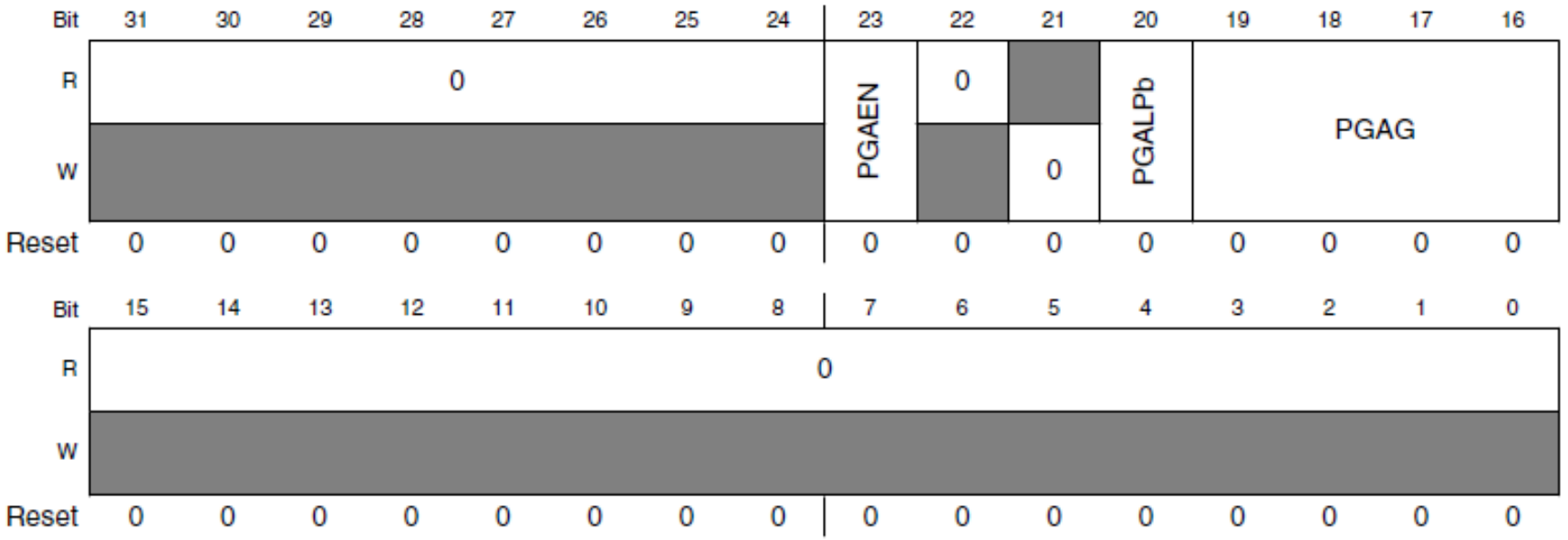
- $\mathbf{V_{TEMP}}$ is the voltage of the temperature sensor channel at the ambient temperat.
- $\mathbf{V_{TEMP25}}$ is the voltage of the temperature sensor channel at 25 °C.
- **m** is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the $\mathbf{V_{TEMP25}}$ and **m** values from the ADC electricals table.

Programmable Gain Amplifier (PGA)

The Programmable Gain Amplifier (PGA) is designed to increase the dynamic range by amplifying low-amplitude signals before they are fed to the 16-bit SAR ADC. The gain of this amplifier is ranged between 1 to 64 in (2^N) steps (1,2,4,8,16,32,64).

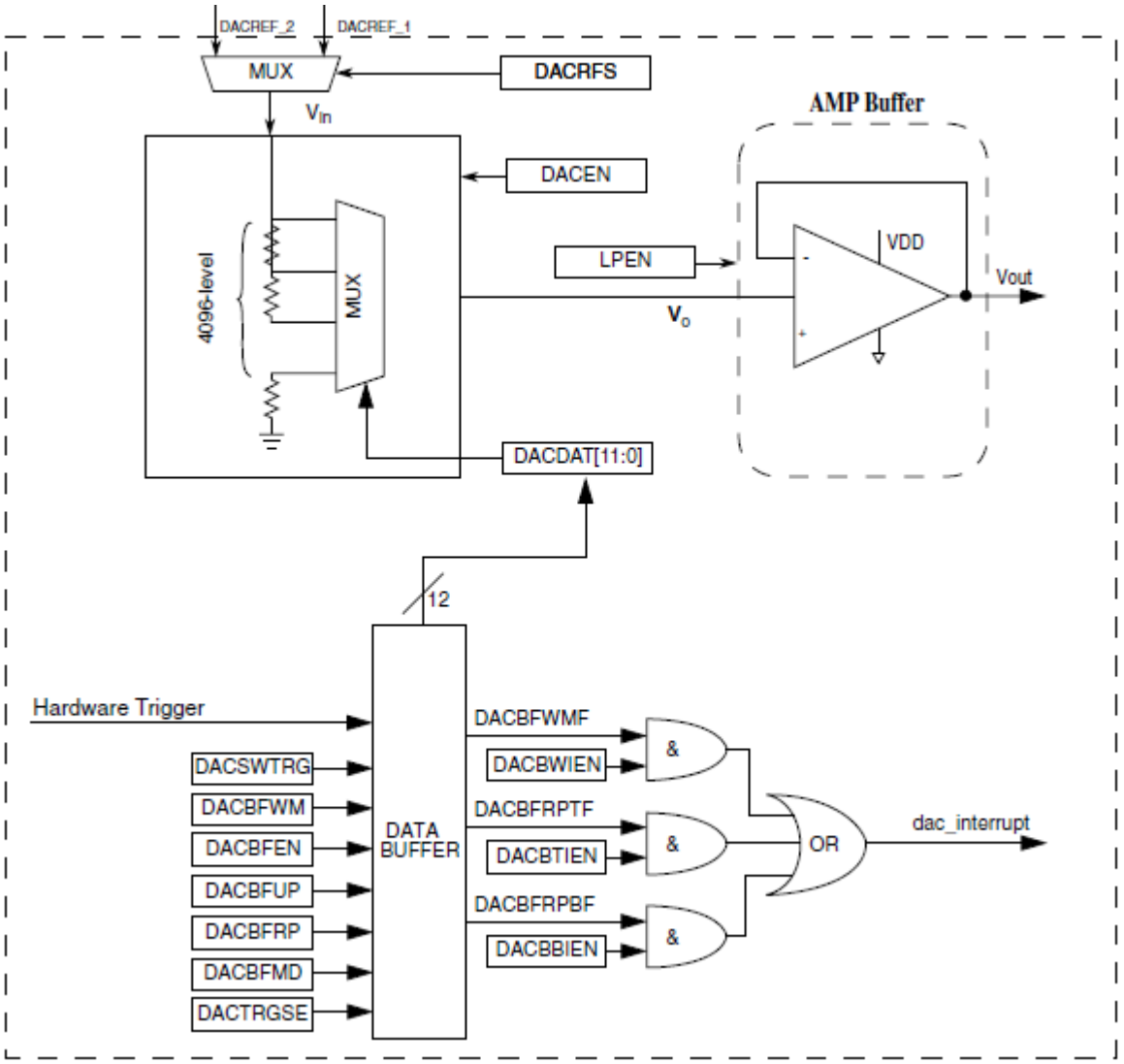
This block is designed to work with differential input and output with input signals that range from 0 -1.2 V — 10 mV. The output common mode of the PGA is determined based on the SAR ADC requirement.



12-bit Digital-to-Analog Converter (DAC)

The 12-bit digital-to-analog converter (DAC) is a low power general purpose DAC. The output of this DAC can be placed on an external pin or set as one of the inputs to the analog comparator, Op-Amps, ADC, or other peripherals.

The DAC module features include:
- On-chip programmable reference generator output
- Static operation in Normal Stop mode
- DMA support



DAC Memory Map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
400C_C000	DAC Data Low Register (DAC0_DAT0L)	8	R/W	00h
400C_C001	DAC Data High Register (DAC0_DAT0H)	8	R/W	00h
400C_C002	DAC Data Low Register (DAC0_DAT1L)	8	R/W	00h
400C_C003	DAC Data High Register (DAC0_DAT1H)	8	R/W	00h
400C_C004	DAC Data Low Register (DAC0_DAT2L)	8	R/W	00h
400C_C005	DAC Data High Register (DAC0_DAT2H)	8	R/W	00h
400C_C006	DAC Data Low Register (DAC0_DAT3L)	8	R/W	00h

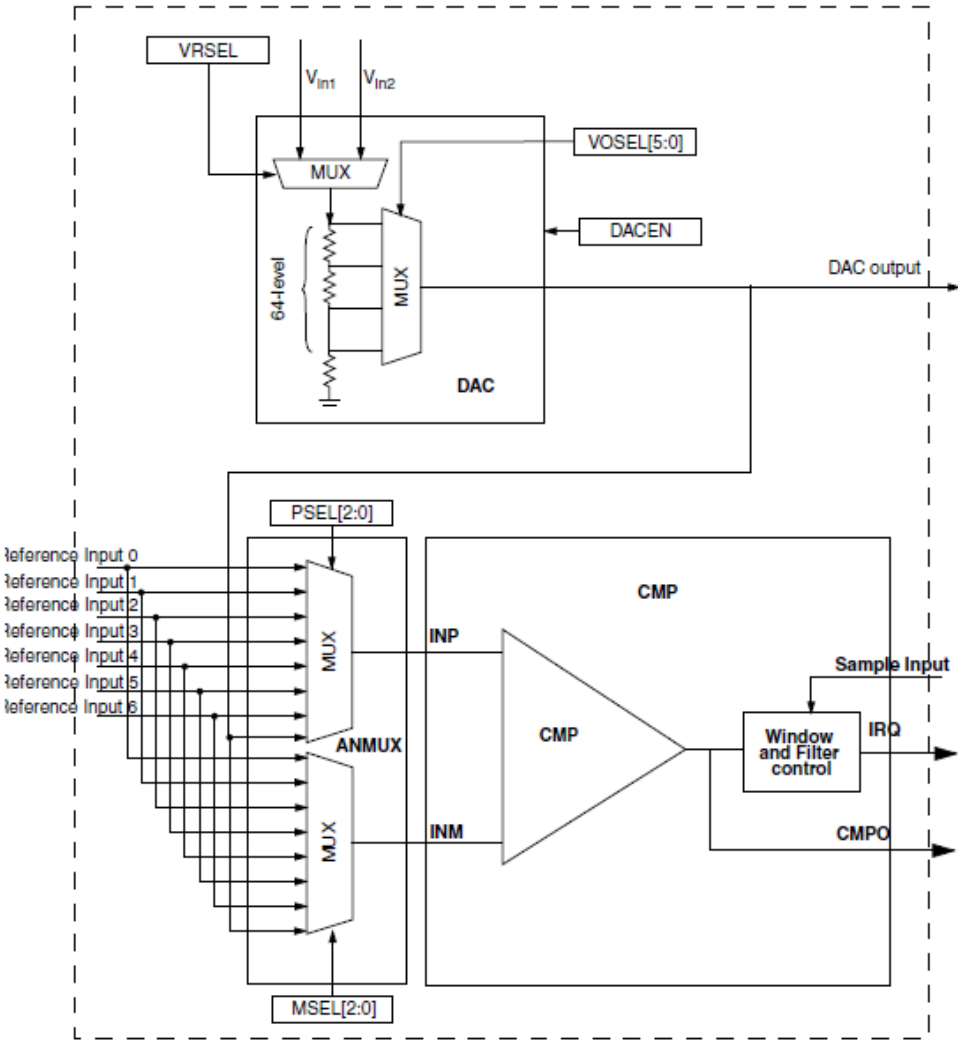
... alltogether 72 Registers ...

Comparator (CMP)

The Comparator module (CMP) provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage (rail to rail operation).

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal provided by the 6-bit DAC. The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed.



Comparator Memory Map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value
4007_3000	CMP Control Register 0 (CMP0_CR0)	8	R/W	00h
4007_3001	CMP Control Register 1 (CMP0_CR1)	8	R/W	00h
4007_3002	CMP Filter Period Register (CMP0_FPR)	8	R/W	00h
4007_3003	CMP Status and Control Register (CMP0_SCR)	8	R/W	00h
4007_3004	DAC Control Register (CMP0_DACCR)	8	R/W	00h
4007_3005	MUX Control Register (CMP0_MUXCR)	8	R/W	00h
4007_3008	CMP Control Register 0 (CMP1_CR0)	8	R/W	00h

... alltogether 18 Registers ...

Analog Interrupts

Address	Vector	IRQ[1]	NVIC non-IPR register number [2]	NVIC IPR register number [3]	Source module	Source description
0x0000_0124	73	57	1	14	ADC0	—
0x0000_0128	74	58	1	14	ADC1	—
0x0000_012C	75	59	1	14	CMP0	—
0x0000_0130	76	60	1	15	CMP1	—
0x0000_0134	77	61	1	15	CMP2	—
0x0000_0184	97	81	2	20	DAC0	—
0x0000_0188	98	82	2	20	DAC1	—

Literature and Links

http://www.arm.com/products/processors/cortex-m/cortex-m3.php

http://www.arm.com/products/processors/technologies/instruction-set-architectures.php

http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=K60

K60 Sub-Family Reference Manual

Kinetis Peripheral Module Quick Reference

Cortex-M4 Technical Reference Manual

http://www.arm.com/products/processors/cortex-m/index.php

TECHNISCHE HOCHSCHULE DEGGENDORF

Technische Hochschule Deggendorf – Edlmairstr. 6 und 8 – 94469 Deggendorf