XMC4000

32-bit Microcontroller Series for Industrial Applications

Delta Sigma Demodulator (DSD)

AP32302

Application Note

About this document

Scope and purpose

This document describes the features of the Delta Sigma Demodulator (DSD) peripheral for the XMC4000 microcontroller family. This document also describes how to configure the DSD for a number of use cases such as high resolution measurement, fast overcurrent detection and resolver interface.

Applicable Products

- XMC4000 Microcontrollers Family

References

Infineon: Example code: http://www.infineon.com/XMC4000 Tab: Documents

Infineon: XMC Lib, http://www.infineon.com/DAVE

Infineon: DAVE™, http://www.infineon.com/DAVE

Infineon: XMC Reference Manual, http://www.infineon.com/XMC4000 Tab: Documents

Infineon: XMC Data Sheet, http://www.infineon.com/XMC4000 Tab: Documents

Table of Contents

1 Delta Sigma (ΔΣ) Basics

The delta-sigma (ΔΣ) principle is a digital signal processing method for encoding analog signals into digital signals. It consists of a delta-sigma modulation (DSM) for encoding analog signals into a bitstream, as found in an ADC, and the delta-sigma demodulation (DSD) for encoding the bitstream into a data word, as found in the XMC microcontroller.
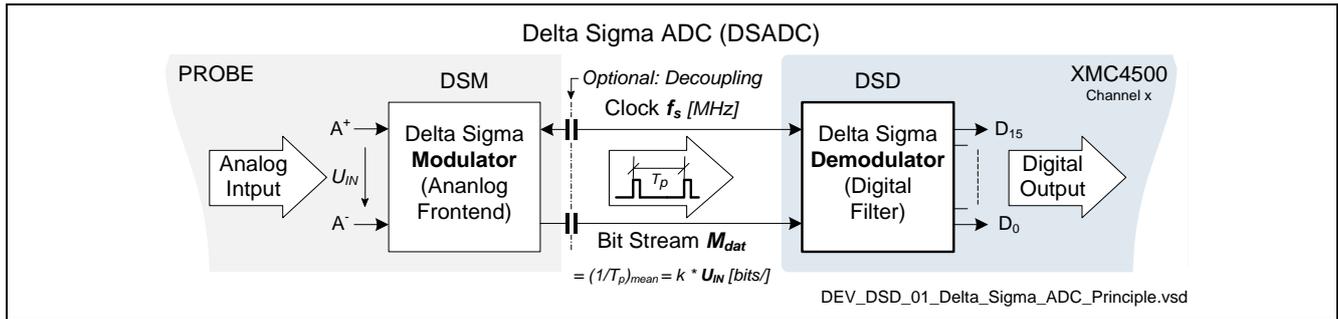

Figure 1   Delta Sigma ADC (DSADC) Principle

Compared to other ADC methods like successive-approximation-register (SAR) the ΔΣ ADC is more flexible in terms of dynamics and resolution. Also, this method can be used for very high resolution sampling. Due to the state of data as bitstream the connection can easily be isolated. This can be used for safety isolation to protect against human shock, or functional isolation to level shift between nonlethal voltages.

1.1   Delta Sigma Modulator

A Delta Sigma Modulator converts an analog input signal (Vin) to a digital output Bitstream. The bit density (Mdat bits/s) is proportional to the analog mean value. This is given by the equilibrium condition ΔV=0 at the integrator input, where the analog signal is balanced to the feedback of impulse quanta (Vref* tp) from the output Bitstream.

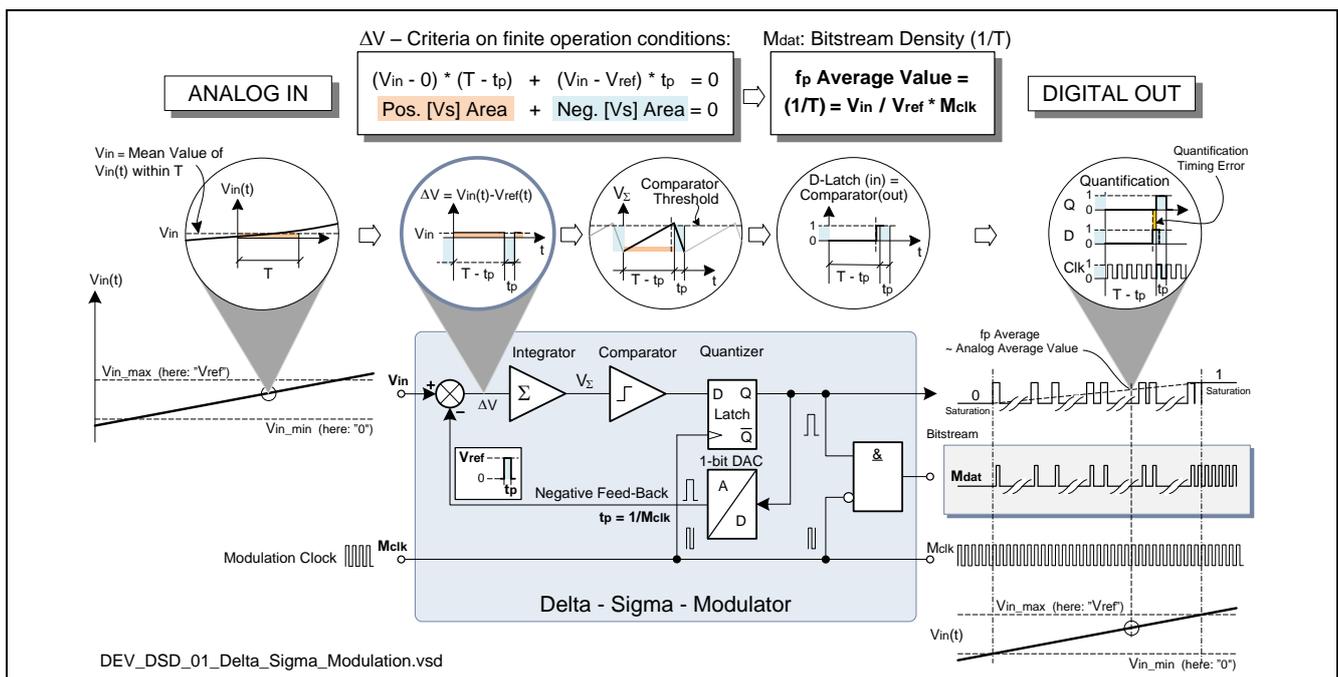**Figure 2   Delta Sigma Modulator – 1st Order**

Delta Sigma (ΔΣ) Basics

A 2nd order (or higher) DSM is more precise than a 1st order DSM. It can process input signals at a higher bandwidth and at a lower clock rate. The output from a 2nd order DSM has less noise and is much closer to the ideal pulse proportion output signal. In particular, a 2nd order DSM significantly avoids "non-random" noise.
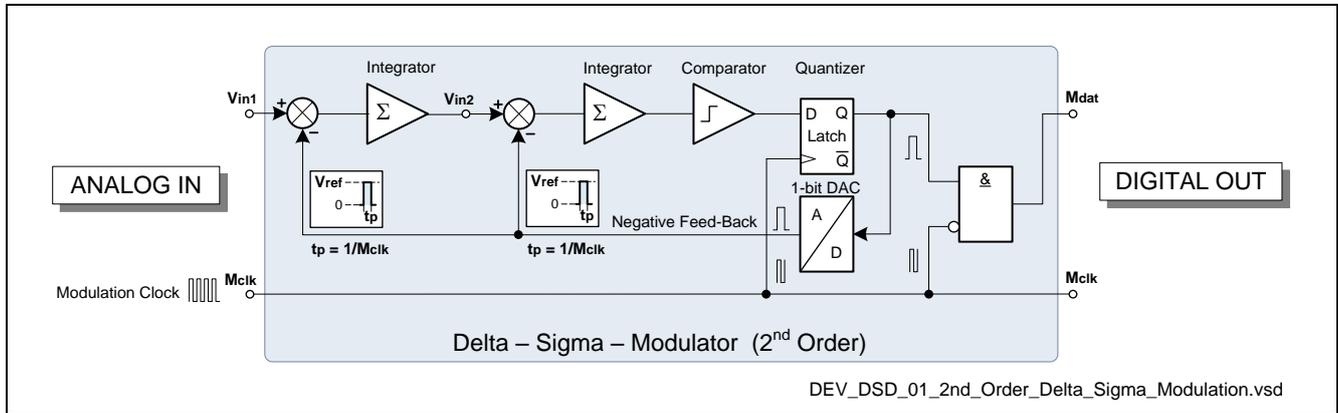

Figure 3   Delta Sigma Modulator – 2nd Order

To reconstruct a signal, the sampling frequency (fs) needs to be more than double signal frequency ($f_{Nyquist}$).

$$f_s > 2 * f_{Nyquist}$$

This provides anti-aliasing. Due to the delta sigma method the sampling frequency is factors higher than the signal frequency.
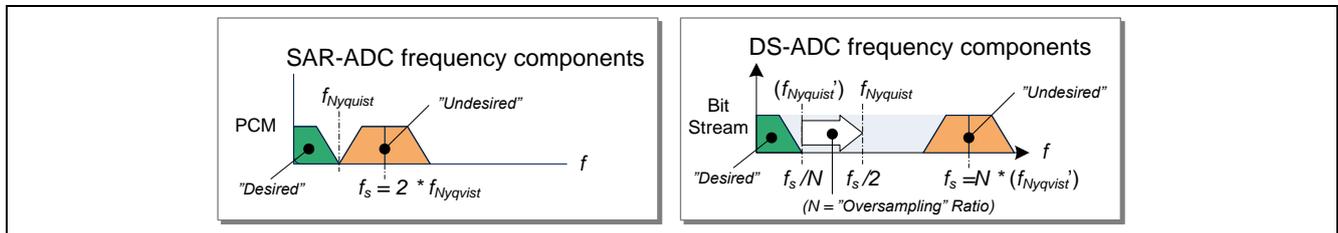

Figure 4   ΔΣ ADC- Oversampling

1.2 Delta Sigma Demodulator

The Delta Sigma Demodulator (DSD) task is to extract the analog information out of a bitstream. For this purpose the essential parts are generally a digital decimation filter - and optionally data refinement.

Figure 5 Delta Sigma Demodulator Filter shows the digital filter structure of a Cascaded Integrator-Comb (CIC) filter -without data refinement. These type of filters are defined by the cascade stage (k) and the decimation factor (N). These factors influence the filter characteristics: response time, resolution, output period and filter group delay.
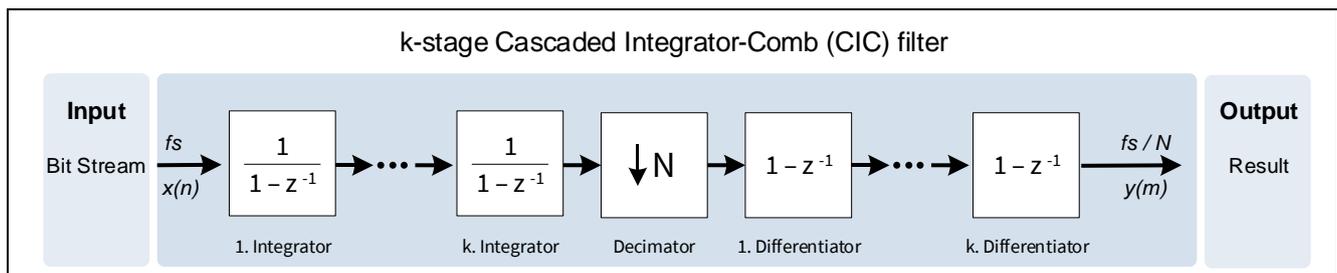
Figure 5   Delta Sigma Demodulator Filter

When the filter is cleared a new valid value is available after the filter response time (T_r). The response time can be calculated by:

$$T = k * {}_N T_r = k * \frac{N}{f_s}$$

After the filter output period (T_{Out}) a new valid value is available in the result register. This time can be calculated by:

$$T_{Out} = \frac{N}{f_s} T_f = \frac{N}{f_s} T_f = \frac{N}{f_s}$$

The resolution in bits (p_K) can be calculated by the following equation. Note that the sign bit is included in the calculation.

$$p_K = 1 + k * \log_2 N$$

The filter group delay (T_G) in the steady state can be calculated by:

$$T_G = k * \frac{N-1}{2}$$

2 XMC DSD implementation

The delta sigma demodulator (DSD) unit in the XMC4000 family offers 4 DSD channels. Each channel can be connected to standard external delta sigma modulators via selectable bit stream inputs and clock input or output. There are configurable CIC filters with decimation rates of 4 - 256, offset compensation, fast limit checking and facilities for resolver applications such as a carrier generator.
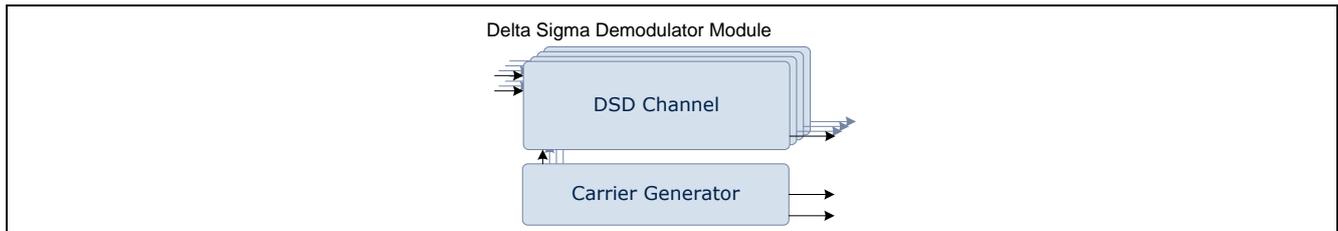
Figure 6 **Delta Sigma Demodulator Module**

2.1 XMC DSD Unit overview

Each DSD channel has a dedicated *Main Filter* with an associated *Auxiliary Filter* in parallel. The *Main Filter* intend for high resolution results and provides an *Integrator* and a *Rectifier* for data refinement (such as noise or carrier rejection). The parallel *Auxiliary Filter* (Aux Filter) is intend for a fast reaction and provides fast limit checking by an upper-lower boundary *comparator.*

The *Rectify* and *Integrator* unit after the *Main Filter* and the *Comparator* after the *Auxiliary Filter* can be disabled and bypassed. If they are disabled, the output form the *Main* and *Auxiliary filter* is stored in the respective result register.

Figure 7 **DSD Channel**

2.2 XMC DSD Unit Connections
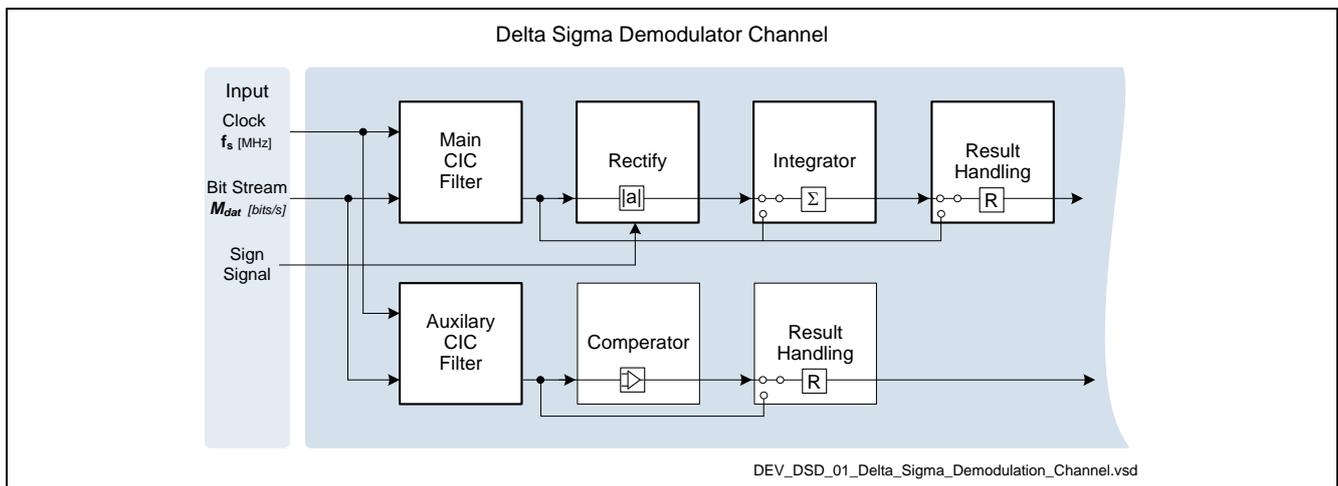
A DSD channel bit stream input (DINxA/-B) and the Modulator Clock Input/Output (MCLKxA/-B) are routed to GPIOs. The Trigger Signal Input ITRx [A…D] is routed via the Event Request Unit 1 (ERU1). This offers a trigger signal from a Timer, a GPIO, or a conditional trigger signal. Additionally, the rectification signal *SIGNA/B* for the Rectifier is also routed through the ERU1.

The Carrier Patten Generator is routed to the GPIOs and the *Carrier Sign Signal* is routed to the Rectify unit. The *Service Request Main* output (*SRMx*) is linked to NVIC interrupts and GPDMA service providers and the corresponding *Service Request Auxiliary* (SRAx) is linked only to the NVIC interrupt network.
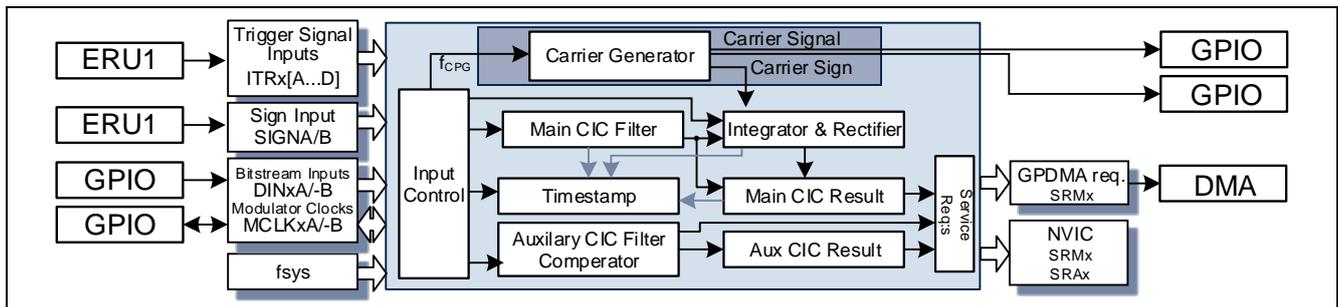
Figure 8   DSD interconnection

The bit stream input and the modulator clock is connected to *the Main CIC Filter* and the *Auxiliary CIC Filter*. The trigger signal input is connected to the *Integrator* and the *Timestamp*. A timestamp trigger stores information from the *Main CIC Filter*, the *Main CIC Result* and the *Integrator*. The *Auxiliary CIC Filter* and *Main CIC Filter* are routed to the respective result registers. Alternatively the *Main CIC Filter* can be routed to the *Integrator & Rectifier* and then to the *Main CIC Result*. The sign signal for the *Rectifier* can come from the *Carrier Generator* or from SGNA/B.

3 CIC Filter

The DSD channel Cascaded Integrator-Comb (CIC) filters, Main CIC Filter and Auxiliary CIC Filter, can be enabled for "CIC^k" characteristics. Where k is in range of k=1-3 or F for CIC^1, CIC^2, CIC^3 or CIC^F. The decimation factor can be selected from 4-256 in the Main CIC Filter and from 4-32 in the Auxiliary CIC filter.

3.1 CIC Filter configuration

A higher filter stage increases the resolution but also increase the response time. The demodulator filter stage should be higher than the modulator stage. The decimation factor has influence on the resolution, the response time and the filter output period. A higher decimation factor increases the resolution but also increases the response time and the filter output period.
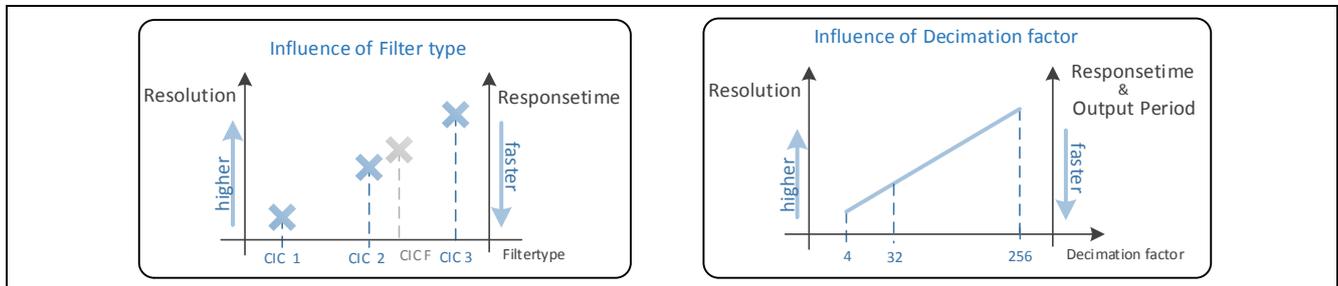

Figure 9   Filter configurations overview

The following table shows the filter calculations for a sampling frequency of 10MHz. The calculations are based on the formulas in 1.2 Delta Sigma Demodulator.

Table 1 **Common filter configurations**

Filter type CIC^k	Decimation factor (N)	Response time (T_r) [µs]	Filter output period (T_{Out}) [µs]	Resolution (p_K) [Bit]
CIC^2				
	16	3.2	1.6	9
	32	6.4	3.2	11
	64	12.8	6.4	13
	128	25.6	12.8	15
	256	51.2	25.6	17
CIC^3				
	16	4.8	1.6	13
	32	9.6	3.2	16
	64	19.2	6.4	19
	128	38.4	12.8	22
	256	76.7	25.6	25
CIC^F				
	16	3.4	1.6	9
	32	6.6	3.2	11
	64	13.0	6.4	14
	128	26.0	12.8	16
	256	52.0	25.6	18

Note: *The XMC DSD provides a 16bit result register. Due to filter type and decimation factor a resolution over 16bit can be achieved. To avoid overflow, the result is automatically shifted based on the decimation factor and filter type.*

3.2 Use case: High resolution ADC measurement

Some applications require an ADC resolution higher than 12bit. For these cases, the DSD ADC has a 16 bit result register.

If the DSM provides a higher resolution than 16bit the XMC DSD can also be configured to support this higher resolution. To avoid an overflow, the result is automatically shifted based on the decimation factor and filter type to a 16bit value.

In the following example, there is no additional data refinement such as an auxiliary filter or integrator.

An external two stage delta sigma modulator with 16bit resolution, 14bit accuracy and 10MHz sample frequency is used.

The Main CIC Filter is configured as CIC^3 to match the two stages modulator. The dynamic of the measured signals allows a slower response time and a 16bit resolution can be achieved with a smaller decimation factors. The decimation factor is set to 32. This will lead to a 16bit resolution, a response time of 9.6µs and a filter output period of 3.2µs.

The 10MHz clock signal is generated from the external modulator. The modulator clock input and sample frequency are equal. The Data is sampled at rising clock edge.

3.2.1 XMC Lib configuration

In this example the XMC4400 peripheral clock frequency is 120MHz. The clock is provided by an external device, therefore the clock_source is XMC_DSD_CH_CLOCK_SOURCE_A and the clock_divider configuration has no influence. The external clock and the bit stream have the same frequency. Therefore a strobe of XMC_DSD_CH_STROBE_DIRECT_CLOCK_RISE is used.

The decimation factor of 32 together with the filter type XMC_DSD_CH_FILTER_TYPE_CIC3 fulfills the request of 16bit resolution with a good accuracy. With different start_values two or more filters can be shifted to each other. This is not necessary in this example therefore the decimation factor is used.

The offset can be used to compensate a HW offset. By default there is no compensation.

```
XMC_DSD_CH_FILTER_CONFIG_t filter_1 = {
        .clock_divider = XMC_DSD_CH_CLK_DIV_2,
        .clock_source =  XMC_DSD_CH_CLOCK_SOURCE_A,
        .data_source = XMC_DSD_CH_DATA_SOURCE_A_DIRECT,
        .decimation_factor = 32U,
        .filter_start_value = 32U,
        .filter_type = XMC_DSD_CH_FILTER_TYPE_CIC3,
        .offset = 0U,
        .result_event = XMC_DSD_CH_RESULT_EVENT_DISABLE,
        .strobe = XMC_DSD_CH_STROBE_DIRECT_CLOCK_RISE,
    } ;
```

3.2.2 Initialization

In this example, the DSD_CH0 is configured with the filter_1 configuration specified in the previous section. The initialization sequence is important. Make sure that the DSD is enabled before the Main Filter is initialized. The GPIO ports should be initialized after the filter initialization.

```
        XMC_DSD_Enable(DSD);
        XMC_DSD_EnableClock(DSD);


        XMC_DSD_CH_MainFilter_Init(DSD_CH0,&filter_1);
```

After initialization the module can be started.

Note: If two filters are used simultaneously both should be started with one function call (for example: XMC_DSD_CH_ID_0| XMC_DSD_CH_ID_1).

```
XMC_DSD_Start(DSD,XMC_DSD_CH_ID_0);
```

3.2.3   Function implementation

The result can now be readout in a loop. Remember that the result is updated every 3.2μs due to the clock input and decimation factor.

```
        XMC_DSD_CH_GetResult(DSD_CH0,&result0);
```

4 Timestamp and Offset

The sampling frequency of a ΔΣ ADC is many times less than an SAR ADC. This is due to the system structure and the lower sample frequency. Due to external components the data can be contain a constant offset. In this case this offset has to be subtracted to gain the correct data.

4.1 Timestamp implementation

In many applications, the measured ADC data needs to available at a pre-determined frequency. This frequency might be out of sync with the ΔΣ ADC cycle so that data needs to be extrapolated.

Therefore it is necessary to know the last valid result and the age of the result.

For this use case the XMC DSD provides a Timestamp function. The last result, the decimation counter (d_{count}) and the integration counter (i_{count}) are stored in the timestamp register when a trigger occurs. (See: Figure 8 DSD interconnection).

With decimation factor (N) the age of the result can be calculated in DSD clock cycles:

$$clock_{cycle} = i_{count} * N + (N - d_{count})$$

4.2 Offset implementation

In some applications a permanent offset is added to the result. The XMC DSD provides the possibility to subtract a 16bit signed offset value before the data is stored into the result register. This subtraction is done in hardware so there is no load added to the CPU.

4.3 Use case: Extrapolation of ADC result using a timestamp

The main program cycle time is 10µs so an ADC value is required in this period. The configured output period is 3.2µs so jitter is introduced if there is no synchronization. To avoid jitter based on the DSD filter output period and the program cycle time, the ADC data needs to be extrapolated. Also a known offset of 100mV has to be subtracting from the result.

The CCU40.CC40 timer is used to trigger the timestamp of the DSD every 10µs. The timer also triggers an interrupt service routine which reads the timestamp register. The age of the result is calculated using the formula in section above.

The offset is 100mV, the input range in the used hardware is ±2.5V, and the resolution is configured to 16bit signed. This leads to an offset of (int16_t) 1310.

4.3.1   XMC Lib configuration

Main filter configuration

The Main filter configuration is the same as in section 3.2Use case: High resolution ADC measurement is reused. But the timestamp function and an offset of 1310 is added. This offset is automatically subtracted from the result.

```
XMC_DSD_CH_FILTER_CONFIG_t filter_1 = {
        .clock_divider = XMC_DSD_CH_CLK_DIV_2,
        .clock_source =  XMC_DSD_CH_CLOCK_SOURCE_A,
        .data_source = XMC_DSD_CH_DATA_SOURCE_A_DIRECT,
        .decimation_factor = 32U,
        .filter_start_value = 32U,
        .filter_type = XMC_DSD_CH_FILTER_TYPE_CIC3,
        .offset = 1310U,
        .result_event = XMC_DSD_CH_RESULT_EVENT_DISABLE,
        .strobe = XMC_DSD_CH_STROBE_DIRECT_CLOCK_RISE,
    } ;
```

Timestamp configuration

The trigger_mode configures the timestamp to react on rising edge. With the trigger source A on the XMC4400 the ERU1.PDOUT1 is selected. A timer triggers the DSD through the ERU. The ERU and Timer (CCU4) configurations are described in the respective Application Notes.

```
XMC_DSD_CH_TIMESTAMP_CONFIG_t timestamp_1= {
        .trigger_mode = XMC_DSD_CH_TIMESTAMP_TRIGGER_RISE,
        .trigger_source = XMC_DSD_CH_TRIGGER_SOURCE_A,
        };
```

4.3.2   Initialization

In this example, the DSD_CH0 is configured with the filter_1 configuration defined in the previous section. The initialization sequence is important. Make sure that the DSD is enabled before the Main Filter is initialized. The Timestamp should be initialized after the filter initialization.

```
XMC_DSD_Enable(DSD);
XMC_DSD_EnableClock(DSD);

XMC_DSD_CH_MainFilter_Init(DSD_CH0,&filter_1);
XMC_DSD_CH_Timestamp_Init(DSD_CH0,&timestamp_1);
```

After initialization the module can be started.

Note: If two filters are used simultaneously both should be started with one function call (e.g. XMC_DSD_CH_ID_0| XMC_DSD_CH_ID_1).

```
XMC_DSD_Start(DSD,XMC_DSD_CH_ID_0);
```

4.3.3   Function implementation

The result can now be readout in a loop. Remember that the result is updated every 10µs due to the trigger input.

```
XMC_DSD_CH_GetResult_TS_Time(DSD_CH0,&result0,&time0);
```

5 Auxiliary Filter with Comparator

In some applications not only a high resolution but also a fast reaction is necessary. As described in section 3 CIC Filter the filter run and response time of a $\Delta\Sigma$ ADC system are influenced by the filter type and the decimation factor. The two requirements of high resolution and fast reaction are conflicting because a smaller filter stage (k) and lower decimation factor (N) decreases the filter response and run time but also reduces the resolution.

5.1 Auxiliary Filter implementation

To meet this requirements two filters in parallel are implemented. One main filter, for normal operation, and one auxiliary filter, with a fast reaction on invalid ADC values. The reaction time on an invalid ADC value can be further decreased by checking the auxiliary filter result in hardware and then triggering an interrupt.

For this use case each XMC DSD channel provides an auxiliary filter in parallel to the main filter. The auxiliary filter provides the same filter stages (k) as the main filter and offers a decimation factor from 4-32.

5.2 Comparator implementation

The result of the auxiliary can be checked in hardware against a boundary band and an event is triggered if the result is outside or inside the band.
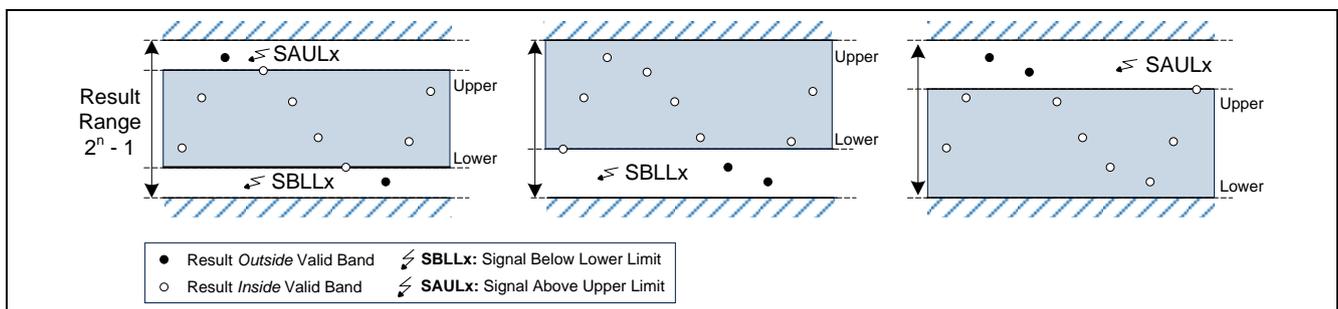
Figure 10   Auxiliary Filter boundaries

5.3 Use case: Fast overcurrent detection

This example uses the delta sigma modulator from 3.2 Use case: High resolution ADC measurement project. Configurations are also the same. An auxiliary filter and a comparator are added.

In this example the ΔΣ ADC measures a phase current and an overcurrent needs to be detected in less than 2µs. To get a high resolution the main filter is configured as CIC^3 with a decimation factor of 32. This leads to a response time of 9.6µs and a filter output period of 3.2µs. The ΔΣ ADC can detect up to ±3A. Currents over ±2.5A are defined as overcurrent.

An auxiliary filter configured as CIC^3 and a decimation factor of 4 has a response time T_r =1.2µs, a filter output period T_{Out} = 0.4µs and a resolution of 7bits. This leads to a safe detection in 1.6µs with a resolution of 0.05A.

Therefore, the boarders are set to UPPER= 53 (+2.5A) LOWER = -53 (-2.5A).
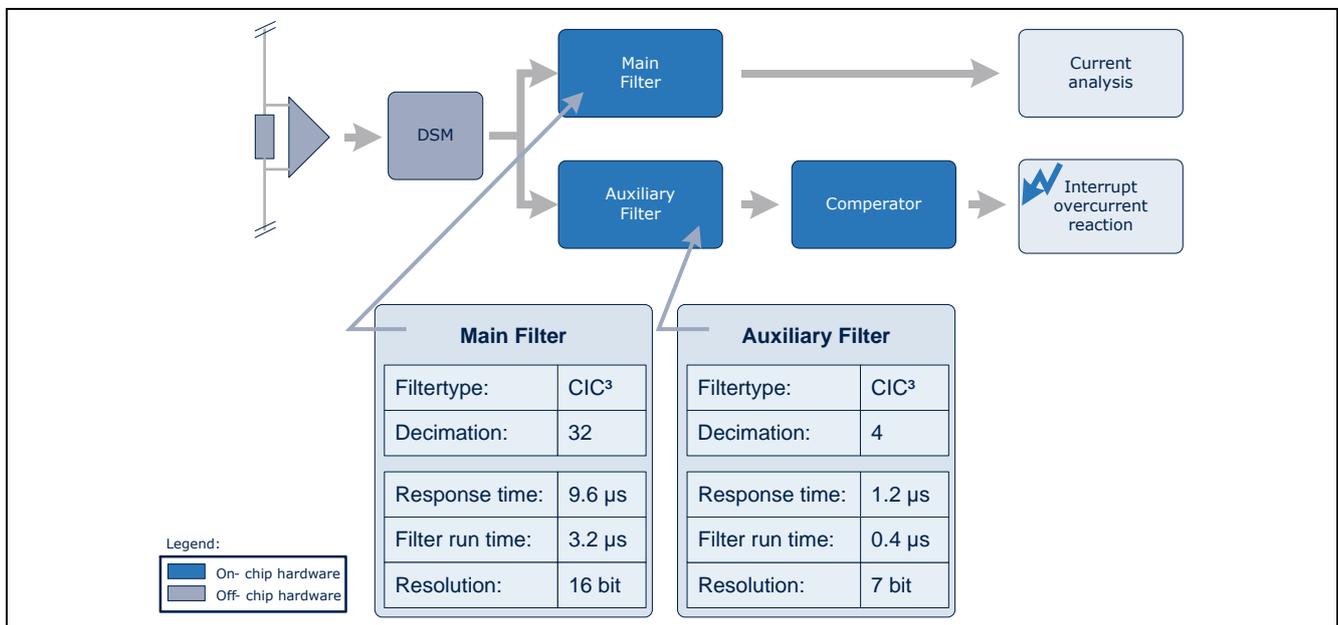
Figure 11   DSD Main and Auxiliary filter configuration

6 Triggered Measurement

The ΔΣ ADC is very well suited for continuous signal measurement. However, in some applications only a small measurement window is available, or the measurement has to be synchronized.

6.1 Fixed and variable integration window

The XMC DSD provides trigger functionality in the integrator stage. This allows the measurement to be started with a rising or falling edge of the trigger. Depending on the application the integration window can be fixed by the number of results or can vary depending on the length of the trigger.
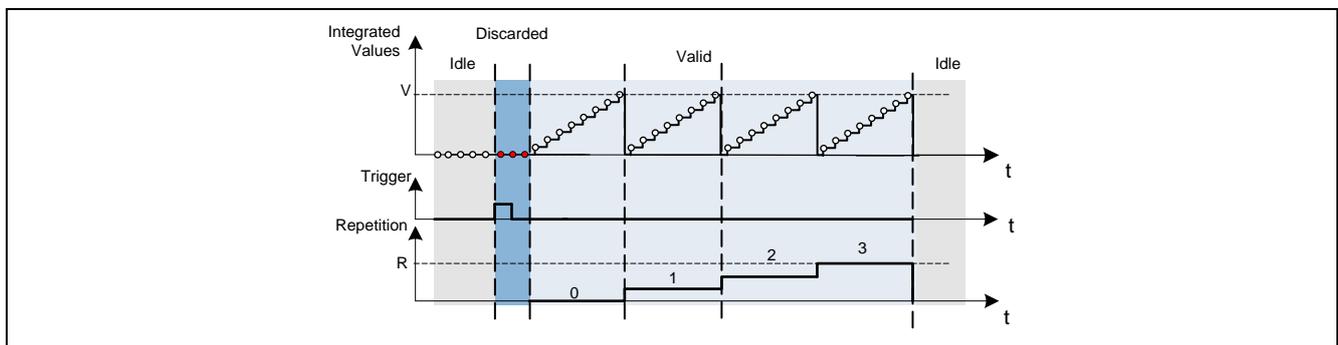
Figure 12 Triggered measurement with fixed window

Figure 12 shows a Triggered measurement with a fixed window. A trigger clears the filter and the integrator. At the same time it starts a new measurement. First a user defined number of filter results are discarded (discarded window). This allows to blank the filter response time and to add an additional delay. The filter response time depends on the filter type. After the discarded window the integration window starts. A user defined amount of filter results [V] are integrated. When this number is reached a new data is available in the result register. This also increases the repetition counter [R]. After a user defined number of repetition the integration stops.
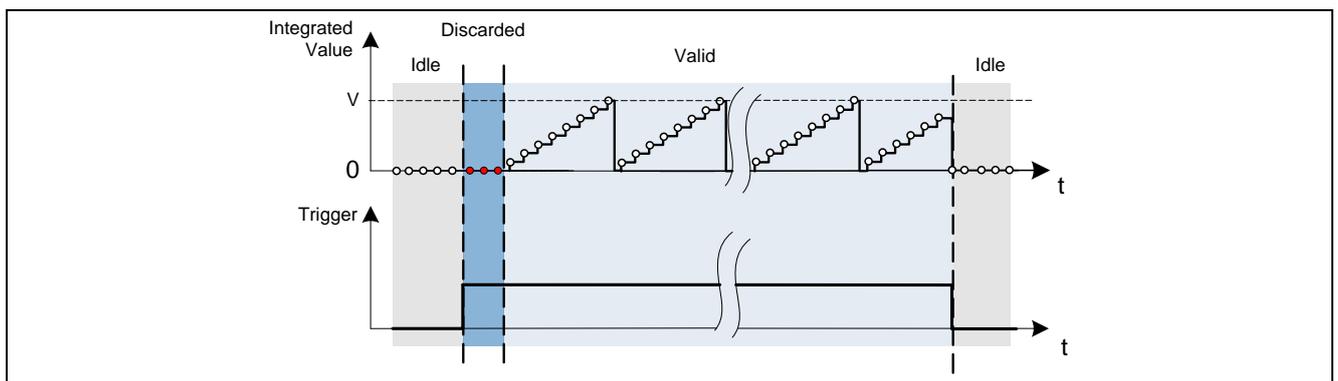
Figure 13 Triggered measurement with variable window

Different from a fixed window, a triggered measurement with a variable window repeats the measurement not based on the repetition counter but on the end of the Trigger. Therefore the integration repetition is variable. Is the window closed by the trigger, the filter and the integrator are cleared. If the trigger ends in the middle of integration, the result from the last full integration is available in the result register.

Triggered Measurement

With both methods up to 64 main filter results can be integrated to one integration result. To avoid overflow the integration result is automatically shifted right by 6 bits (divided by 64) independent of the number of integrated values.

6.2 Use case: Periodic measurement with fixed window

This example uses the same delta sigma modulator configuration from the 3.2 Use case: High resolution ADC measurement. An integrator stage and an integrator start trigger are added.

In this example, the signal measured by the $\Delta\Sigma$ ADC has a noise pulse that occurs periodically at 15 kHz. A trigger marks the beginning of the noise which lasts for a maximum for 2 µs. One measurement has to be done after each noise pulse. To blank the noise pulse the $\Delta\Sigma$ ADC is started via the trigger and stopped after one measurement.

For this use case the CIC^3 is used to meet the DSM requirements. A decimation factor of 16 leads to a filter output period of 1.6 µs, a response time of 4.8 µs and a resolution of 13 bit.

To keep a good resolution and to increase the accuracy, 32 of these 13 bit results are summated by the integrator. This leads to a new filter output period of 51.2 µs and a resolution of 12 bit. The resolution is decreased due to the shift after integration but the accuracy is increased due to the integration.

To avoid influence of the noise the filter is reset with the trigger. To meet the response time of 4.8µs, the first three values are discarded. Additionally this feature can blank the noise while the trigger is generated with the start of the noise. In this use case the noise pulse last for 2µs and the filter rune time is 1.6µs, when two additional two values are discarded the noise pulse is secure blanked.

With this filter configuration a result is generated 59.2µs after trigger.

7 Resolver Support

A resolver is a feedback system for an absolute angle, for example in a motor control application. In general a resolver has three coils where either one or two coils are extended and the remaining coils are measured. This Application Note covers a single excitation and dual measurement.
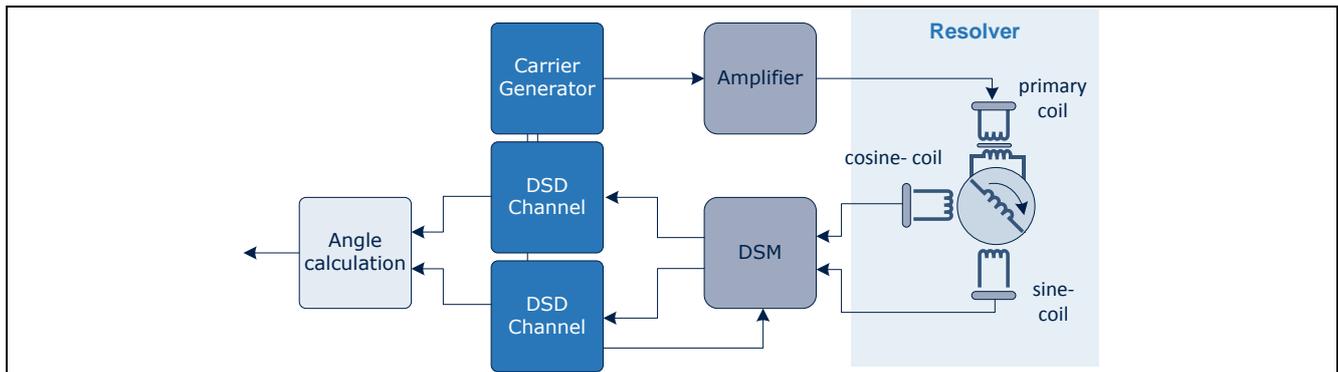

Figure 14   DSD Resolver Support

In this concept the primary coil is excited by a carrier generator with an AC voltage. This voltage is transmitted to a rotor. In the next step it is transmitted from the rotor to two 90° shifted coils (sine and cosine) where the ratio is angle depending. This leads to an angle depending amplitude of the carrier signal in the sine and cosine coil. An arc tangent (arctan) calculation of the sine and cosine amplitude will result in an angle.

While the primary coil is excited with an AC voltage, this AC signal is multiplied with the amplitude of the sine and cosine coil. This means the excitation signal has to be canceled out.
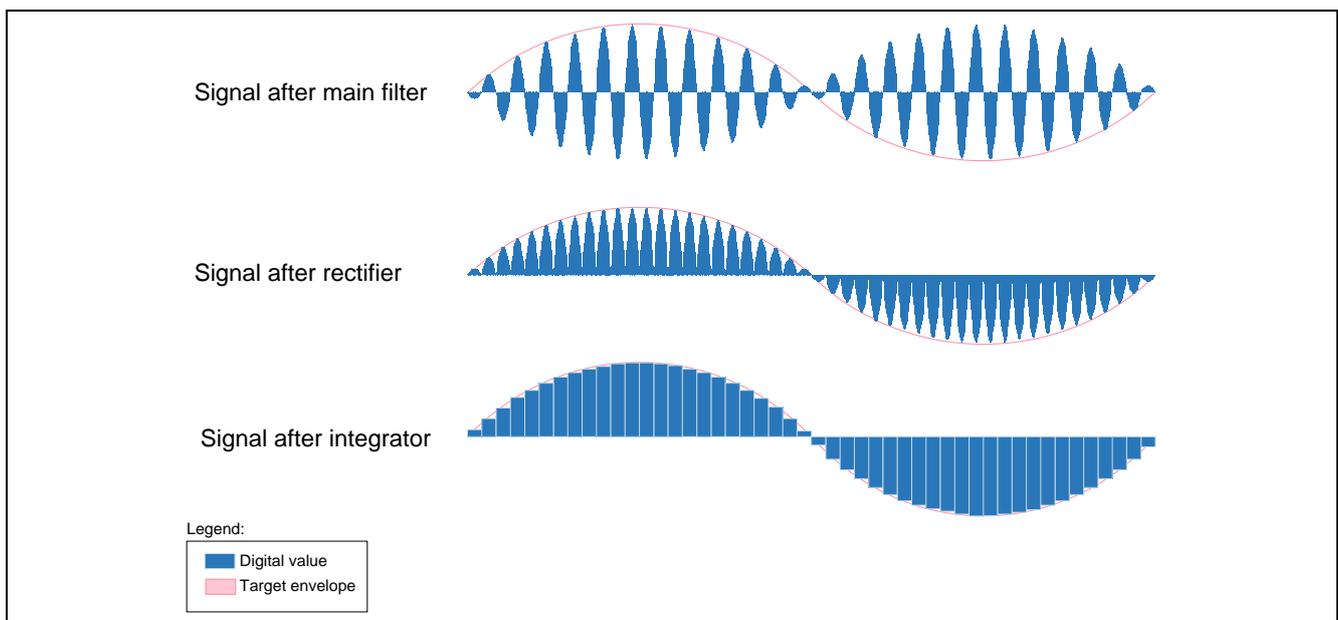
Figure 15   Signal shapes

Figure 15 shows the target envelope and the digital value after the specific part of the DSD for one mechanical rotation. To achieve the target envelope the data needs to be rectified and integrated.

Different concepts about the integration length exist. In this example the integration is done over a half carrier wave. A full period would additionally eliminate an offset error but the filter output period is longer.

When the excitation is canceled, the angle can be calculated with an arctan function or a PLL approach.

7.1 Carrier generation and ΔΣ ADC synchrony

With the DSD module, the XMC controller provides a carrier signal and full carrier cancelation in hardware. Therefore the carrier generator can generate rectangle, triangle or a sine wave with the frequency range of 3.7 to 58 kHz at 120 MHz CPU clock. The carrier generator provides a bit reverse counting for a higher switching rate. Most resolvers are specified at 10 kHz.
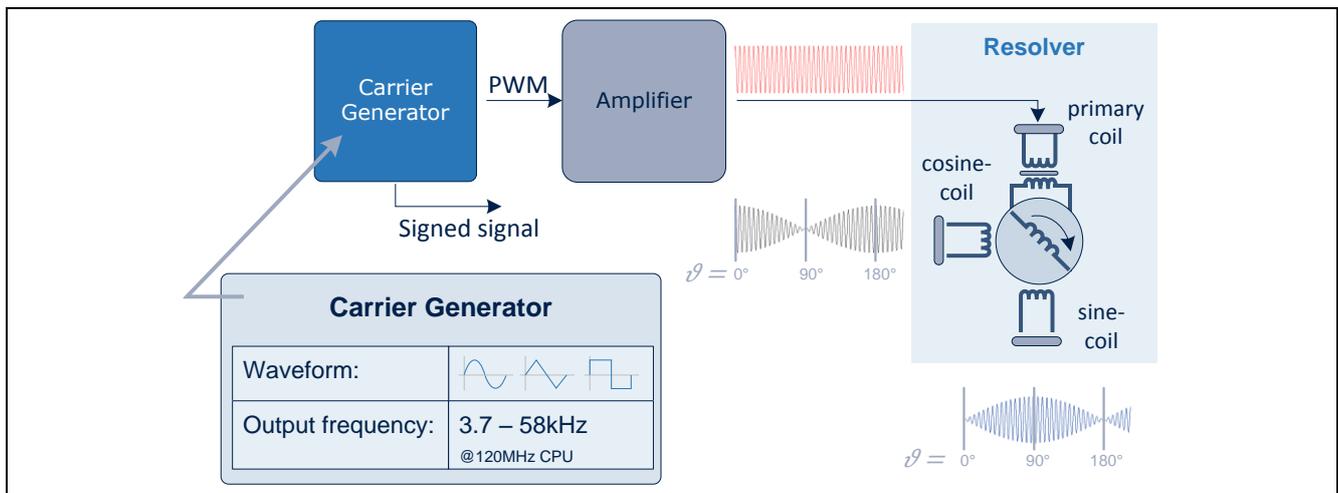
Figure 16   Carrier Generator overview

The sine and cosine amplitudes are sampled by an external DSM and the bit stream is provided for the DSD. The synchrony between DSD, DSM and carrier generator is mandatory. If they are not synchronized there is jitter in the result even at a standstill. For this synchronization the DSD XMC provides two options:

A carrier generator with a clock signal for the DSM or a clock and sign signal input from an external device.
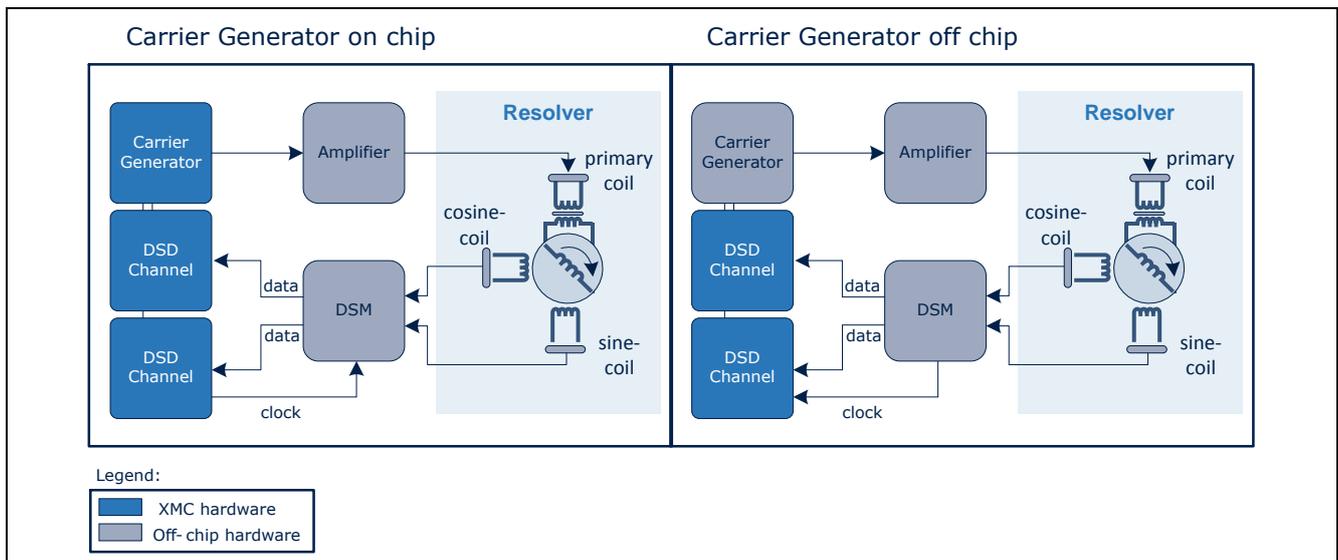
Figure 17   Carrier Generator and DSM synchronization

7.2 Rectification and delay compensation

After the main filter the sine and cosine amplitudes with the overlaid carrier signal are provided. With the rectification unit of the DSD the signals are rectified in hardware.
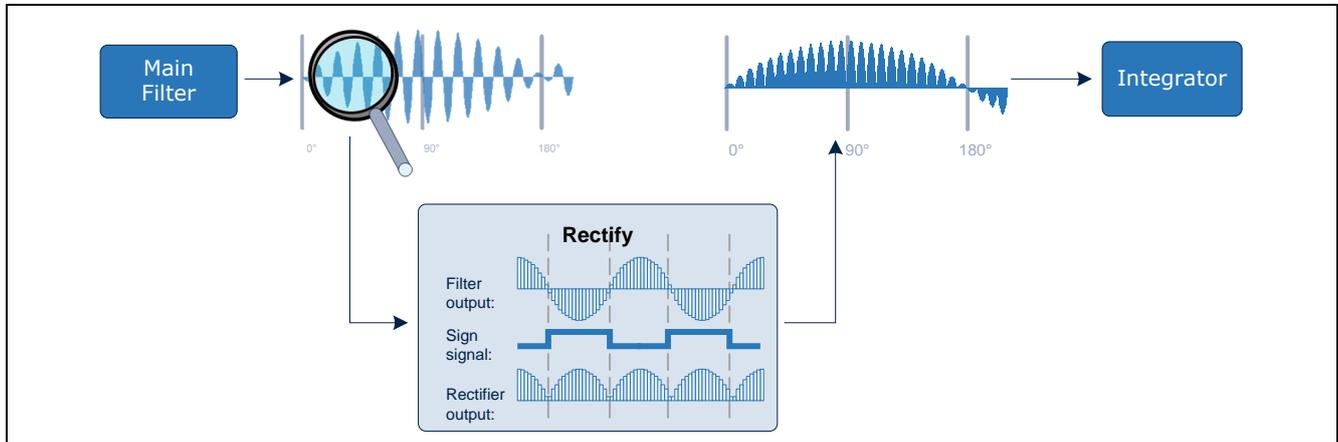
Figure 18   Rectification

For this either the sign signal from the carrier generator can be used, or if an external carrier generator is used, the sign signal can be fed in externally.

Rectification delay

The carrier generator provides the carrier signal and the sign signal.
The sign signal is directly connected to the DSD channel. The carrier signal however, has to pass the amplifier, the resolver and the Delta Sigma Moderator before it reaches the DSD channel. This creates a delay between the signal and the sign signal.
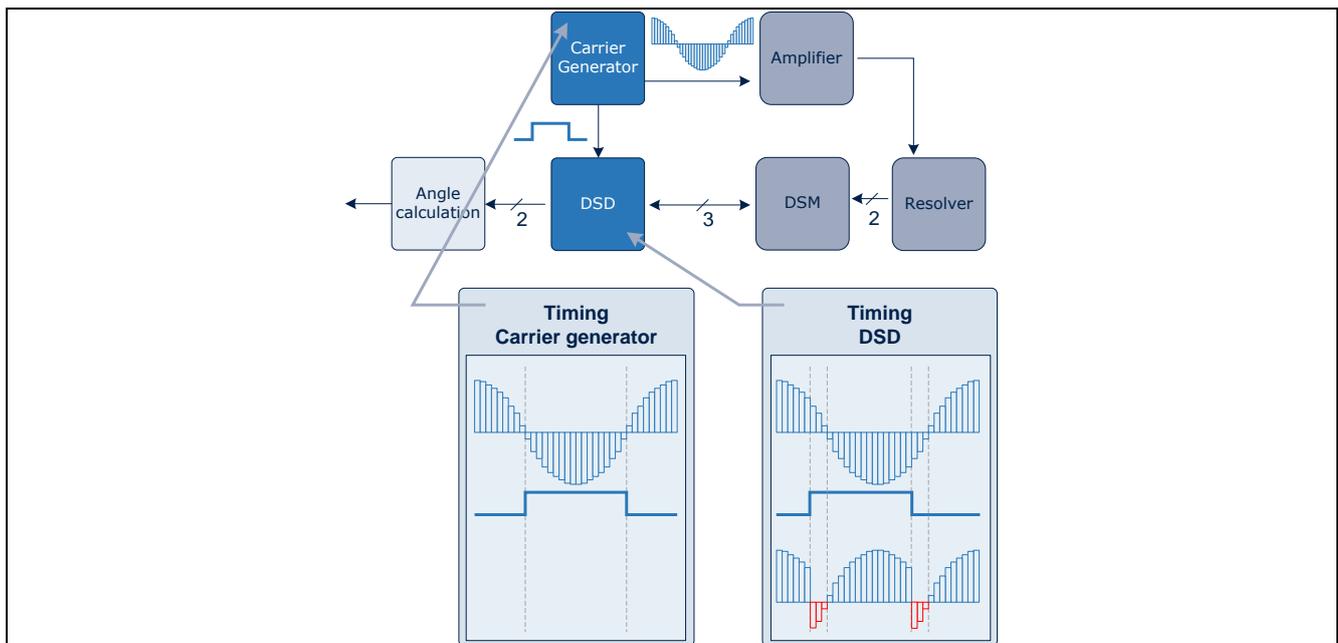
Figure 19   Rectification delay

This delay, if not compensated, leads to an incorrect rectification. Figure 19 shows a delay between carrier generator and data. In this example a delay of three filter results is shown. These results in 6 wrong rectified

values, they are red marked in the Figure 19. The values are negative instead of positive which leads to a reduced integration.

The error is visible after an integration of a measurement with and without rectification error.

The left diagram in Figure 20 shows the measurement with rectification errors. The right diagram shows the measurement without rectification errors. The maximum value for the integration is reduced. While the rectification error affects the sine and cosine signal equally the calculated angle is right, but the resolution is reduced. The XMC DSD unit can capture this delay and provide hardware delay compensation independently for each channel.
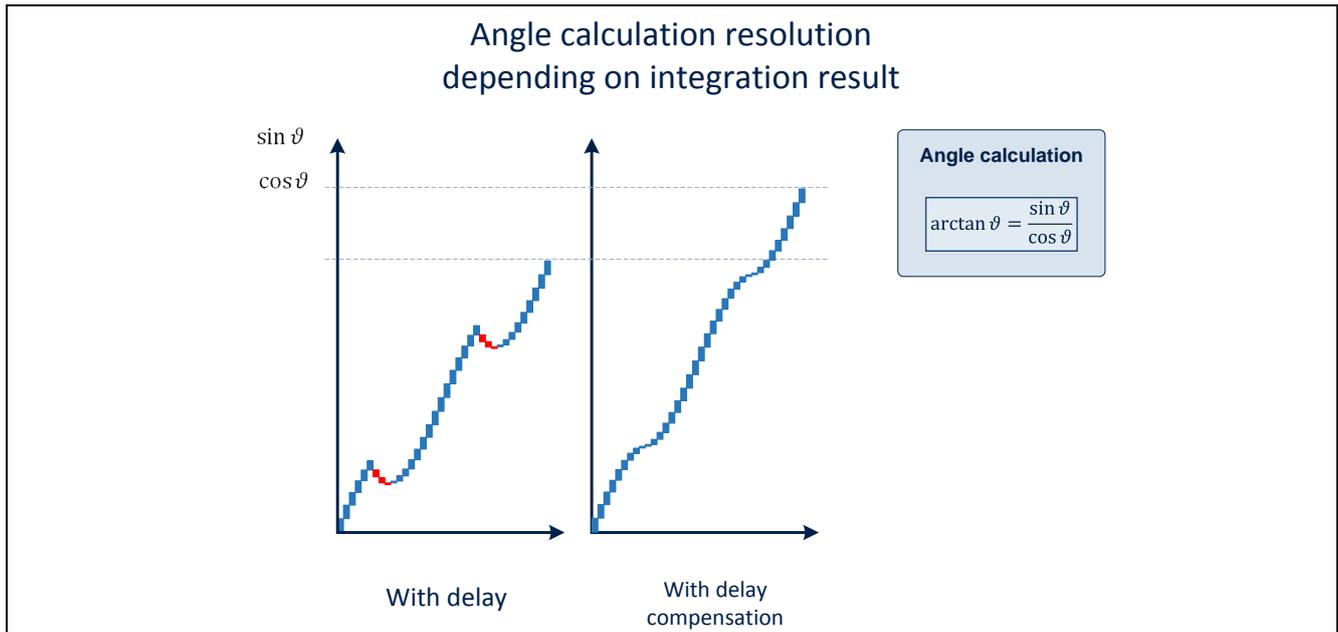
Figure 20   Angle resolution loss through rectification error

7.3   Integration and synchronization

The integration can be done in two different ways. In the continuous method the DSD is integrating continuously. DSD cycle time is the filter output period. In most applications the DSD cycle time will not match the main program cycle time. Therefore the actual angle can be extrapolated with the Timestamp function. The Timestamp function is described in 4 Timestamp and Offset. The second solution is the triggered method, here a trigger marks the start of the integration and the window size is either fixed or variable via trigger length. This method is described in 6 Triggered Measurement.
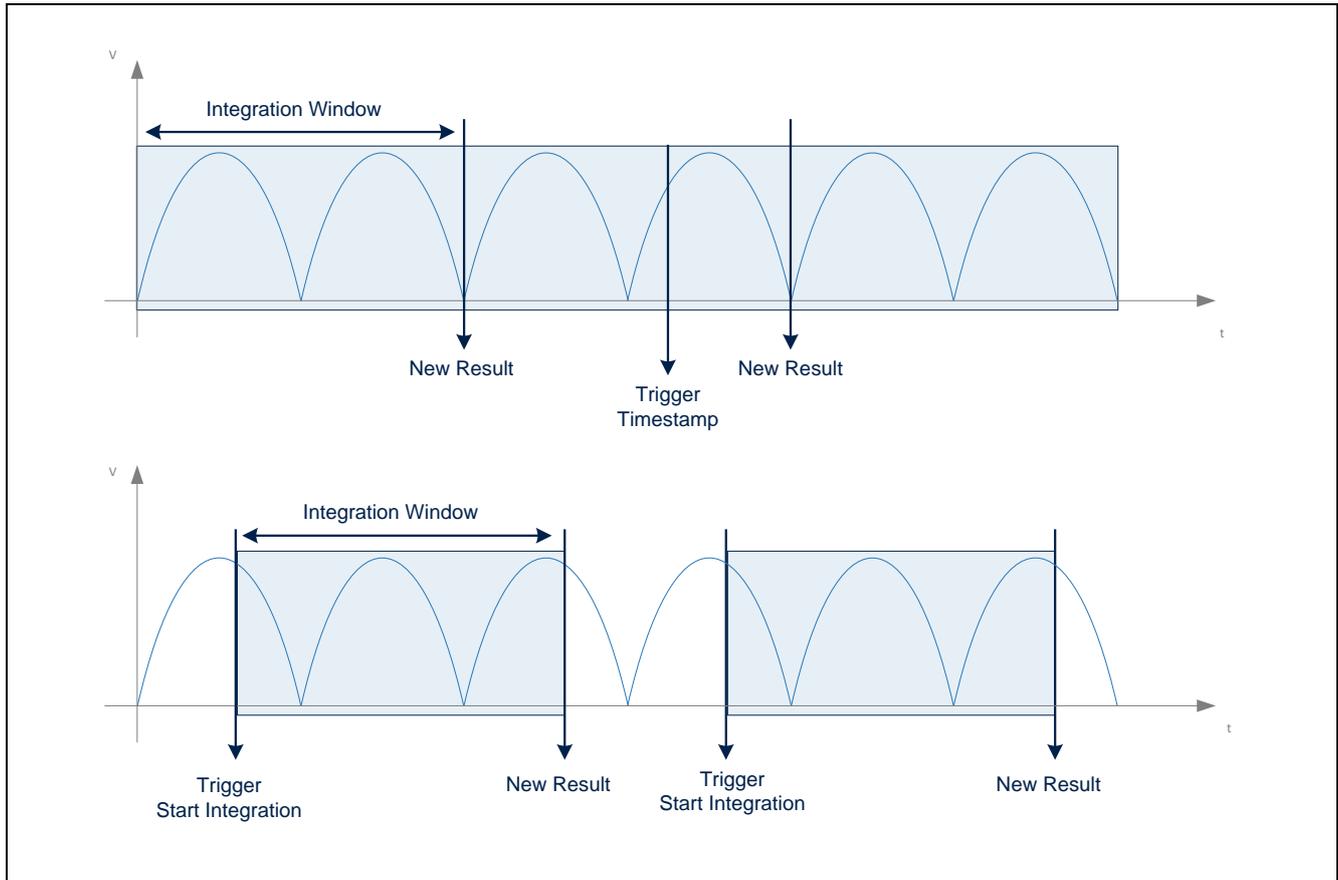
Figure 21   Resolver integration methods

7.4   Use case: Resolver analysis

A resolver is used as angle feedback system in a motor control application with Field Orientated Control (FOC). The PWM Frequency of 15 kHz is used. The current control has to be done each second cycle, therefore all 133µs a new angle has to be measured. To compensate the offset error and to provide the highest accuracy a full carrier period will be integrated. The resolver needs to be excited with 10 kHz sine wave with a tolerance of ±5 %. In this example a Modulator with 10 MHz sample frequency and 16 bit resolution is used.

To synchronize the carrier generator, delta sigma modulator and demodulator, the carrier signal and the clock for the delta sigma modulator are generated by the XMC. The carrier frequency is generated from the XMC clock. A divider of 12 and 1024 steps per period will lead to a carrier frequency of 9.765 kHz. With decimation factor of 16 and a modulator frequency of 10MHz exactly 64 main filter results fit in one carrier period. This synchrony is mandatory to avoid chitter. As the requirement is to integrate over one complete period the integration factor is also 64. Due to the modulator and to achieve high resolution, a CIC[3] filter is used. This main filter and integrator configuration leads to a filter response time of 4.8 µs, a filter update time of 102.4 µs and a resolution of 13 bit.

For the integration the triggered method with fixed integration window is used. The start trigger occurs with a period of 133 µs. Based on the filter response time of 4.8 µs 3 main filter results will be discarded and 64 integrated. This means the trigger has to occurs 107.2 µs before the phase current measurement for the FOC ends. This guarantees a synchronization between current and angle measurement.

infineon

7.4.1 XMC Lib configuration

Main filter configuration

The clock is provided by the XMC. The bit stream frequency of the DSM is the half the received clock frequency. Therefore the DSD provides a clock of 20 MHz. In this example the XMC4400 peripheral clock frequency is 120 MHz so the clock_divider is XMC_DSD_CH_CLK_DIV_6. And the clock_source is XMC_DSD_CH_CLOCK_SOURCE_INTERN. As data_source source A is selected. The decimation factor is 16 and the start value is also 16. The filter Type is set to CIC³ and no offset is added. The result event is enabled to read the angle when a new result is ready. The DSM clock is set at double of the rate of the data clock and the strobe XMC_DSD_CH_STROBE_DOUBLE_CLOCK_FALL is selected.

```
XMC_DSD_CH_FILTER_CONFIG_t filter_1 = {
    .clock_divider = XMC_DSD_CH_CLK_DIV_6,
    .clock_source =  XMC_DSD_CH_CLOCK_SOURCE_INTERN,
    .data_source = XMC_DSD_CH_DATA_SOURCE_A_DIRECT,
    .decimation_factor = 16U,
    .filter_start_value = 16U,
    .filter_type = XMC_DSD_CH_FILTER_TYPE_CIC3,
    .offset = 0U,
    .result_event = XMC_DSD_CH_RESULT_EVENT_ENABLE,
    .strobe = XMC_DSD_CH_STROBE_DOUBLE_CLOCK_FALL,
};
```

Integrator configuration

The relation between carrier generator and main filter is configured so that 64 main filter results occur in one carrier period. One complete period has to be integrated, therefore counted_values are set to 64. When a trigger occurs the main filter and integrator are reset. To blank the response time 3 values are discarded. There will be only one measurement per trigger so integration_loop is 1. As trigger source A is selected. The start condition is a rising trigger and it stops after the number of loops, which is 1.

```
XMC_DSD_CH_INTEGRATOR_CONFIG_t integ_1 ={
    .counted_values = 64,
    .discarded_values = 3,
    .integration_loop = 1,
    .integrator_trigger = XMC_DSD_CH_TRIGGER_SOURCE_A,
    .start_condition = XMC_DSD_CH_INTEGRATOR_START_TRIGGER_RISE,
    .stop_condition = XMC_DSD_CH_INTEGRATOR_STOP_END_OF_LOOPS,
};
```

Rectifier configuration

The rectification needs a sign source. The on chip carrier generator is used, so the sign_source is set to XMC_DSD_CH_SIGN_SOURCE_ON_CHIP_GENERATOR. The relation between carrier generator and main filter is configured so that 64 main filter results occur in one carrier period. Therefore the half_cycle is set to 64/2 = 32. The delay is specific to the hardware setup. In this example a delay of 3 was measured.

```
XMC_DSD_CH_RECTIFY_CONFIG_t rect_1 = {
     .sign_source = XMC_DSD_CH_SIGN_SOURCE_ON_CHIP_GENERATOR,
     .delay = 3,
     .half_cycle = 32,
};
```

Carrier Generator

In this example the XMC4400 peripheral clock frequency is 120 MHz and target frequency is 10 kHz. Dividing the peripheral clock frequency by 12288 gives a result of 9,7kHz, so the DSD is configured to XMC_DSD_GENERATOR_CLKDIV_12288. The required wave form is a sine wave, so the mode is set as XMC_DSD_GENERATOR_MODE_SINE. For a better sine shape the bit reverse mode is selected. The polarity is set to normal.

```
XMC_DSD_GENERATOR_CONFIG_t generator_1 = {
     .frequency = XMC_DSD_GENERATOR_CLKDIV_12288,
     .mode = XMC_DSD_GENERATOR_MODE_SINE,
     .bit_reverse = true,
     .inverted_polarity = false,
};
```

7.4.2   Initialization

The DSD_CH0 is configured with the filter_1 configuration from the previous above. The initialization sequence is important. Make sure that the DSD is enabled before the Main Filter is initialized. The Timestamp should be initialized after the filter initialization.

```
     XMC_DSD_Enable(DSD);
     XMC_DSD_EnableClock(DSD);

     XMC_DSD_CH_MainFilter_Init(DSD_CH0,&filter_1);
     XMC_DSD_CH_MainFilter_Init(DSD_CH2,&filter_1);

     XMC_DSD_CH_Timestamp_Init(DSD_CH0,&timestamp_1);
     XMC_DSD_CH_Timestamp_Init(DSD_CH2,&timestamp_1);
```

After initialization the module can be started. If two filters are used simultaneously both should be started with one function call.

```
XMC_DSD_Start(DSD,XMC_DSD_CH_ID_0| XMC_DSD_CH_ID_2);
```

7.4.3 Function implementation

Both DSD channels create an event when there is a new result. For readout the channel 0 interrupt is enabled.

```
NVIC_EnableIRQ(DSD0_M_0_IRQn);
```

In the interrupt service routine (ISR) the results can be stored.

```
#define result_ISR          DSD0_0_IRQHandler
void result_ISR(void)
{
     XMC_DSD_CH_GetResult(DSD_CH0,&result0);
     XMC_DSD_CH_GetResult(DSD_CH2,&result2);
}
```

After the results for sine and cosine are stored, the angle can be calculated. This can be done using the arctan() function. Various arctan() calculations are available with different execution times and resolutions. In a FOC motor control application the Cartesian to polar transformation function also provides an arctan() calculation. For example, the Infineon MOTOR_LIB_Car2Pol() available in the motor_lib.c when using DAVE™ v4 Apps.

8 Revision History

Current Version is V1.0, 2015-07

Page or Reference	Description of change
V1.0, 2015-07	
	Initial Version

www.infineon.com