

# X20DS1319

## 1 General information

### 1.1 Other applicable documents

For additional and supplementary information, see the following documents.

#### Other applicable documents

Document name	Title
MAX20	<a href="#">X20 system user's manual</a>
MAEMV	<a href="#">Installation / EMC guide</a>

### 1.2 Order data


Order number	Short description	Figure
	<b>Digital signal processing and preparation</b>	
X20DS1319	X20 multifunction digital signal processor, 4 digital input channels, 4 digital channels configurable as inputs or outputs, 2 event counters 1 universal counter pair as AB counter or up/down counter, linear movement generator (A/B, direction/frequency) with max. 2 reference pulses, SSI absolute encoder, NetTime function	
	<b>Required accessories</b>	
	<b>Bus modules</b>	
X20BM11	X20 bus module, 24 VDC keyed, internal I/O power supply connected through	
X20BM15	X20 bus module, with node number switch, 24 VDC keyed, internal I/O power supply connected through	
	<b>Terminal blocks</b>	
X20TB12	X20 terminal block, 12-pin, 24 VDC keyed	

Table 1: X20DS1319 - Order data

### 1.3 Module description

This module is a multifunction digital signal processor module. It can be used extremely flexibly for a wide variety of tasks involving digital signal processing or digital signal generation. Two primary example applications include controlling stepper output stages with pulse and direction signals or using as encoder emulation. In this application, for example, frequency inverters or servo axes with the speed follow function can follow a real or virtual master axis.

Functions:

- [Direct I/O](#)
- [Oversampled I/O](#)
- [Edge detection](#)
- [Motion functions](#)
- [Synchronous Serial Interface \(SSI\)](#)
- [Counter](#)
- [NetTime timestamp](#)

**Direct I/O**

The module is equipped with 8 channels. "Direct I/O" makes it possible to use the physical I/Os like normal digital inputs and outputs.

**Oversampled I/O**

Identical to Direct I/O, only with the difference that the inputs or outputs can be read in or switched several times within one cycle. This allows higher frequency signals to be analyzed or output.

**Edge counter**

The module is equipped with edge counters to evaluate positive or negative edges.

**Motion functions**

Encoder emulation can be used to generate up/down counters (direction/frequency) and ABR encoder signals.

**SSI absolute encoder**

The module provides one SSI absolute encoders that is directly supported by the hardware.

Absolute encoders are linear or angular encoders used as position encoders on machine tools, in handling and automation technology, and on measuring and testing equipment. The absolute measured value is available without referencing immediately after switching on.

**Counter functions**

The module is equipped with 4 internal counters that can be configured in 3 different modes. The following configurations are possible:

- Edge counter
- Up/Down counter
- Incremental encoders

**NetTime timestamp**

An additional major feature is the module's integrated timestamp function. It allows counter ramps curves to be generated virtually independently of bus cycle times during encoder emulation, for example. Only the target counter value and moment when it should be reached are transferred. The module automatically generates the corresponding counter values at the appropriate time, precisely in microsecond resolution and independently of the bus clock.

## 2 Technical description

### 2.1 Technical data

Order number	X20DS1319
Short description	
I/O module	4 digital input channels, 4 digital channels configurable as inputs or outputs, 1 universal counter pair (2 event counters, AB counter or up/down counter), linear movement generator (A/B, direction/frequency) with up to two reference pulses, SSI absolute encoder, relative or absolute moments of input edges with $\mu\text{s}$ resolution, time-triggered I/O, I/O oversampling
General information	
B&R ID code	0x2547
Status indicators	I/O function per channel, operating state, module status
Diagnostics	
Module run/error	Yes, using LED status indicator and software
Outputs	Yes, using LED status indicator
Power consumption	
Bus	0.01 W
Internal I/O	1.5 W
Additional power dissipation caused by actuators (resistive) [W]	-
Type of signal lines	Shielded lines must be used for all signal lines.
Certifications	
CE	Yes
UKCA	Yes
ATEX	Zone 2, II 3G Ex nA nC IIA T5 Gc IP20, Ta (see X20 user's manual) FTZÚ 09 ATEX 0083X
UL	cULus E115267 Industrial control equipment
HazLoc	cCSAus 244665 Process control equipment for hazardous locations Class I, Division 2, Groups ABCD, T5
DNV	Temperature: <b>B</b> (0 to 55°C) Humidity: <b>B</b> (up to 100%) Vibration: <b>B</b> (4 g) EMC: <b>B</b> (bridge and open deck)
LR	ENV1
KR	Yes
ABS	Yes
BV	<b>EC33B</b> Temperature: 5 - 55°C Vibration: 4 g EMC: Bridge and open deck
EAC	Yes
KC	Yes
Linear motion generator	
Quantity	1
Encoder outputs	24 V, asymmetrical (A/B, direction/frequency)
Counter size	16/32-bit
Digital inputs	
Quantity	4 + 4, configuration as input or output using software
Nominal voltage	24 VDC
Input voltage	24 VDC -15% / +20%
Input current at 24 VDC	Approx. 1.3 mA
Input circuit	Sink
Input filter	
Hardware	$\leq 2 \mu\text{s}$
Software	-
Input resistance	18.4 k $\Omega$
Additional functions	SSI absolute encoder, universal counter pair, latch function for universal counter pair
Input frequency	100 kHz
Switching threshold	
Low	<5 VDC
High	>15 VDC
Overload characteristics of encoder power supply	Short-circuit proof, overload-proof
Insulation voltage between channel and bus	500 V <sub>eff</sub>
SSI absolute encoder	
Quantity	1
Counter size	Up to 32-bit depending on encoder
Max. transfer rate	125 kbit/s
Encoder power supply	Module-internal, max. 600 mA
Nominal voltage	24 V, asymmetrical


Table 2: X20DS1319 - Technical data

Order number	X20DS1319
<b>Universal counter pair</b>	
Quantity	1
Operating modes	2x event counter, up/down counter, AB counter
Encoder inputs	24 V, asymmetrical
Counter size	16/32-bit
Input frequency	Max. 100 kHz
Evaluation	
AB counter	4x
Event counters	2x
Up/Down counter	2x
Signal form	Square wave pulse
Encoder power supply	Module-internal, max. 600 mA
<b>Digital outputs</b>	
Quantity	Up to 4, configuration as input or output using software
Variant	Push / Pull / Push-Pull
Nominal voltage	24 VDC
Switching voltage	24 VDC -15% / +20%
Nominal output current	0.1 A
Total nominal current	0.4 A
Output circuit	Sink and/or source
Output protection	Thermal shutdown in the event of overcurrent or short circuit, integrated protection for switching inductive loads
Diagnostic status	Output monitoring
Leakage current when the output is switched off	Max. 25 µA
Residual voltage	<0.9 V at 0.1 A nominal current
Peak short-circuit current	<10 A
Switch-on in the event of overload shutdown or short-circuit shutdown	Approx. 10 ms (depends on the module temperature)
Switching delay	
0 → 1	<2 µs
1 → 0	<2 µs
Switching frequency	
Resistive load	Max. 125 kHz
Inductive load	See section "Switching inductive loads".
Braking voltage when switching off inductive loads	Switching voltage + 0.6 VDC
Additional functions	Clock for SSI absolute encoder, linear movement generator
<b>Electrical properties</b>	
Electrical isolation	Channel isolated from bus Channel not isolated from channel
<b>Operating conditions</b>	
Mounting orientation	
Horizontal	Yes
Vertical	Yes
Installation elevation above sea level	
0 to 2000 m	No limitation
>2000 m	Reduction of ambient temperature by 0.5°C per 100 m
Degree of protection per EN 60529	IP20
<b>Ambient conditions</b>	
Temperature	
Operation	
Horizontal mounting orientation	-25 to 60°C
Vertical mounting orientation	-25 to 50°C
Derating	-
Storage	-40 to 85°C
Transport	-40 to 85°C
Relative humidity	
Operation	5 to 95%, non-condensing
Storage	5 to 95%, non-condensing
Transport	5 to 95%, non-condensing
<b>Mechanical properties</b>	
Note	Order 1x terminal block X20TB12 separately. Order 1x bus module X20BM11 separately.
Pitch	12.5 <sup>+0.2</sup> mm

Table 2: X20DS1319 - Technical data

## 2.2 LED status indicators

For a description of the various operating modes, see section "Additional information - Diagnostic LEDs" in the X20 system user's manual.

Figure	LED	Color	Status	Description
	r	Green	Off	No power to module
			Single flash	Mode RESET
			Double flash	Mode BOOT (during firmware update) <sup>1)</sup>
			Blinking	Mode PREOPERATIONAL
			On	Mode RUN
	e	Red	Off	Module not supplied with power or everything OK
			Single flash	I/O error. Possible causes: • SSI error <sup>2)</sup>
			Double flash	System error. Possible causes: • Motion function error <sup>3)</sup> • I/O oversampling error <sup>4)</sup> • Edge detection error <sup>4)</sup>
			Triple flash	I/O error and system error occur together
			On	Error or reset state
	1 - 8	Green		State of the corresponding digital signal

1) Depending on the configuration, a firmware update can take up to several minutes.

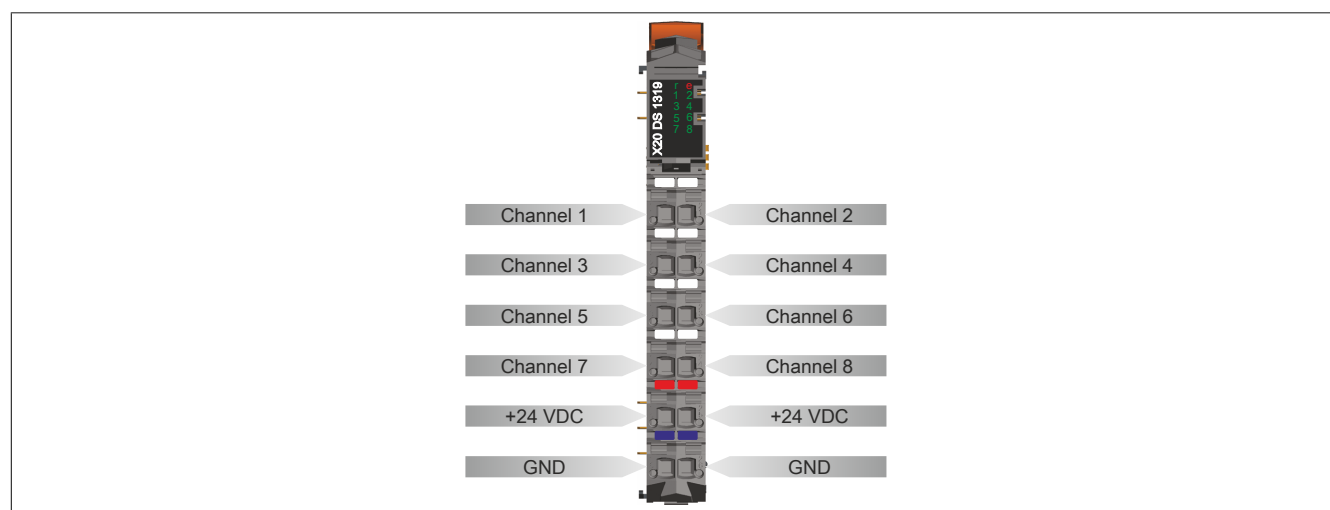
2) See register "Error state - SSI" on page 50 for the exact error description.

3) See register "Error state - Motion functions" on page 51 for the exact error description.

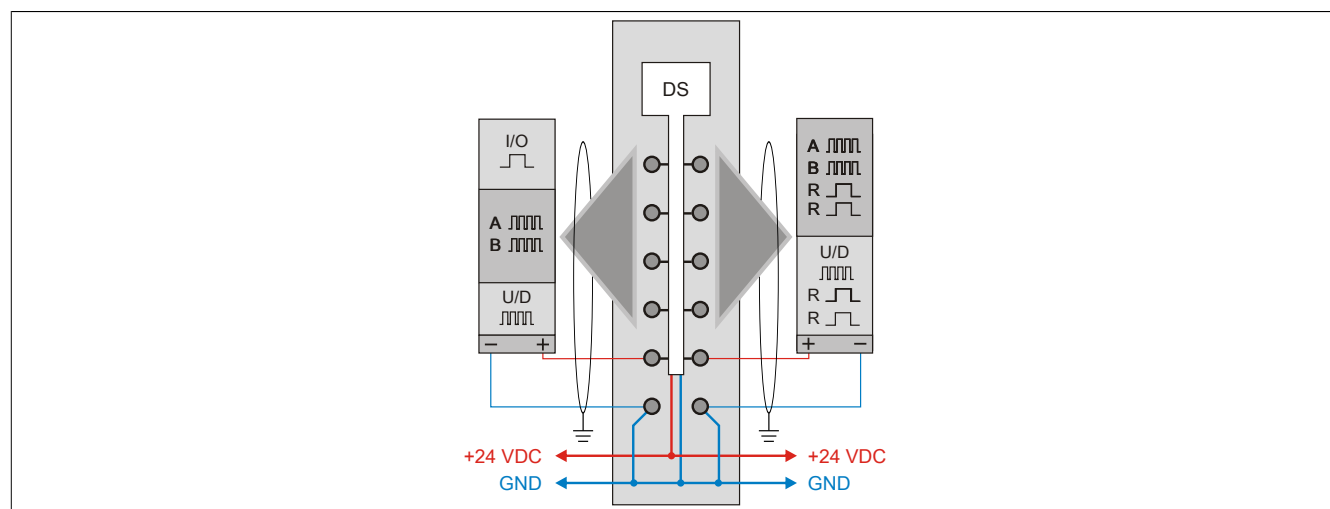
4) See register "Error state - Output data and edge detection" on page 50 for the exact error description.

## 2.3 Pinout

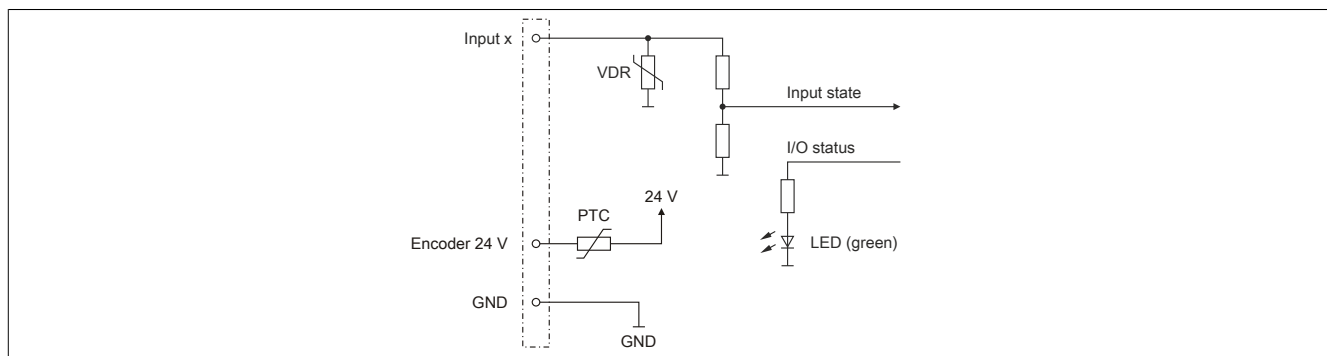
Shielded cables must be used for all signal lines.



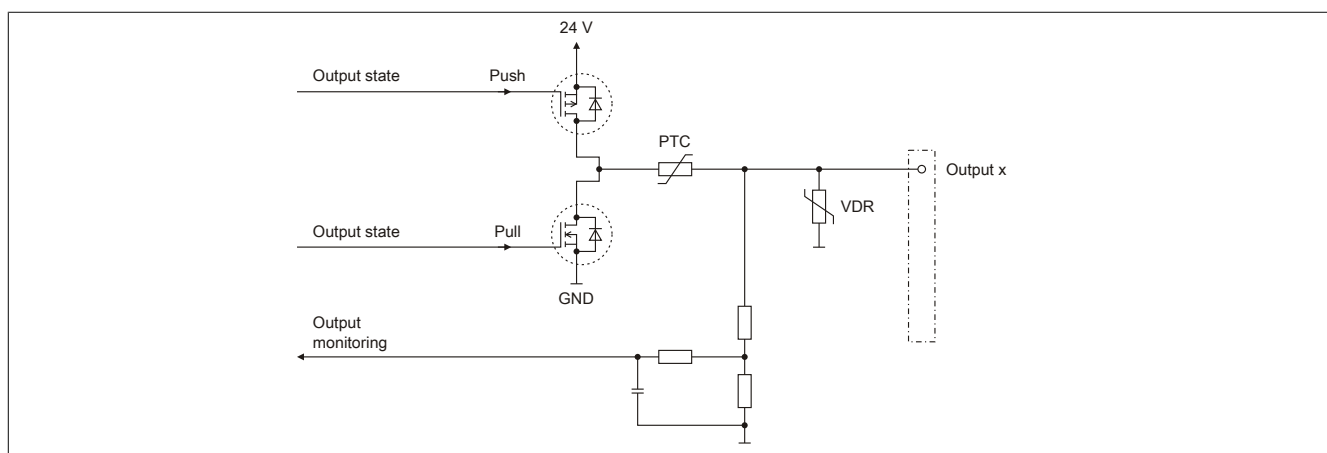
## 2.4 Connection example



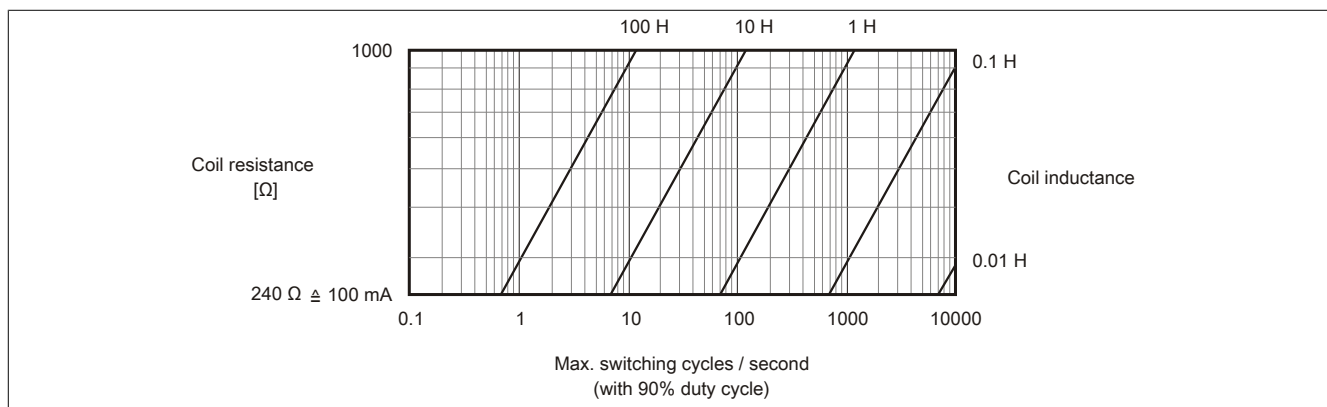
## 2.5 Input circuit diagram



## 2.6 Output circuit diagram



## 2.7 Switching inductive loads



## 3 Function description

### 3.1 Connection options

This module is equipped with 8 channels that must be connected and configured according to the desired functionality.

#### Digital input/output

Channel	Function
1	Input
2	Input
3	Input / Output
4	Input / Output
5	Input
6	Input
7	Input / Output
8	Input / Output

#### Wiring of the SSI absolute encoder

Channel	Function
5 (input)	Data
7 (output)	Clock

#### Wiring the linear motion generator

Channel	Up/Down	AB
3 (output)	Direction	A
4 (output)	Frequency	B
7 (output)	Reference 1	
8 (output)	Reference 2	

#### Wiring of the universal counter pair

Channel	Edge counter	Up/Down counter	Incremental
1 (input)	Input 1	Direction	A
2 (input)	Input 2	Frequency	B
5 (input)	Latch input 1 (R)		
6 (input)	Latch input 2 (E)		

### Information:

The register is described in ["Configuring the I/O channels" on page 30](#).

### 3.2 System functions

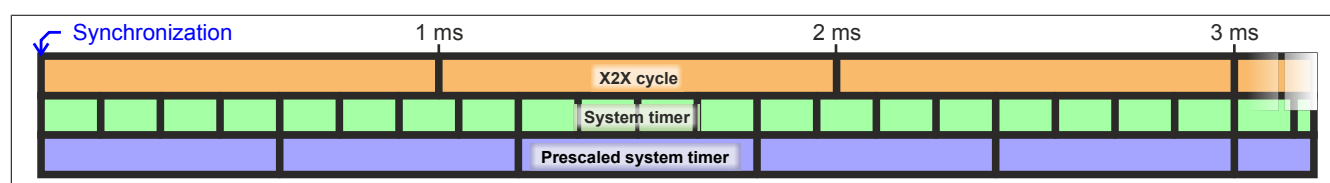
#### 3.2.1 System timer

The module's individual functions all depend on a system timer. This internal "system cycle time" can be set from 25 to 255  $\mu$ s. The functions can also be run using a configurable "prescaled system timer" to minimize the load on the module, thereby making it possible to use the shortest possible X2X cycle time.

The cycle of the [prescaled system timer](#) (and system timer) is referenced with the X2X Link as soon as the module has been started up and the X2X Link has been initialized. Since the system timer and the module's internal [NetTime](#) use the same timer, the two run synchronously from that point on. If the X2X cycle time is not a multiple of the system cycle time, there will be an offset; this can be calculated, however.

The following values apply to the following example:

X2X cycle            1 ms  
 System timer        150  $\mu$ s  
 Prescaled system timer    4



## Cycle prescaler

The "prescaled system timer" can be used as an alternative time source for the individual functions. This is useful if a function requires a very short system cycle. To reduce the module load in such a situation, other functions can be processed in a slower cycle.

### 3.2.2 Reference cycle

A reference cycle must be defined for data acquisition or transfer for oversampled I/Os, edge detection, SSI and counter functions. At the beginning of each cycle, the acquired data is transferred as input or output data according to the respective function.

The following options are available as sources:

Source	Information
System timer	The value set in register "CfO_SystemCycleTime" is used as the reference cycle.
Prescaled system timer	The value set in register "CfO_SystemCyclePrescaler" is used as the reference cycle.
AOAI <sup>1)</sup>	The sample cycle is referenced with the AOAI interrupt of the X2X cycle.
SOSI <sup>1)</sup>	The sample cycle is referenced with the SOSI interrupt of the X2X cycle.

1) Not usable for the poll cycle configuration.

## Input data

- **Oversampled I/O**

During each sample cycle, one bit from the output control buffers of the oversampled I/O channels is output to the configured physical output, and the status of the configured inputs is read into one bit of the respective input status buffer.

- **Edge detection**

When using edge detection mode "Polling", the polling cycle must be  $\leq 255 \mu\text{s}$ . If the configured cycle  $> 255 \mu\text{s}$ , EdgeDetectError occurs.

- **Counters**

The maximum counting frequency depends on the selected reference cycle. The module can process a maximum of 200 increments (edges) within one reference cycle.

## Output data

- **Oversampled I/O**

- The input data is referenced at the moment of the reference cycle. The referenced data is then copied to the "oversample input sample register" on page 36 at the moment of SI frame generation, taking into account the oversample input window.
- With relative addressing of the output control buffer, the new sample data is copied to an address relative to the output control buffer address current to the "reference cycle".
- The reference cycle is also used to reference the sample cycle and thus the output data production and input data acquisition (e.g. to the X2X cycle).

- **SSI**

SSI transfer is started at the update cycle. The clock sequence is generated on the SSI clock output. The first edge of the clock signal triggers the monoflop in the encoder and latches the current position. At the same time, the current NetTime is also logged in register "SSITimeValid" on page 46. As soon as all bits have been transferred via the SSI, the position is passed on with the next "SIframeGenCycle" via the X2X Link. Error SSICycleTimeViolation is reported if the SSI transfer is not completed within the SSI update cycle (e.g. system timer as update cycle). The SSI transfer is still fully completed and then started again with the next update cycle.



### 3.2.3 Transfer mode

There are 2 modes available for the time the input or output data is transferred:

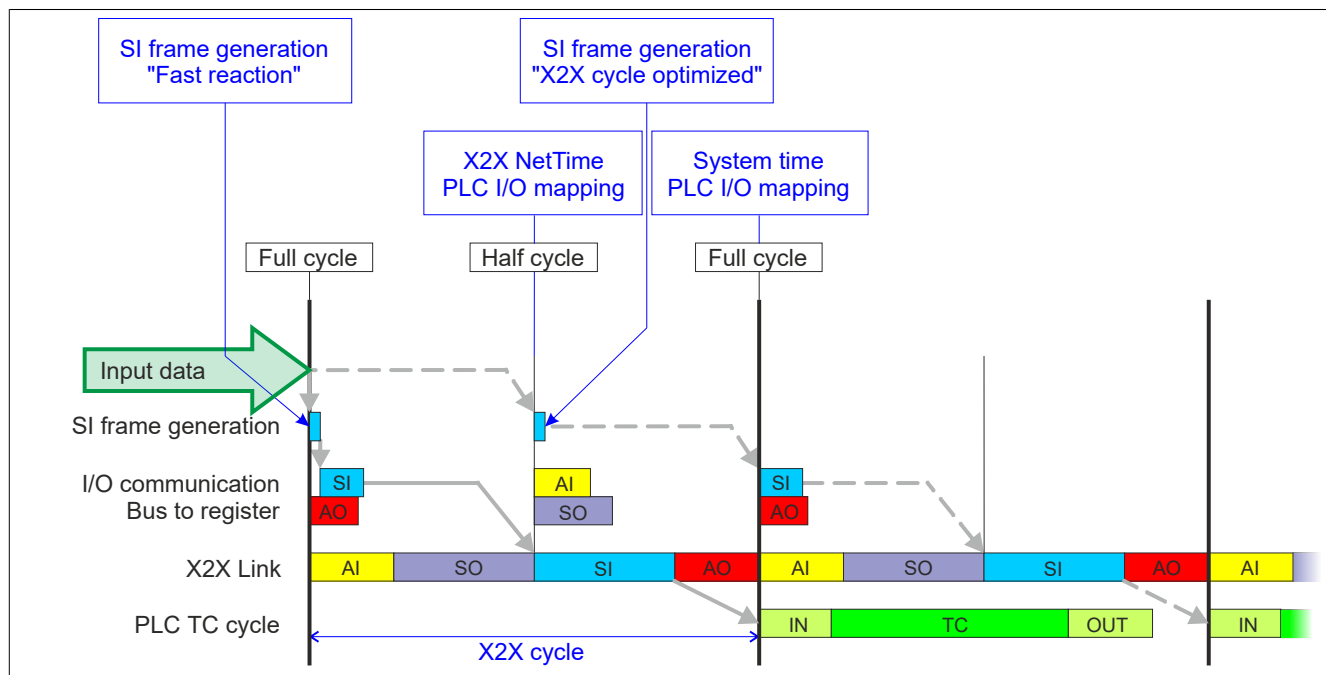
- Fast reaction
- X2X cycle optimized

The set mode has a decisive influence on the timing characteristics of the input or output data.

#### 3.2.3.1 Input data

When using mode "Fast reaction", the input data is available in the controller one X2X cycle earlier. However, this setting also has a negative effect on the minimum X2X cycle time.

The following graphic shows the transfer of the input data according to the selected configuration.



#### 3.2.3.2 Oversampled I/O output data

When using mode "Fast reaction", the output data is copied to the output control buffer immediately after it has been received. However, it is not possible to determine exactly when the data is copied to the output control buffer. The copy cycles will experience a certain degree of jitter depending on the module load. However, this only affects the moment of the internal copy procedures and therefore the moment of the earliest possible output sample. This will not affect the quality of the output signal. "Output copy cycle = Fast reaction" also has a negative effect on the minimum X2X cycle time.

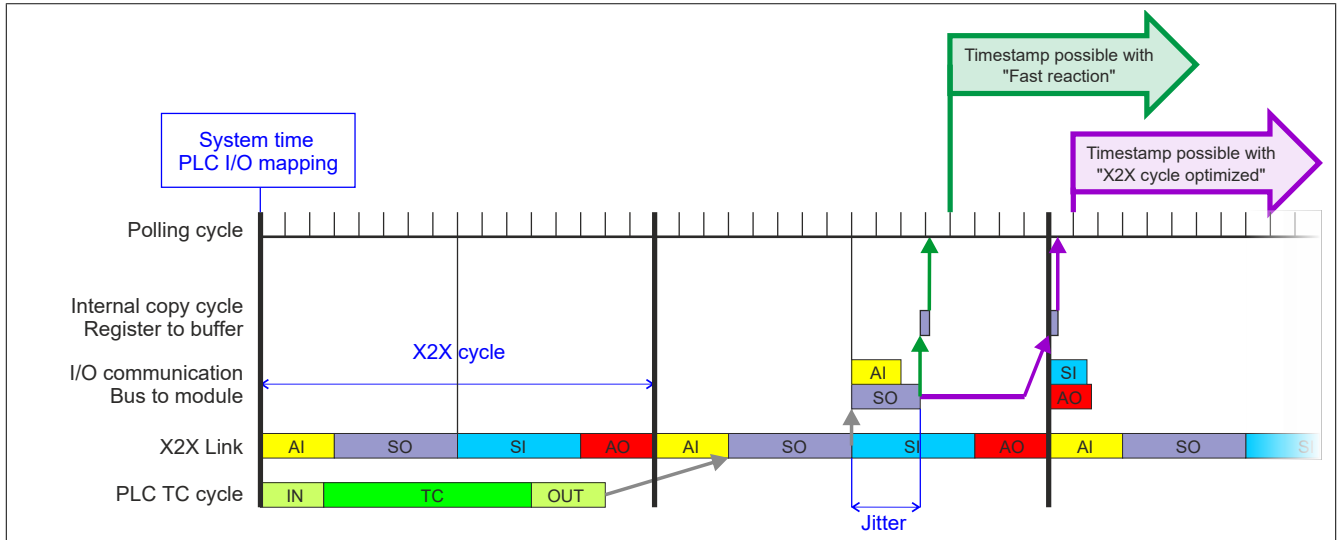
When using mode "X2X cycle optimized", the output data is copied to the output control buffer with the AOAI interrupt of the X2X cycle. It is important to note that due to the internal copy cycle to the output control buffer, however, output of the sampled data cannot be started immediately to the "output copy cycle".

### 3.2.3.3 Edge generation

The data is applied at different times depending on the set mode:

- **X2X cycle optimized**  
The data is applied permanently between the ASYNC IN (AI) and ASYNC OUT (AO) periods.
- **Fast reaction (jitter)**  
The data is applied immediately after SYNC OUT (SO) processing.

Setting "Fast reaction" results in jitter because the copy cycle of the SYNC OUT data can take different amounts of time. However, this only affects the moment at which the internal copy cycle takes place and therefore possibly also the earliest possible timestamp. Timestamps that are set outside of this jitter range are not affected by this.



### 3.2.4 Synchronization jitter

Since the controller that specifies the X2X NetTime and the module have different clocks, the module-internal X2X NetTime must be synchronized with the NetTime of the controller. Due to this synchronization, the module's internal X2X NetTime is corrected by a maximum of  $1/8 \mu\text{s}$  per system cycle if necessary. This synchronization jitter becomes noticeable when using the NetTime with  $1/8 \mu\text{s}$  resolution (max.  $\pm 1/8 \mu\text{s}$ ).

If a 100% exact  $1/8 \mu\text{s}$  resolution without jitter is required, then the "localtime  $1/8 \mu\text{s}$ " must be used.

### 3.2.5 Use with Automation Studio

The module is supported via X2X Link and POWERLINK!

X2X Link supports a maximum of 28 bytes of synchronous cyclic data per module. To optimize use and avoid needless data transfer, data points can be adjusted as needed in Automation Studio, i.e. unnecessary data points can be disabled, and the bit width of data points can be set.

## 3.3 Direct I/O

Direct I/O makes it possible to use the physical I/Os like normal I/Os. Additionally, the application can only set or reset I/Os (e.g. an output channel is set by the edge generator and manually reset by the application).

### Information:

The registers are described in "[Direct I/O](#)" on page 31.

### 3.4 Oversampled I/O

With oversampled I/O, the inputs or outputs can be read in or switched several times within one cycle. This allows higher frequency signals to be analyzed and output than with direct I/O.

"Oversampled I/O" is based on input status buffers and output control buffers. The input data acquisition and output control occur in one sample cycle (one sample cycle corresponds to one bit in the buffer). The precise moment of an input buffer entry is indicated by its position in the buffer and the **NetTime** assigned to the buffer.

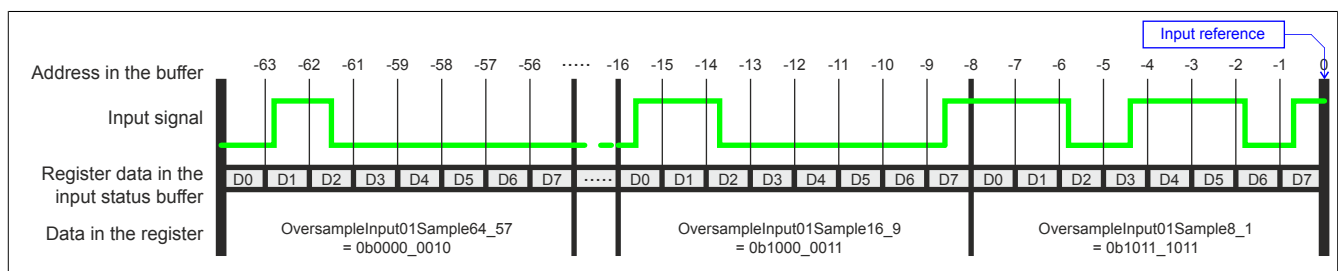
### 3.4.1 Input data

The data of the 4 oversample input status buffers is copied at the moment defined in the SI frame generation register. A maximum of 64 samples (8 bytes) per oversample I/O channel can be synchronously retrieved from the oversample input status buffer with each X2X cycle and stored in byte "OversampleInputSample".

The most recent input sample bit is stored in "OversampleInputSample8\_1" bit 7. The oldest input sample is stored in "OversampleInputSample64\_57" bit 0.

### Example

Input signal and resulting data in "OversampleInputSample":



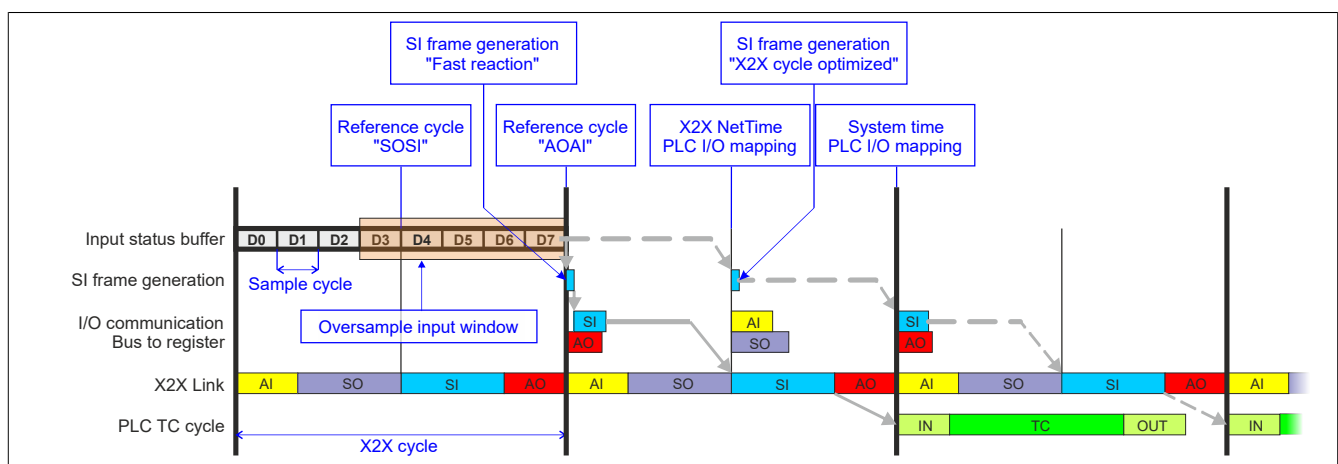
### Defining the reference time point

The reference time point is located chronologically before SI frame generation. If the reference time point (reference cycle) is within this window, then the referenced data from the input status buffer is copied to register "OversampleInput0NSample". If the reference time point is outside the "oversample input window", then the data that is most current at the moment of "SI frame generation" is copied from the input status buffer to register "OversampleInput0NSample".

This register is limited internally to the value from register "CfO\_OversampleInputBits".

### Information:

**As a result, the oversample input time and oversample input cycle are set either at the reference time point or at the moment of "SI frame generation".**



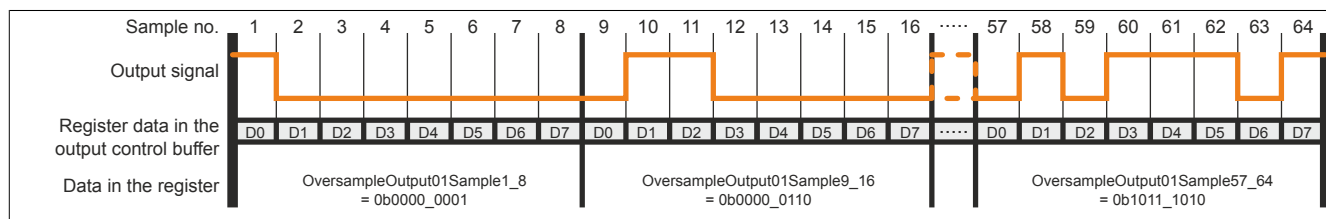
### 3.4.2 Output data

Up to 64 samples (8 bytes) for each oversample I/O channel can be synchronously transferred with a X2X cycle. This data is copied to the specified address (absolute or relative) in the output control buffer at the set output copy cycle. 1 bit of this data is then output during each "sample cycle" to the physical output that is assigned to the oversample I/O channel.

Bit 0 of "OversampleOutputSample1\_8" is copied to the output control buffer first, meaning that it is the first bit that is output. "OversampleOutputSample57\_64" bit 7 is the last bit to be output.

#### Example

Assignment of "OversampleOutputSample" register data to the output signal:



#### Output operation

When "Output control mode = Single", every output buffer entry is marked as invalid once it has been executed. This ensures that the outputs are not supplied with invalid data. In this mode, the application needs to ensure that the module is always supplied with valid data.

When using "Output control mode = Continuous" the contents of the buffer are output again if the module is not supplied with new oversample output data.

#### Cyclic output control

If cyclic output control is enabled, then all data in the output control buffer is marked invalid as soon as it is output ("Output control mode = Single"). OutputControlError is generated if the module is not supplied with new data in time; in this case, a bit already output in the buffer would be output again. In this type of error situation, the output takes on the "Output default state" configured in register "CfO\_OversampleConfigOutput".

If cyclic output control is disabled, then the data is output again if the output control buffer overflows ("Output control mode = Continuous").

#### Information:

All 256 bits of the output control buffer are always output.

#### Information:

The registers are described in ["Oversampled I/O" on page 32](#).

### 3.4.3 Addressing the output control buffer

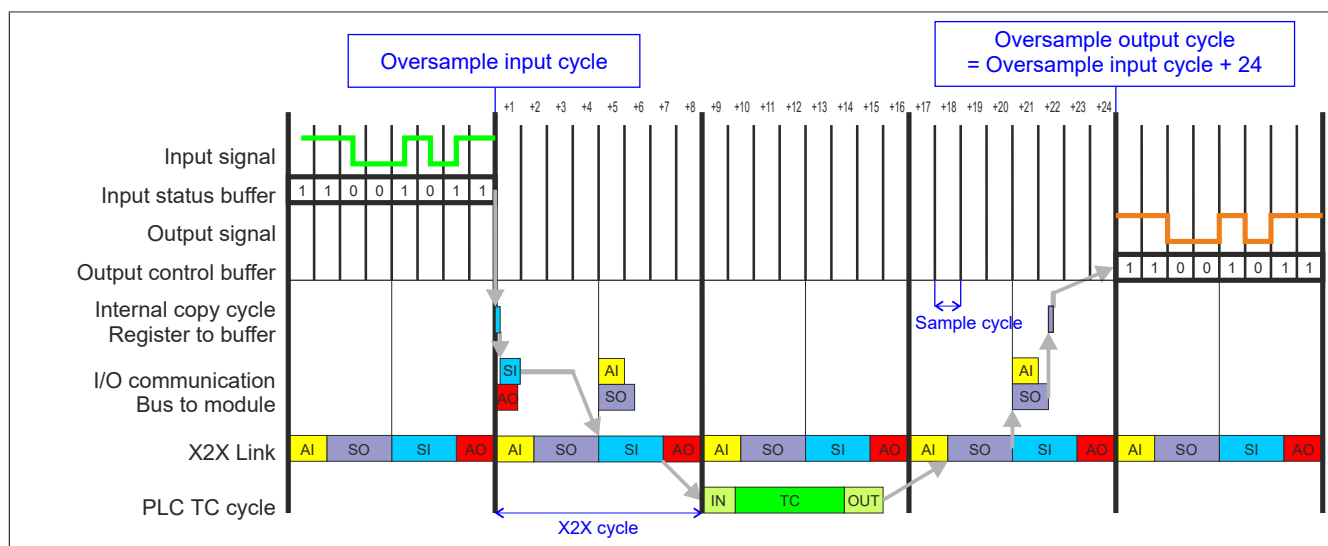
The module has one cyclic 256-bit output control buffer for each oversample channel. One bit is output from these buffers to the configured physical output channels in each "sample cycle". When new data is transferred to one of these buffers, the application must define where in the respective buffer the data should be written to. There are 2 possibilities available for this (absolute or relative "Output mode" in the Automation Studio I/O configuration).

#### 3.4.3.1 Absolute addressing of the output control buffer

With absolute addressing, in each cycle where "OversampleOutputValidate = True", in addition to the oversample output sample data (in the "OversampleOutput0NSample" on page 35 registers) an address must also be transferred in register "OversampleOutputCycle" on page 35. This address defines where in the output control buffer the new data should be copied. In order to calculate this address, the contents of register "OversampleInputCycle" on page 36, which contains the address of the most recently output data, and the transfer time to the module must be taken into account. To help avoid incorrect addressing of the output control buffer, the buffer section that is capable of being written to can be limited using register "OversampleOutputWindow" on page 33. This window will always be shifted relative to the current sample address. An "OutputCopyError" will be triggered if an attempt is made to write to an address that is outside of this window.

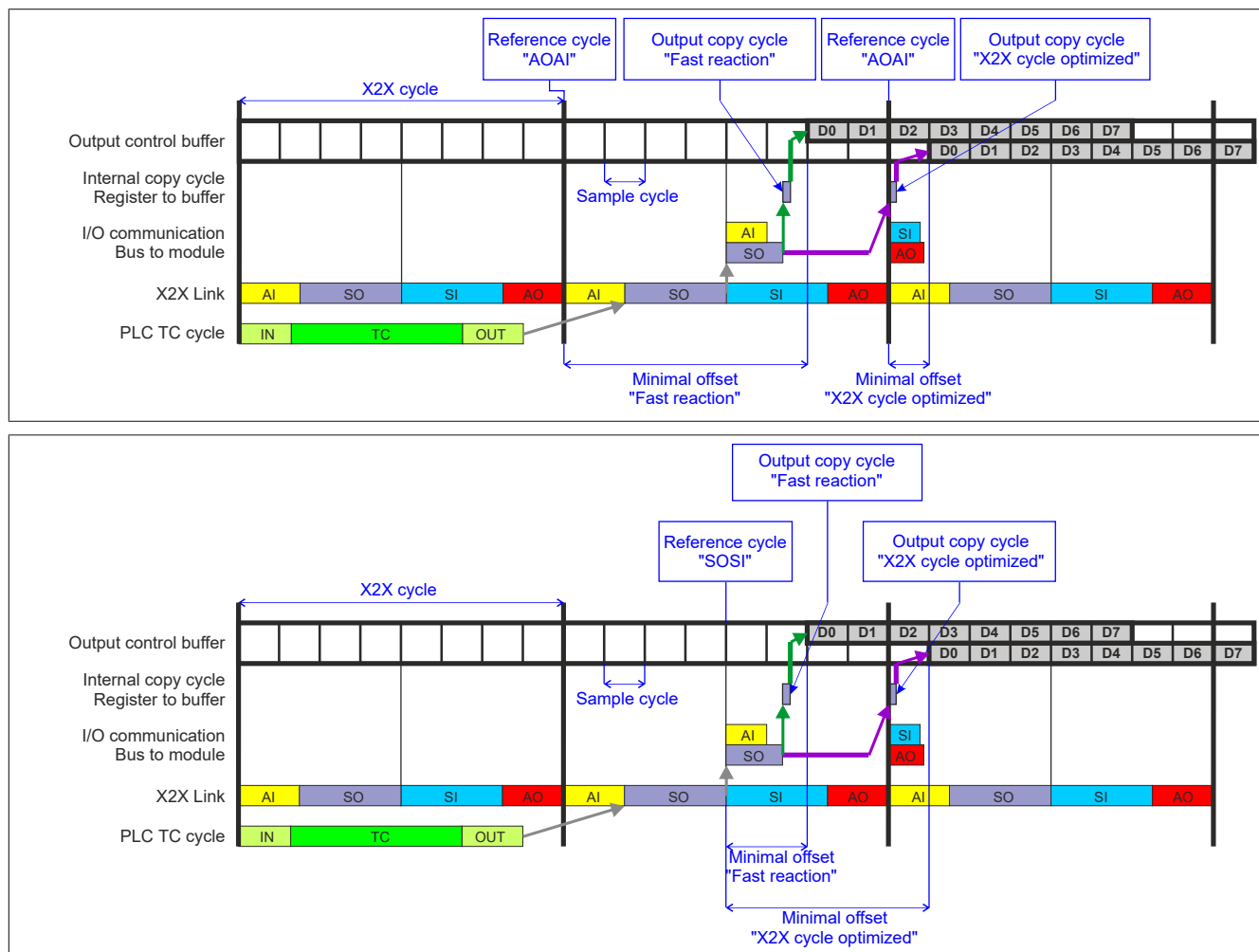
#### Example

Timing characteristics from the oversample input cycle to the oversample output cycle in absolute output mode ("SI frame generation = Fast reaction", "Output copy cycle = Fast reaction", 8 samples per X2X cycle):



### 3.4.3.2 Relative addressing of the output control buffer

When "OversampleOutputValidate = True", then the oversample output sample data is automatically copied to an address relative to the last referenced address at the set **output copy cycle** moment. Register "OversampleSampleOffset" on page 35 serves as the offset. The new data cannot start being output immediately at the **output copy cycle** moment because it takes time to copy the data from the registers to the buffer. This means that an offset of 0 is not allowed. The relative output control buffer address + offset must point to an address within the "oversample output window". The **oversample output window** is always offset relative to the current sample address. An **OutputCopyError** is triggered if an attempt is made to write to an address that is outside of this window.



### 3.5 Edge detection

The module's edge detection function allows edges to be measured with microsecond precision. The concept is based on a maximum of 4 units. One master and one slave edge can be configured for each unit.

#### Master edges

At the moment of each master edge, the [NetTime](#) of the master edge and the NetTime of a previous slave edge (if present) are logged. A "master counter" and a "slave counter" can always be used to determine how many edges have been detected since the last X2X cycle.

#### Slave edges

When a slave edge occurs, the current NetTime is always saved within the module. Memory is provided within the module that always stores the last 256 slave timestamps (even when a master edge occurs).

When a master edge occurs, the exact NetTime of any slave edge that may have occurred prior to the master edge and addressed by "Slave leading" is copied. If multiple slave edges occur before a master edge, then only the NetTime of the last edge that was not ignored by "Slave leading" is stored.

#### 3.5.1 Edge detection mode

2 modes can be selected for the moment when an interrupt for edge detection should be triggered:

- In mode "Event-triggered", the [NetTime](#) of each edge is recorded as an interrupt immediately when the edge occurs. However, an extremely large amount of interrupts within a short amount of time can prevent the module from being able to process any other operations in time!.
- In mode "Polling", only the NetTime of the first edge that occurs within a polling cycle is recorded. This ensures that the module is not overloaded by too many edges.

#### 3.5.2 Time base

When using a time base with 1/8  $\mu$ s resolution, it is important to note that the timestamps produced also resolve exactly to 1/8  $\mu$ s. Corresponding conversions must be made for calculating in connection with the system time of the controller or [X2X NetTime](#).

In addition, synchronization jitter also plays a role when using "Time base = Nettime resolution 1/8 usec" (see "[Synchronization jitter](#)" on page 11). This means that exactly identical input edges can cause slight differences in the results. If 100% exact 1/8  $\mu$ s resolution is required, then "Local resolution 1/8 usec" must be used.

#### Information:

The registers are described in "[Edge detection](#)" on page 36.



### 3.6 Motion functions

Encoder emulation can be used to generate up/down counters (direction/frequency) and ABR encoder signals. The following conditions must be met to achieve an exact match of the position of the module with the remote station:

- Up/Down counter: The remote station must evaluate both rising and falling edges.
- ABR encoder: The remote station must employ 4x evaluation.

The motion function can be operated in 2 different operating modes:

- ["Mode "Position control" on page 18](#)
- ["Mode "Speed control" on page 19](#)

#### Minimizing jitter

Depending on the configuration of the module, unfavorable system-related jitter times can result in every motion function. In order to increase the smooth running of the motor, however, the edge switching times and thus the unfavorable jitter can be minimized using register "CfO\_ResolPosition".

#### Information:

The registers are described in ["Motion functions" on page 39](#).

### 3.6.1 Mode "Position control"

Each time register "MovTargetTime" on page 44 changes, a new position setpoint is transferred from register "MovTargetPosition" on page 44 to the FIFO buffer. The time/position data in the FIFO buffer is then processed in such a manner that the positions are always reached at the moment of the respective timestamps. This means that the module internally ensures that the positions are reached by the set timestamps (number/frequency of the pulses is calculated automatically). The timestamps can be based on the X2X NetTime, the system time of the controller or register "MovTimeValid" on page 45. If timestamps are set in such a way that the required position change cannot be achieved within the time up to the timestamp (output frequency of the pulses would exceed "CfO\_SpeedLimit" on page 40), a **MovMaxFrequencyViolation** error is caused.

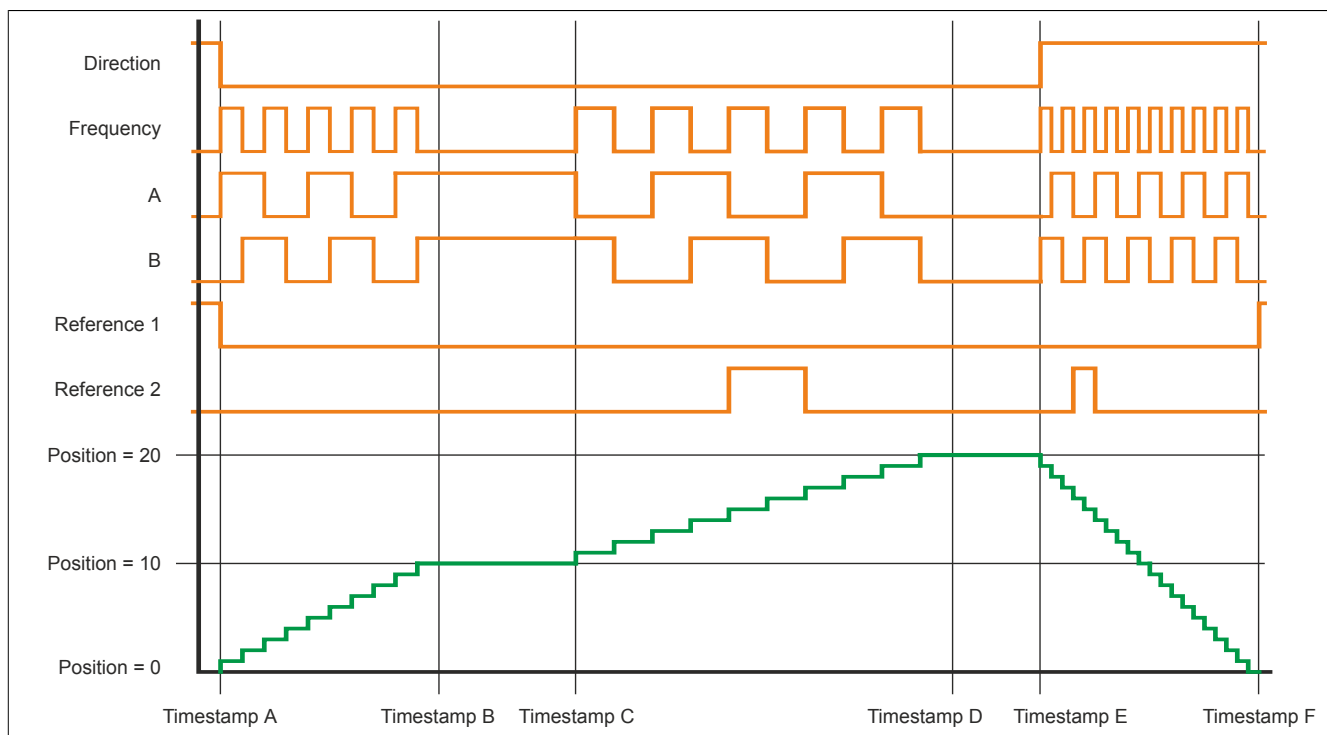
For additional information about NetTime and timestamps, see "NetTime Technology" on page 23.

Selected values for the example "Timing of movement":

Timestamp A = <b>MovTimeValid</b> + 40,000	Position for timestamp A = 0
Timestamp B = Timestamp A + 40,000	Position for timestamp B = 10
Timestamp C = Timestamp B + 25,000	Position for timestamp C = 10
Timestamp D = Timestamp C + 70,000	Position for timestamp D = 20
Timestamp E = Timestamp D + 15,000	Position for timestamp E = 20
Timestamp F = Timestamp E + 40,000	Position for timestamp F = 0

Configuration: Reference pulse 1 = Start position and margin, Start position = 0, Margin = 1

Configuration: Reference pulse 2 = Starting and end position, Starting position = 15, End position = 17



### 3.6.2 Mode "Speed control"

In mode "Speed control", the application only specifies the speed setpoint. The module returns the current position in register "MovPosition (32-bit)" on page 45.

The default setting (resolSpeed = 24) is designed in such a way that a value of 16,777,216 (0x01000000) in register "MovSpeed" on page 45 results in exactly one increment per "control period".

First, an internal speed value must be calculated:

$$v_{Intern} = v_{Out} * 2^{resolPos}$$

This results in the following correlation for a 32-bit speed specification (data format of the speed values = 32-bit):

$$MovSpeed = v_{Intern} * 2^{resolSpeed} * period$$

Atypically to other registers, when writing to register "MovSpeed (16-bit)", the 2 higher-order bytes of "MovSpeed (32-bit)" are written. This results in the following correlation for direct calculation with "MovSpeed (16-bit)".

$$MovSpeed = \frac{v_{Intern} * 2^{resolSpeed} * period}{2^{16}}$$

Variable	Description	Unit
MovSpeed	Value for register "MovSpeed" (16- or 32-bit)	
vIntern	Internally calculated speed value.	Inc/s
vOut	Desired output speed. Each edge (rising or falling) counts as an increment.	Inc/s
resolPos	Configured value of register "CfO_ResolPosition" on page 42	
resolSpeed	Configured value of register "CfO_ResolSpeed" on page 43	Bits
period	Configured value of register "CfO_SpeedCycleTime_32Bit" on page 42  <div style="border-left: 2px solid black; padding-left: 10px; margin-left: 20px;"> <b>Must be set in microseconds in Automation Studio. The calculation is performed in seconds, however.</b> </div>	s

### 3.6.3 Performing a movement in mode "Position control"

Several things must be kept in mind when operating the module in order to perform a movement without errors and avoid error messages.

#### Information:

**The specified time/position pairs are not "movement commands", but position data that is continuously processed by the module.**

- To allow the module to calculate movement pulses, the first time/position data pair (t, x) is interpreted as the home position. In this case, "t" represents the starting moment and "x" the current position. A movement is not yet performed.
- As long as bit 0 "MovEnable - For position control" on page 44 is set to "1", time/position data pairs must be continuously transmitted to the module. As soon as the last data pair has been processed and the module does not find another data pair in the FIFO buffer, error message MovFifoEmpty is sent (see "Error state - Motion functions" on page 51). In addition, error message MovTargetTimeViolation occurs because no "future moment" for another movement was found.
- To enable a standstill, the time/position data pairs must be specified with an unchanged position but future moments in time.
- Ending the movement with bit 0 = "0" "MovEnable - For position control" on page 44  
This only stops filling the FIFO buffer and subsequently suppresses error message MovFifoEmpty. All entries in the FIFO buffer are still processed. The last specified position is applied as the reference position. As soon as bit 0 = "1" again, all movements are performed relative to this position.
- Ending the movement with bit 7 = "1" "MovReset - Movement reset (immediate stop)" on page 44  
This stops the movement immediately. No more pulses are output. To restart the movement, bit 7 must be set to "0" and bit 0 must be set to "0" for a short time and then back to "1".

### 3.7 Synchronous Serial Interface (SSI)

Synchronous Serial Interface makes it possible to receive data from SSI absolute encoders.

2 cables are required for data exchange:

SSI clock: Generated by the module on output 7 (if configured)  
 SSI data: With each clock pulse, one data bit is transferred from the encoder to the module (input 5 can be used as the SSI input).

#### 3.7.1 Procedure for SSI transfer

On the first edge on the SSI clock, a monostable multivibrator is triggered in the encoder and the value currently present in parallel is latched to the shift register (the low level of the monostable multivibrator prevents the transfer of additional values to the shift register during data transfer).

On the next edge, the most significant bit is transferred to the module.

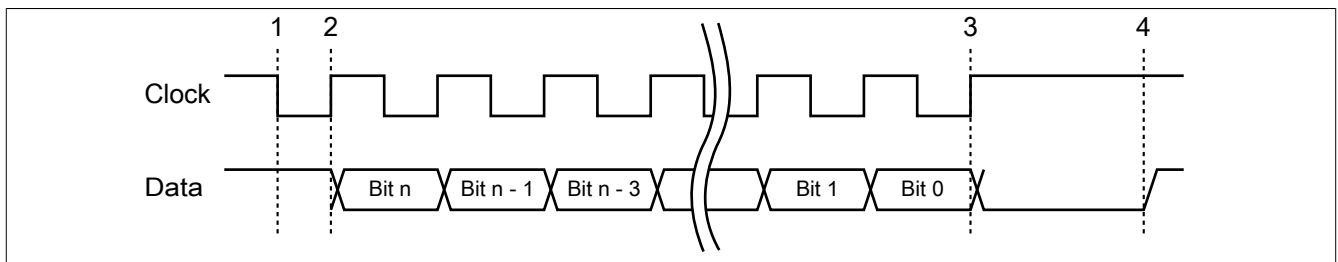
With each additional clock pulse, the next least significant bit is transferred. The clocks constantly retrigger the monostable multivibrator, which prevents its output from accepting new data.

If the set number of data bits has been received, the clock sequence is ended.

The monostable multivibrator is no longer triggered; after a time dependent on the encoder, the output of the monostable multivibrator accepts the output level again, thus allowing parallel data to be transferred to the encoder's shift register.

With monostable multivibrator testing, the data line is queried for the configured level before starting a new transfer. This makes it possible to ensure that the monostable multivibrator has really returned before a new transfer is started.

#### Transferring using Synchronous Serial Interface



#### Processing the measured value

- 1) Start bit ... The measured value is saved.
- 2) Output of the first data bit
- 3) All data bits are transferred; the monostable multivibrator time starts to run.
- 4) The monostable multivibrator returns to its initial state; a new transfer can be started.

#### Information:

The registers are described in "[Synchronous Serial Interface \(SSI\)](#)" on page 45.

### 3.8 Counter

Signals up to 100 kHz can be reliably acquired with the universal counter pair. Up to 4 latch inputs can be configured in all modes. If required, the enabled latch inputs are negated and linked to a latch condition with a logical AND operator. If the latch condition is met, the current counter value is saved to a separate register.

#### Inputs

Depending on the function model, the physical inputs are permanently configured for the counter.

Mode	Input 1	Input 2	Input 5	Input 6
Edge counter	Counter input for counter 1 Latch input 1	Counter input for counter 2 Latch input 2	- Latch input 3	- Latch input 4
Up/Down counter	Counting direction Latch input 1	Counter frequency Latch input 2	- Latch input 3	- Latch input 4
Incremental encoders	A Latch input 1	B Latch input 2	- Latch input 3	- Latch input 4

#### Counter modes

The universal counter pair can be used in 3 different modes.

Mode	Information
Edge counter	The two counters serve as edge counters in this mode. The counter input of counter 1 is permanently connected to input 1; the counter input of the second counter is permanently connected to input 2. Both rising and falling edges are counted.
Up/Down counter	The up/down counter works according to the direction/frequency principle. Input 1 sets the counting direction (LOW = positive, HIGH = negative); input 2 is used as the counter frequency input. Both rising and falling edges on the counter frequency input are counted.
Incremental encoder (AB counter)	When configured as an AB counter, input 1 is used as the A channel and input 2 is used as the B channel. All edges are evaluated (4x evaluation).

#### Latch function

As latch inputs, inputs 1, 2, 5 and 6 can each be queried for a HIGH or LOW level.

With "Latch mode = Continuous", the counters are latched once as soon as "LatchEnable = True" and the configured latch condition is met. When the latch condition is met again, the counter content is latched again (i.e. a latch event is triggered on every rising edge of the output of the AND operator of all latch inputs).

With "Latch mode = Single", the counters are latched once as soon as "LatchEnable = True" and the configured latch condition is met. When the latch condition is met again, the counter content is not automatically recopied. Another latch event can only be processed after "LatchEnable = False" and another "LatchEnable = True".

#### Information:

The registers are described in ["Counters" on page 47](#).

### 3.9 Error handling

If one of the functions detects an error, then an error bit is set in one of the error state registers. The application is now able to react to this and acknowledge the errors by setting a respective bit in the "Acknowledge error messages" registers. This resets the bit in the error status register. If the error source persists, then the error bit is set again as soon as the error is detected again (i.e. resetting is not possible).

Error acknowledgment has no effect on the functionality of the module. The module resumes processing, automatically if possible, as soon as the error source is eliminated.

If an error occurs (not a warning), this is indicated by the red "e" LED on the module (double flash). This signal is automatically acknowledged as soon as the error source has been eliminated.

#### Information:

The registers are described in ["Error handling" on page 50](#).

### 3.10 Timestamp

Timestamps are provided by the module for the following:

- NetTime of the last counter value change
- NetTime of the current counter value
- NetTime of the last position change
- NetTime of the current position
- NetTime of the target position
- NetTime when a slave edge occurs
- NetTime when a master edge occurs

The timestamp function is based on synchronized timers. If a timestamp event occurs, the module immediately saves the current NetTime. After the respective data is transferred to the controller, including this precise moment, the controller can then evaluate the data using its own NetTime (or system time), if necessary.

Conversely, the controller can predefine output events, apply a timestamp and transfer them to the module. The module then executes the predefined action at the precise moment defined by the CPU.

The resolution of the timestamp is up to 1/8  $\mu$ s in both directions.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

### 3.10.1 NetTime Technology

NetTime refers to the ability to precisely synchronize and transfer system times between individual components of the controller or network (controller, I/O modules, X2X Link, POWERLINK, etc.).

This allows the moment that events occur to be determined system-wide with microsecond precision. Upcoming events can also be executed precisely at a specified moment.



#### 3.10.1.1 Time information

Various time information is available in the controller or on the network:

- System time (on the PLC, Automation PC, etc.)
- X2X Link time (for each X2X Link network)
- POWERLINK time (for each POWERLINK network)
- Time data points of I/O modules

The NetTime is based on 32-bit counters, which are increased with microsecond resolution. The sign of the time information changes after 35 min, 47 s, 483 ms and 648  $\mu$ s; an overflow occurs after 71 min, 34 s, 967 ms and 296  $\mu$ s.

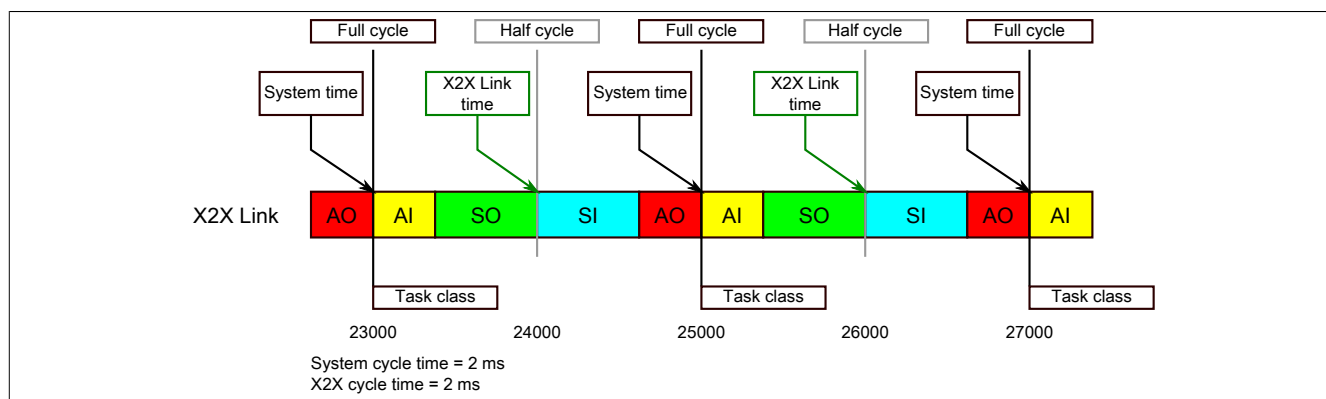
The initialization of the times is based on the system time during the startup of the X2X Link, the I/O modules or the POWERLINK interface.

Current time information in the application can also be determined via library AslOTime.

##### 3.10.1.1.1 Controller data points

The NetTime I/O data points of the controller are latched to each system clock and made available.

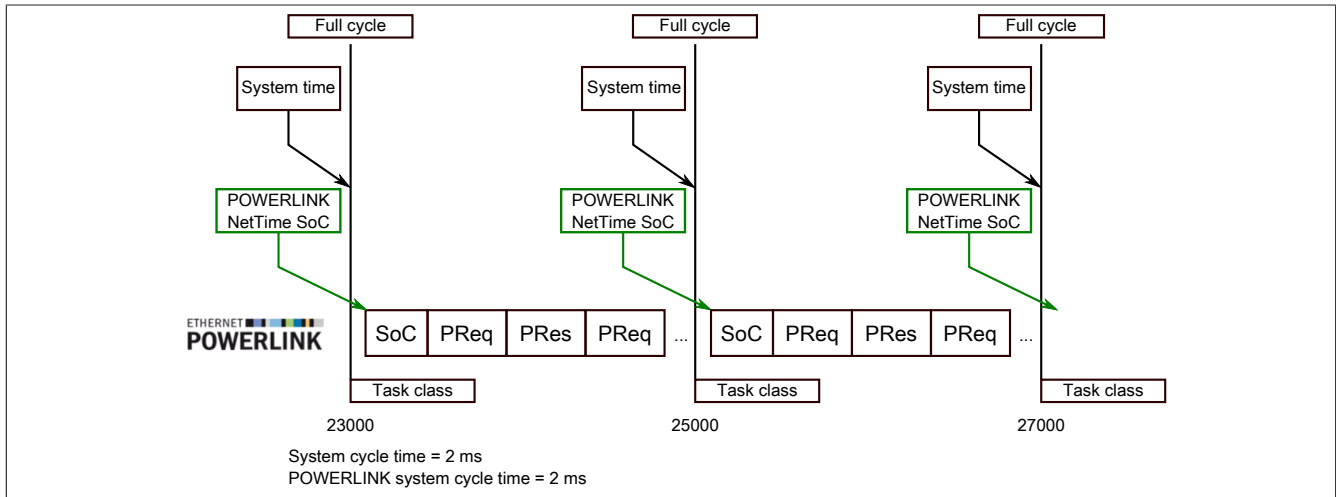
##### 3.10.1.1.2 X2X Link - Reference time point



The reference time point on the X2X Link network is always calculated at the half cycle of the X2X Link cycle. This results in a difference between the system time and the X2X Link reference time point when the reference time is read out.

In the example above, this results in a difference of 1 ms, i.e. if the system time and X2X Link reference time are compared at time 25000 in the task, then the system time returns the value 25000 and the X2X Link reference time returns the value 24000.

### 3.10.1.1.3 POWERLINK - Reference time point

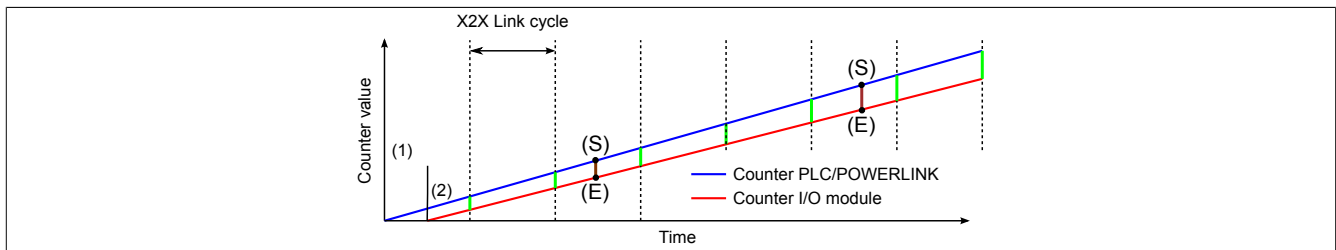


The POWERLINK reference time point is always calculated at the start of cycle (SoC) of the POWERLINK network. The SoC starts 20 μs after the system clock due to the system. This results in the following difference between the system time and the POWERLINK reference time:

POWERLINK reference time = System time - POWERLINK cycle time + 20 μs

In the example above, this means a difference of 1980 μs, i.e. if the system time and POWERLINK reference time are compared at time 25000 in the task, then the system time returns the value 25000 and the POWERLINK reference time returns the value 23020.

### 3.10.1.1.4 Synchronization of system time/POWERLINK time and I/O module



At startup, the internal counters for the controller/POWERLINK (1) and the I/O module (2) start at different times and increase the values with microsecond resolution.

At the beginning of each X2X Link cycle, the controller or POWERLINK network sends time information to the I/O module. The I/O module compares this time information with the module's internal time and forms a difference (green line) between the two times and stores it.

When a NetTime event (E) occurs, the internal module time is read out and corrected with the stored difference value (brown line). This means that the exact system moment (S) of an event can always be determined, even if the counters are not absolutely synchronous.

#### Note

The deviation from the clock signal is strongly exaggerated in the picture as a red line.



### 3.10.1.2 Timestamp functions

NetTime-capable modules provide various timestamp functions depending on the scope of functions. If a timestamp event occurs, the module immediately saves the current NetTime. After the respective data is transferred to the controller, including this precise moment, the controller can then evaluate the data using its own NetTime (or system time), if necessary.

#### 3.10.1.2.1 Time-based inputs

NetTime Technology can be used to determine the exact moment of a rising edge at an input. The rising and falling edges can also be detected and the duration between 2 events can be determined.

##### **Information:**

**The determined moment always lies in the past.**

#### 3.10.1.2.2 Time-based outputs

NetTime Technology can be used to specify the exact moment of a rising edge on an output. The rising and falling edges can also be specified and a pulse pattern generated from them.

##### **Information:**

**The specified time must always be in the future, and the set X2X Link cycle time must be taken into account for the definition of the moment.**

#### 3.10.1.2.3 Time-based measurements

NetTime Technology can be used to determine the exact moment of a measurement that has taken place. Both the starting and end moment of the measurement can be transmitted.

## 4 Register description

### 4.1 General data points

In addition to the registers described in the register description, the module has additional general data points. These are not module-specific but contain general information such as serial number and hardware variant.

General data points are described in section "Additional information - General data points" in the X20 system user's manual.

### 4.2 Function model 0 - Standard

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Configuration - General						
513	CfO_SlframeGenID	USINT				•
Configuration - System timer						
642	CfO_SystemCycleTime	UINT				•
646	CfO_SystemCycleOffset	INT				•
650	CfO_SystemCyclePrescaler	UINT				•
Configuration - Physical I/Os						
769 + (N-1) * 2	CfO_PhylOConfigCh0N (index N = 1 to 8)	USINT				•
Configuration - Direct I/O						
899	CfO_DirectIOClearMask0_7	USINT				•
903	CfO_DirectIOSetMask0_7	USINT				•
905	CfO_OutputUpdateCycle	USINT				•
Configuration - Oversampled I/O						
1025	CfO_OversampleMode	USINT				•
1027	CfO_OversampleSampleCycleID	USINT				•
1029	CfO_OversampleRelativeCycleID	USINT				•
1031	CfO_OversampleConsumeCycleID	USINT				•
1033	CfO_OversampleOutputBits	USINT				•
1035	CfO_OversampleInputBits	USINT				•
1037	CfO_OversampleOutputWindow	USINT				•
1039	CfO_OversampleInputWindow	USINT				•
1041 + (N*2)	CfO_OversampleConfigInputN (index N = 0 to 3)	USINT				•
1049 + (N*2)	CfO_OversampleConfigOutputN (index N = 0 to 3)	USINT				•
Configuration - Edge detection						
1537	CfO_EdgeDetectPollCycleID	USINT				•
1548	CfO_EdgeDetectEventEnable	UDINT				•
1665 + (N-1) * 16	CfO_EdgeDetectUnit0NMode (index N = 1 to 4)	USINT				•
1667 + (N-1) * 16	CfO_EdgeDetectUnit0NLeading (index N = 1 to 4)	USINT				•
1669 + (N-1) * 16	CfO_EdgeDetectUnit0NMaster (index N = 1 to 4)	USINT				•
1671 + (N-1) * 16	CfO_EdgeDetectUnit0NSlave (index N = 1 to 4)	USINT				•
Configuration - Motion functions						
4097	CfO_FifoSize	USINT				•
4099	CfO_Mode	SINT				•
4101	CfO_SpeedLimit	USINT				•
4103	CfO_FormatAdjust	USINT				•
4105	CfO_TimeStampRange	SINT				•
4107	CfO_PositionRange	SINT				•
4109	CfO_Reference0Range	SINT				•
4111	CfO_Reference1Range	SINT				•
4116	CfO_TimeStampDelay	DINT				•
4124	CfO_SpeedCycleTime_32bit	UDINT				•
4129	CfO_ResolPosition	SINT				•
4131	CfO_ResolSpeed	SINT				•
4220	CfO_AccelDataInit	UDINT				•
4260	CfO_Reference0Start	DINT				•
4268	CfO_Reference0StopMargin	DINT				•
4276	CfO_Reference1Start	DINT				•
4284	CfO_Reference1StopMargin	DINT				•
Configuration - SSI						
2049	CfO_CycleSelect	USINT				•
2051	CfO_PhysicalMode	USINT				•
2053	CfO_DataBits	USINT				•
2055	CfO_NullBits	USINT				•

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Configuration - Universal counter						
6145	CfO_CounterCycleSelect	USINT				•
6147	CfO_CounterMode	USINT				•
6149	CfO_LatchMode	USINT				•
6151	CfO_LatchComparator	USINT				•
6153	CounterControl	USINT			•	
	CounterReset	Bit 0				
	LatchEnable	Bit 1				
Communication - General						
546	ProtocolError (16-bit)	USINT	•			
547	ProtocolError (8-bit)	UINT	•			
550	ProtocolSequenceViolation (16-bit)	UINT	•			
551	ProtocolSequenceViolation (8-bit)	USINT	•			
Communication - Error register						
257	Error state - Output data and edge detection	USINT	•			
	OutputControlError	Bit 4				
	OutputCopyError	Bit 5				
	EdgeDetectError	Bit 6				
259	Error state - SSI	USINT	•			
	SSICycleTimeViolation	Bit 0				
	SSIParityError	Bit 1				
261	Error state - Motion functions	USINT	•			
	MovFifoEmpty	Bit 0				
	MovFifoFull	Bit 1				
	MovTargetTimeViolation	Bit 2				
	MovMaxFrequencyViolation	Bit 3				
321	Acknowledging error messages - Output data and edge detection	USINT			•	
	QuitOutputControlError	Bit 4				
	QuitOutputCopyError	Bit 5				
	QuitEdgeDetectError	Bit 6				
323	Acknowledging error messages - SSI	USINT			•	
	SSIQuitCycleTimeViolation	Bit 0				
	SSIQuitParityError	Bit 1				
325	Acknowledging error messages - Motion functions	USINT			•	
	MovQuitFifoEmpty	Bit 0				
	MovQuitFifoFull	Bit 1				
	MovQuitTargetTimeViolation	Bit 2				
	MovQuitMaxFrequencyViolation	Bit 3				
Communication - System timer						
683	SDCLifeCount	SINT	•			
Communication - Direct I/O						
915	Output state	USINT			•	
	DigitalOutput03	Bit 2				
	DigitalOutput04	Bit 3				
	DigitalOutput07	Bit 6				
	DigitalOutput08	Bit 7				
927	Input state	USINT	•			
	DigitalInput01	Bit 0				
	...	...				
	DigitalInput08	Bit 7				
Communication - Oversampled I/O (output)						
1059	Oversampling configuration	USINT			•	
	OversampleEnable	Bit 0				
	OversampleOutputValidate	Bit 1				
1063	OversampleOutputCycle	USINT			•	
	OversampleSampleOffset	USINT				
1088 + N	OversampleOutput0NSample1_8 (index N = 1 to 4)	USINT			•	
1092 + N	OversampleOutput0NSample9_16 (index N = 1 to 4)	USINT			•	
1096 + N	OversampleOutput0NSample17_24 (index N = 1 to 4)	USINT			•	
1100 + N	OversampleOutput0NSample25_32 (index N = 1 to 4)	USINT			•	
1104 + N	OversampleOutput0NSample33_40 (index N = 1 to 4)	USINT			•	
1108 + N	OversampleOutput0NSample41_48 (index N = 1 to 4)	USINT			•	
1112 + N	OversampleOutput0NSample49_56 (index N = 1 to 4)	USINT			•	
1116 + N	OversampleOutput0NSample57_64 (index N = 1 to 4)	USINT			•	
Communication - Oversampled I/O (input)						
1074	OversampleInputTime	INT	•			
1079	OversampleInputCycle	USINT	•			
1120 + N	OversampleInput0NSample64_57 (index N = 1 to 4)	USINT	•			
1124 + N	OversampleInput0NSample56_49 (index N = 1 to 4)	USINT	•			
1128 + N	OversampleInput0NSample48_41 (index N = 1 to 4)	USINT	•			
1132 + N	OversampleInput0NSample40_33 (index N = 1 to 4)	USINT	•			
1136 + N	OversampleInput0NSample32_25 (index N = 1 to 4)	USINT	•			
1140 + N	OversampleInput0NSample24_17 (index N = 1 to 4)	USINT	•			

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
1144 + N	OversampleInput0NSample16_9 (index N = 1 to 4)	USINT	•			
1148 + N	OversampleInput0NSample8_1 (index N = 1 to 4)	USINT	•			
<b>Communication - Edge detection</b>						
1794 + (N-1) * 32	EdgeDetect0NMastercount (16-bit) (index N = 1 to 4)	INT	•			
1795 + (N-1) * 32	EdgeDetect0NMastercount (8-bit) (index N = 1 to 4)	SINT	•			
1798 + (N-1) * 32	EdgeDetect0NSlavecount (16-bit) (index N = 1 to 4)	INT	•			
1799 + (N-1) * 32	EdgeDetect0NSlavecount (8-bit) (index N = 1 to 4)	SINT	•			
1804 + (N-1) * 32	EdgeDetect0NDifference (32-bit) (index N = 1 to 4)	DINT	•			
1806 + (N-1) * 32	EdgeDetect0NDifference (16-bit) (index N = 1 to 4)	INT	•			
1812 + (N-1) * 32	EdgeDetect0NMastertime (32-bit) (index N = 1 to 4)	DINT	•			
1814 + (N-1) * 32	EdgeDetect0NMastertime (16-bit) (index N = 1 to 4)	INT	•			
1820 + (N-1) * 32	EdgeDetect0NSlavetime (32-bit) (index N = 1 to 4)	DINT	•			
1822 + (N-1) * 32	EdgeDetect0NSlavetime (16-bit) (index N = 1 to 4)	INT	•			
<b>Communication - Motion functions</b>						
4225	MovementControl	USINT			•	
	MovEnable - For position control	Bit 0				
	MovEnable - For speed control	Bit 1				
	MovReset - Movement reset (immediate stop)	Bit 7				
4244	MovTargetTime (32-bit)	DINT			•	
4246	MovTargetTime (16-bit)	INT			•	
4252	MovTargetPosition (32-bit)	DINT			•	
4254	MovTargetPosition (16-bit)	INT			•	
4260	MovReference1Start (32-bit)	DINT			•	
4262	MovReference1Start (16-bit)	INT			•	
4268	MovReference1StopMargin (32-bit)	DINT			•	
4270	MovReference1StopMargin (16-bit)	INT			•	
4276	MovReference2Start (32-bit)	DINT			•	
4278	MovReference2Start (16-bit)	INT			•	
4284	MovReference2StopMargin (32-bit)	DINT			•	
4286	MovReference2StopMargin (16-bit)	INT			•	
4212	MovSpeed (32-bit)	DINT			•	
4210	MovSpeed (16-bit)	INT			•	
4220	MovAcceleration (32-bit)	UDINT			•	
4218	MovAcceleration (16-bit)	UINT			•	
4292	MovTimeValid (32-bit)	DINT	•			
4294	MovTimeValid (16-bit)	INT	•			
4300	MovPosition (32-bit)	DINT	•			
4302	MovPosition (16-bit)	INT	•			
<b>Communication - SSI</b>						
2084	SSITimeValid (32-bit)	DINT	•			
2086	SSITimeValid (16-bit)	INT	•			
2092	SSITimeChanged (32-bit)	DINT	•			
2094	SSITimeChanged (16-bit)	INT	•			
2100	SSIPosition (32-bit)	(U)DINT	•			
2102	SSIPosition (16-bit)	UINT	•			
<b>Communication - Universal counter</b>						
6303	LatchCount	SINT	•			
6308	CounterTimeValid (32-bit)	DINT	•			
6310	CounterTimeValid (16-bit)	INT	•			
6324	Counter01TimeChanged (32-bit)	DINT	•			
6326	Counter01TimeChanged (16-bit)	INT	•			
6332	Counter02TimeChanged (32-bit)	DINT	•			
6334	Counter02TimeChanged (16-bit)	INT	•			
6340	CounterValue01 (32-bit)	DINT	•			
6342	CounterValue01 (16-bit)	INT	•			
6348	CounterValue02 (32-bit)	DINT	•			
6350	CounterValue02 (16-bit)	INT	•			
6356	CounterLatch01 (32-bit)	DINT	•			
6358	CounterLatch01 (16-bit)	INT	•			
6364	CounterLatch02 (32-bit)	DINT	•			
6366	CounterLatch02 (16-bit)	INT	•			
6372	CounterRel01 (32-bit)	DINT	•			
6374	CounterRel01 (16-bit)	INT	•			
6380	CounterRel02 (32-bit)	DINT	•			
6382	CounterRel02 (16-bit)	INT	•			

## 4.3 General registers

### 4.3.1 Defining the moment for generating synchronous input data

Name:

CfO\_SlframeGenID

"SI frame generation" in the Automation Studio I/O configuration.

When the synchronous input data is generated for transfer is defined in this register.

Data type	Values	Information
USINT	9	X2X cycle optimized
	14	Fast reaction

### 4.3.2 Number of X2X protocol errors

Name:

ProtocolError

This register contains an error counter that specifies the number of X2X protocol errors. In the I/O configuration, parameter "Network information" can be used to help configure a data point for this register with a bit width of 8 or 16 bits in the I/O mapping.

Data type	Values	Information
USINT	0 to 255	Error counter (8-bit)
UINT	0 to 65535	Error counter (16-bit)

### 4.3.3 Number of X2X sequence violations

Name:

ProtocolSequenceViolation

This register contains an error counter that specifies the number of X2X sequence violations. In the I/O configuration, parameter "Network information" can be used to help configure a data point with a bit width of 8 or 16 bits in the I/O mapping.

Data type	Values	Information
USINT	0 to 255	Error counter (8-bit)
UINT	0 to 65535	Error counter (16-bit)

### 4.3.4 System clock counter for checking the validity of the data frame

Name:

SDCLifeCount

Counter that is incremented with each system timer cycle. "SDC information" in the Automation Studio I/O configuration can be used to enable this register in the I/O mapping as data point "SDCLifeCount".

The 8-bit counter register is needed for the SDC software package. It is incremented with the system clock to allow the SDC to check the validity of the data frame.

Data type	Values
SINT	-128 to 127

## 4.4 System timer

The module's individual functions all depend on a system timer.

### 4.4.1 Setting the cycle time of the system timer

Name:

CfO\_SystemCycleTime

"Cycle time" in the Automation Studio I/O configuration.

The cycle time of the system timer can be set in steps of 1/8  $\mu$ s in this register. The value entered in the Automation Studio I/O configuration is automatically multiplied by 8.

#### Information:

**A setting <50  $\mu$ s has a negative effect on the minimum X2X cycle time!**

Data type	Values	Information
UINT	200 to 2047	System timer cycle time in steps of 1/8 $\mu$ s (25 to 255.875 $\mu$ s)

#### 4.4.2 Offsetting the synchronization moment of the system cycle

Name:

CfO\_SystemCycleOffset

"Cycle offset" in the Automation Studio I/O configuration.

The synchronization moment for the system cycle can be offset in steps of 1/8 µs in this register. The value entered in the Automation Studio I/O configuration is automatically multiplied by 8.

Data type	Values	Information
INT	-32768 to 32767	Cycle offset in steps of 1/8 µs (-4096 to 4095.875 µs)

#### 4.4.3 Configuration of the cycle prescaler

Name:

CfO\_SystemCyclePrescaler

"Cycle prescaler" in the Automation Studio I/O configuration.

The prescaler for setting the [prescaled system timer](#) can be configured in this register. The cycle time of the specified system timer is a product of the system timer multiple set in this register.

Data type	Values	Information
UINT	2 to 128	Multiple of the system timer

### 4.5 Physical I/O configuration

#### 4.5.1 Configuring the I/O channels

Name:

CfO\_PhylIOConfigCh01 to CfO\_PhylIOConfigCh08

Each physical I/O channel can be configured individually in these registers.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Push driver <sup>1)</sup>	0	Disabled
		1	Enabled
1	Pull driver <sup>1)</sup>	0	Disabled
		1	Enabled
2	Input inverted	0	Not inverted
		1	Inverted
3	Output inverted <sup>1)</sup>	0	Not inverted
		1	Inverted
4 - 7	Output function <sup>1)</sup>	0 to 15	See "Overview of output channel functions".

1) Only available for I/O channels 3, 4, 7 and 8

#### Overview of output channel functions

Values of bits 4 to 7	Output channel 3	Output channel 4	Output channel 7	Output channel 8
0	Direct I/O			
1				SSI clock output
2	ABR emulation (A)	ABR emulation (B)		
3	Up/Down emulation (direction)	Up/Down emulation (frequency)	Emulation (reference 1) <sup>1)</sup>	Emulation (reference 2) <sup>1)</sup>
4 - 15	Reserved			

1) Applies to both ABR and up/down emulation of output channels 3 and 4

## 4.6 Direct I/O

"Direct I/O" makes it possible to use the physical I/Os like normal I/Os.

### 4.6.1 Direct operation of the output channel - Reset

Name:

CfO\_DirectIOClearMask0\_7

"Direct operation of output channel 03" to "Direct operation of output channel 08" in the Automation Studio I/O configuration.

If the bit for the respective channel is set in this register, then the output is reset as soon as its direct I/O output channel (register ["DigitalOutput" on page 32](#) or "DigitalOutput0x" in the Automation Studio I/O mapping) is reset.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	Reserved	-	
2	Output channel 3	0	No change
		1	Reset channel
3	Output channel 4	0	No change
		1	Reset channel
4 - 5	Reserved	-	
6	Output channel 7	0	No change
		1	Reset channel
7	Output channel 8	0	No change
		1	Reset channel

### 4.6.2 Direct operation of the output channel - Set

Name:

CfO\_DirectIOSetMask0\_7

"Direct operation of output channel 03" to "Direct operation of output channel 08" in the Automation Studio I/O configuration.

If the bit for the respective channel is set in this register, then the output is set as soon as its direct I/O output channel (register ["DigitalOutput" on page 32](#) or "DigitalOutput0x" in the Automation Studio I/O mapping) is set.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	Reserved	-	
2	Output channel 3	0	No change
		1	Set channel
3	Output channel 4	0	No change
		1	Set channel
4 - 5	Reserved	-	
6	Output channel 7	0	No change
		1	Set channel
7	Output channel 8	0	No change
		1	Set channel

### 4.6.3 Direct operation of the output channel - Moment of data output

Name:

CfO\_OutputUpdateCycle

The moment when data is output is set with this register.

Data type	Values	Information
USINT	10	X2X cycle optimized (jitter-free)
	15	Fast reaction (with jitter)

#### 4.6.4 Output state

Name:

DigitalOutput03 and DigitalOutput04, DigitalOutput07 and DigitalOutput08

This register contains the bits for controlling the direct I/O output channels. Depending on the configuration of registers "[CfO\\_DirectIOClearMask0\\_7](#)" on page 31 and "[CfO\\_DirectIOSetMask0\\_7](#)" on page 31, the digital outputs are set to the status of the respective bit in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	Reserved	-	
2	DigitalOutput03	0 or 1	Output state of channel 3
3	DigitalOutput04	0 or 1	Output state of channel 4
4 - 5	Reserved	-	
6	DigitalOutput07	0 or 1	Output state of channel 7
7	DigitalOutput08	0 or 1	Output state of channel 8

#### 4.6.5 Input state

Name:

DigitalInput01 to DigitalInput08

The state of the digital input channels is contained in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	DigitalInput01	0 or 1	Input state of channel 1
...	...	...	
7	DigitalInput08	0 or 1	Input state of channel 8

### 4.7 Oversampled I/O

#### 4.7.1 Configuration of the output control buffer

Name:

CfO\_OversampleMode

"Output mode" in the Automation Studio I/O configuration

"Output control mode" in the Automation Studio I/O configuration

The output control buffer can be configured globally for all channels in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Addressing the output control buffer "Output mode"	0	Absolute addressing of the output control buffer
		1	Relative addressing of the output control buffer
1	Cyclic output control "Output control mode"	0	Single - Output control buffer entry is marked invalid after execution.
		1	Continuous - Output control buffer entry is not changed.
2 - 7	Reserved	-	

#### 4.7.2 Configuration of the source for the sample cycle

Name:

CfO\_OversampleSampleCycleID

"Sample cycle" in the Automation Studio I/O configuration.

The source for the sample cycle is configured in this register. For details, see "[Reference cycle](#)" on page 8.

Data type	Values	Information
USINT	2	System timer
	3	Prescaled system timer
	10	AOAI
	14	SOSI



### 4.7.3 Configuration of the source for the user interface reference cycle

Name:

CfO\_OversampleRelativeCycleID

"Reference cycle" in the Automation Studio I/O configuration.

The source for the user interface reference cycle is configured in this register. For details, see ["Reference cycle" on page 8](#).

Data type	Values	Information
USINT	2	System timer
	3	Prescaled system timer
	10	AOAI
	14	SOSI

### 4.7.4 Defining the moment for copying the data to the output control buffer

Name:

CfO\_OversampleConsumeCycleID

"Output copy cycle" in the Automation Studio I/O configuration.

At the time of the output copy cycle, data is copied from the ["OversampleOutput0NSample" on page 35](#) registers into the output control buffer. For details, see ["Oversampled I/O output data" on page 9](#).

Data type	Values	Information
USINT	10	X2X cycle optimized The output data is copied to the output control buffer with the AOAI interrupt of the X2X cycle.
	15	Fast reaction The output data is copied to the output control buffer immediately after being received.

### 4.7.5 Number of output bits to be transferred

Name:

CfO\_OversampleOutputBits

"User interface size" in the Automation Studio I/O configuration.

Specifies how many bits are transferred from the ["OversampleOutput0NSample" on page 35](#) registers to the output control buffers at the [output copy cycle](#) moment.

Data type	Values	Information
USINT	1 to 64	Output bits

### 4.7.6 Number of input bits to be transferred

Name:

CfO\_OversampleInputBits

"User interface size" in the Automation Studio I/O configuration.

Specifies how many bits are transferred from the input status buffer to the ["OversampleInput0NSample" on page 36](#) registers during [SI frame generation](#).

Data type	Values	Information
USINT	1 to 64	Input bits

### 4.7.7 Write area in the output control buffer

Name:

CfO\_OversampleOutputWindow

"Output control mode" in the Automation Studio I/O configuration.

Defines the area in the output control buffer to which data is permitted to be written. The window is always shifted relative to the current sample position (e.g. a value of 128 means that the 128 bits following the current sample cycle can be written). [OutputCopyError](#) is triggered if an attempt is made to write output sample data to a location outside of this window.

In Automation Studio, the value for this register is set to 128 bits with "Output control mode = Single" and to 255 bits with "Output control mode = Continuous".

Data type	Values	Information
USINT	0 to 255	Output window

#### 4.7.8 Defining the moment for referencing input data

Name:

CfO\_OversampleInputWindow

"Input mode" in the Automation Studio I/O configuration.

The "oversample input window" defines when the input data is referenced. For details, see ["Defining the reference time point" on page 12](#).

In Automation Studio, the value for this register is set to 63 with "Input mode = Referenced values" and to 0 with "Input mode = Most recent values".

Data type	Values	Information
USINT	0 to 63	Input window

#### 4.7.9 Assigning between the physical input channel and oversample I/O input

Name:

CfO\_OversampleConfigInput

"Oversample I/O 01 → Input" to "Oversample I/O 04 input" in the Automation Studio I/O configuration.

Which physical input channel an oversample I/O input should be linked to is defined in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Number of the physical input channel	0	Input channel 1
		..	
		7	Input channel 8
4 - 7	Reserved	-	

#### 4.7.10 Configuring the outputs of the oversampling channels

Name:

CfO\_OversampleConfigOutput

"Oversample I/O 01 → Output" to "Oversample I/O 04 → Output" in the Automation Studio I/O configuration

"Oversample I/O 01 → Output control" to "Oversample I/O 04 → Output control" in the Automation Studio I/O configuration

"Oversample I/O 01 → Output default value" to "Oversample I/O 04 → Output default value" in the Automation Studio I/O configuration

This register helps configure the outputs of the individual oversample channels.

The "Output default state" bits define which level the respective output takes on before oversampling is started. In addition, the output is set to the defined "Output default state" in the event of an error.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Number of the physical output channel "Oversample I/O 0x → Output"	2	Output channel 3
		3	Output channel 4
		6	Output channel 7
		7	Output channel 8
4	Output: Clear "Oversample I/O 0x → Output control"	0	Output cannot be reset by the oversample channel.
		1	Output can be reset by the oversample channel.
5	Output: Set "Oversample I/O 0x → Output control"	0	Output cannot be set by the oversample channel.
		1	Output can be set by the oversample channel.
6	Default output state: Clear "Oversample I/O 0x → Output default value"	0	Output not cleared by default
		1	Output cleared by default
7	Default output state: Set "Oversample I/O 0x → Output default value"	0	Output not set by default
		1	Output set by default

#### 4.7.11 Oversampling configuration

Name:

OversampleEnable

OversampleOutputValidate

The oversampling and copy process for the output buffer can be configured in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	OversampleEnable	0	Disables oversampling (with the next reference cycle)
		1	Enables oversampling (with the next reference cycle)
1	OversampleOutputValidate	0	Disable the copy procedure to the output control buffer.
		1	Enables the copy procedure to the output control buffer. <ul style="list-style-type: none"> <li>Used to synchronize the oversampling procedure at startup.</li> <li>This makes it possible to prevent new data from being transferred to the "OversampleOutput0NSample" on <a href="#">page 35</a> registers in each X2X cycle.</li> </ul>
2 - 7	Reserved	-	

#### 4.7.12 Address of the new output sampling data in the output control buffer

Name:

OversampleOutputCycle

When absolute addressing of the output control buffer is being used, this register specifies the address from which the new output sample data should be copied to the output control buffer.

Data type	Values	Information
USINT	0 to 255	Address of the output control buffer

#### 4.7.13 Offset of new output sample data

Name:

OversampleSampleOffset

With relative addressing of the output control buffer, this register serves as an offset for the new output sample data (to the [reference cycle](#) Current sample address + Offset = Address to which the new output sample data is copied into the output control buffer).

Data type	Values	Information
USINT	0 to 255	Offset of output sample data

#### 4.7.14 Oversample output sample data

Name:

OversampleOutput01Sample1\_8 to OversampleOutput04Sample1\_8

OversampleOutput01Sample9\_16 to OversampleOutput04Sample9\_16

OversampleOutput01Sample17\_24 to OversampleOutput04Sample17\_24

OversampleOutput01Sample25\_32 to OversampleOutput04Sample25\_32

OversampleOutput01Sample33\_40 to OversampleOutput04Sample33\_40

OversampleOutput01Sample41\_48 to OversampleOutput04Sample41\_48

OversampleOutput01Sample49\_56 to OversampleOutput04Sample49\_56

OversampleOutput01Sample57\_64 to OversampleOutput04Sample57\_64

Contains the oversample output sample data. For details, see ["Output data" on page 13](#).

Data type	Values	Information
USINT	0 to 255	Output sample data

#### 4.7.15 X2X NetTime of the input data

Name:

OversampleInputTime

This register contains the 2 low-order bytes of the X2X NetTime from the moment at which the oversample input data was referenced. This provides an easy way to accurately calculate the moment of each individual input sample.

For additional information about NetTime and timestamps, see "[NetTime Technology](#)" on page 23.

Data type	Values	Information
INT	-32768 to 32767	X2X NetTime of the input data in microseconds

#### 4.7.16 Input status buffer address of the input sample data

Name:

OversampleInputCycle

This register contains the input status buffer address of the input sample data.

In addition, the value in this register can be used to reference an absolute addressing of the output control buffer.

Data type	Values	Information
USINT	0 to 255	Input status buffer address

#### 4.7.17 Input sample data

Name:

OversampleInput01Sample8\_1 to OversampleInput04Sample8\_1

OversampleInput01Sample16\_9 to OversampleInput04Sample16\_9

OversampleInput01Sample24\_17 to OversampleInput04Sample24\_17

OversampleInput01Sample32\_25 to OversampleInput04Sample32\_25

OversampleInput01Sample40\_33 to OversampleInput04Sample40\_33

OversampleInput01Sample48\_41 to OversampleInput04Sample48\_41

OversampleInput01Sample56\_49 to OversampleInput04Sample56\_49

OversampleInput01Sample64\_57 to OversampleInput04Sample64\_57

The data of the 4 oversample input status buffers are copied to this register at the moment of [SI frame generation](#).

For details, see "[Input data](#)" on page 12.

Data type	Values	Information
USINT	0 to 255	Input sample data

### 4.8 Edge detection

The module's edge detection function allows edges to be measured with microsecond precision.

#### 4.8.1 Configuring the source for the polling cycle

Name:

CfO\_EdgeDetectPollCycleID

"Polling cycle" in the Automation Studio I/O configuration.

The source for the polling cycle can be configured in this register.

#### Information:

The polling cycle must be  $\leq 255 \mu\text{s}$ . If the configured cycle  $> 255 \mu\text{s}$ , [EdgeDetectError](#) occurs.

Data type	Values	Information
USINT	2	System timer
	3	Prescaled system timer

## 4.8.2 Edge detection mode

Name:

CfO\_EdgeDetectEventEnable

"Edge detection mode" in the Automation Studio I/O configuration.

The bits in this register define on which edges of the individual input channels an interrupt should be triggered for edge detection.

In the Automation Studio I/O configuration, this register is initialized with 0x00000000 when "Edge detection mode = Polling" and with 0xFFFFFFFF when "Edge detection mode = Event-triggered".

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Physical input 1	0	No interrupt triggered on falling edge
		1	Interrupt triggered on falling edge
...	...	...	...
7	Physical input 8	0	No interrupt triggered on falling edge
		1	Interrupt triggered on falling edge
8 - 15	Reserved	-	
16	Physical input 1	0	No interrupt triggered on rising edge
		1	Interrupt triggered on rising edge
...	...	...	...
23	Physical input 8	0	No interrupt triggered on rising edge
		1	Interrupt triggered on rising edge
24 - 31	Reserved	-	

## 4.8.3 Setting the time base, slave edge and master edge

Name:

CfO\_EdgeDetectUnit01Mode to CfO\_EdgeDetectUnit04Mode

"Time base" in the Automation Studio I/O configuration

"Slave edge" in the Automation Studio I/O configuration

"Master edge" in the Automation Studio I/O configuration

The time base and edge behavior are set in this register. For details about the time base, see ["Time base" on page 16](#).

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	"Time base"	0	Local time 1/8 µs (Automation Studio: Local resolution 1/8 usec)
		1	Local time 1 µs (Automation Studio: Local resolution 1 usec)
		2	NetTime 1/8 µs (Automation Studio: Nettime resolution 1/8 usec)
		3	NetTime 1 µs (Automation Studio: Nettime resolution 1 usec)
2 - 5	Reserved	-	
6	"Slave edge"	0	Disabled
		1	Enabled
7	"Master edge"	0	Disabled
		1	Enabled

## 4.8.4 Load position of the slave time from the FIFO buffer

Name:

CfO\_EdgeDetectUnit01Leading to CfO\_EdgeDetectUnit04Leading

"Slave leading" in the Automation Studio I/O configuration.

This value defines the position from which the slave time should be retrieved from the FIFO buffer when a master edge occurs. This can be used to measure average periodic signals over multiple cycles.

Data type	Values	Information
USINT	0 to 255	Position in the FIFO buffer for slave edges

#### 4.8.5 Source of the master edge per edge detection unit

Name:

CfO\_EdgeDetectUnit01Master to CfO\_EdgeDetectUnit04Master

"Master edge" in the Automation Studio I/O configuration.

The source of the master edge for the respective "edge detection unit" is defined in this register.

Data type	Values	Information
USINT	0	Rising edge on physical input 1
	...	...
	7	Rising edge on physical input 8
	16	Falling edge on physical input 1
	...	...
	23	Falling edge on physical input 8

#### 4.8.6 Source of the slave edge per edge detection unit

Name:

CfO\_EdgeDetectUnit01Slave to CfO\_EdgeDetectUnit04Slave

"Slave edge" in the Automation Studio I/O configuration.

The source of the slave edge for the respective "edge detection unit" is defined in this register.

Data type	Values	Information
USINT	0	Rising edge on physical input 1
	...	...
	7	Rising edge on physical input 8
	16	Falling edge on physical input 1
	...	...
	23	Falling edge on physical input 8

#### 4.8.7 Number of detected master edges

Name:

EdgeDetect01Mastercount to EdgeDetect04Mastercount

Detected master edges are counted in this register.

Data type	Values	Information
SINT	-128 to 127	Number of detected master edges (8-bit)
INT	-32768 to 32767	Number of detected master edges (16-bit)

#### 4.8.8 Number of detected slave edges

Name:

EdgeDetect01Slavecount to EdgeDetect04Slavecount

Continuously counts the detected slave edges. The contents of this register are only updated on a master edge. This counter can detect if several slave edges occur before a master edge.

Data type	Values	Information
SINT	-128 to 127	Number of detected slave edges (8-bit)
INT	-32768 to 32767	Number of detected slave edges (16-bit)

#### 4.8.9 Difference between a master and slave edge

Name:

EdgeDetect01Difference to EdgeDetect04Difference

This register contains the time difference between a master edge and the last slave edge addressed by "Slave leading" on page 37.

Data type	Values	Information
INT	-32768 to 32767	Slave edge / Master edge time difference (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Slave edge / Master edge time difference (32-bit)

#### 4.8.10 NetTime when a master edge occurs

Name:

EdgeDetect01Mastertime to EdgeDetect04Mastertime

The exact NetTime is copied to this register when a master edge occurs.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime master edge in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime master edge in microseconds (32-bit)

#### 4.8.11 NetTime when a slave edge occurs

Name:

EdgeDetect01Slavetime to EdgeDetect04Slavetime

When a master edge occurs, the exact NetTime of any slave edge that may have occurred prior to the master edge and addressed by ["Slave leading" on page 37](#) is copied to this register.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime slave edge in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime slave edge in microseconds (32-bit)

### 4.9 Motion functions

Encoder emulation can be used to generate up/down counters (direction/frequency) and ABR encoder signals.

#### 4.9.1 FIFO buffer size

Name:

FifoSize

"Number of FIFO buffer entries" in the Automation Studio I/O configuration.

Determines the size of the FIFO buffer for ["MovTargetTime" on page 44](#) and ["MovTargetPosition" on page 44](#). One [timestamp](#) and one position that should be reached by the timestamp can be transferred to the FIFO buffer per X2X cycle.

Data type	Values	Information
USINT	0	FIFO buffer disabled
	3	8 entries ( $2^3$ )
	4	16 entries ( $2^4$ )
	5	32 entries ( $2^5$ )
	6	64 entries ( $2^6$ )
	7	128 entries ( $2^7$ )
	8	256 entries ( $2^8$ )

#### 4.9.2 Motion function mode

Name:

CfO\_Mode

The mode of the motion functions can be configured in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Must be enabled when working without timestamps. Enabled in Automation Studio if: <ul style="list-style-type: none"> <li>"Movement = Speed control"</li> <li>"Movement = Position control" and "Data format / Mode of the preset time value = Local time"</li> </ul>	0	Disabled
		1	Enabled
1	If this function is enabled, repositioning is triggered as soon as the value in register <a href="#">"MovTargetPosition" on page 44</a> changes. Enabled in Automation Studio if: <ul style="list-style-type: none"> <li>"Movement = Position control" and "Data format / Mode of the preset time value = Local time"</li> </ul>	0	No position control (speed control)
		1	Position control enabled (position control)
2	Reference mode 1 "Configuration of reference pulse #1" in the Automation Studio I/O configuration.	0	Starting/End position
		1	Starting position and span
3	Reference mode 2 "Configuration of reference pulse #1" in the Automation Studio I/O configuration.	0	Starting/End position
		1	Starting position and span
4 - 7	Reserved	-	

#### 4.9.3 Maximum output frequency

Name:

CfO\_SpeedLimit

"Max. movement frequency" in the Automation Studio I/O configuration.

Configures the maximum permissible output frequency and the maximum internal computing frequency. The higher internal computing frequencies of 500 kHz or 1, 2, 4, 8, 16, 32 and 64 MHz can only be achieved by configuring n bits as decimal places (see register ["CfO\\_ResolPosition" on page 42](#)).

Data type	Values	Max. increment frequency	Max. frequency for frequency output channel	Max. frequency for A/B output channel
USINT	253	64 MHz	125 kHz	62.5 kHz
	254	32 MHz		
	255	16 MHz		
	0	8 MHz		
	1	4 MHz		
	2	2 MHz		
	3	1 MHz		
	4	500 kHz		
	5	250 kHz (default)		
	6	125 kHz	62.5 kHz	31.25 kHz

#### Information:

In movement "Position control", increment frequencies 16, 32 and 64 MHz are not permitted to be used when 29-bit timestamps are set (see register ["CfO\\_TimeStampRange" on page 41](#)) due to an internal range overrun.

#### 4.9.4 Number of absolute bits that can be output

Name:

CfO\_FormatAdjust

The number of bits that can be output absolutely on the signal output are determined in this register (e.g. for a direction/frequency signal, the least significant bit can be output directly on the frequency output; for an AB signal, 2 bits are possible).

Data type	Values	Information
USINT	1 to 2	Number of absolute bits (default value in Automation Studio = 1)



#### 4.9.5 Width of the transferred timestamp data

Name:

CfO\_TimeStampRange

"Data format / Mode of the preset time value" in the Automation Studio I/O configuration.

The width of the transferred **timestamp data** in the module is configured in this register.

##### Information:

Since the module works internally with 1/8 µs resolution, timestamp data is processed internally with a maximum width of 29 bits.

Data type	Values	Information
SINT	16	16-bit timestamp (selection "16-bit" in the Automation Studio I/O configuration)
	24	24-bit timestamp (selection "Local time" or movement "Speed control" in the Automation Studio I/O configuration)
	29	29-bit timestamp (selection "29-bit" in the Automation Studio I/O configuration)

#### 4.9.6 Number of bits for position control

Name:

CfO\_PositionsRange

"Target position range" in the Automation Studio I/O configuration.

The number of bits for position control are configured in this register. "PositionRange" must be reduced if the motion function should follow the absolute value of a 12-bit SSI encoder, for example. In this case, the bit width of the movement position must also be limited to the number of bits of the encoder; otherwise, the movement position would not be overrun if the encoder overflows. In this case, the module would attempt (in the opposite direction) to reach the position of an encoder that had just overflow.

##### Example

The 12-bit SSI encoder overflows from 2047 to -2048. The module would generate 4096 negative increments if more than 12 bits were set for "CfO\_PositionRange" in order to reach position -2048 from the position 2047.

##### Information:

If the 16-bit value of register **"MovPosition"** on page 45 is used, then the number of bits of the position must also be limited to ≤16; otherwise, this would also result in incorrect overflow behavior.

Data type	Values	Information
SINT	8 to 32	Number of bits for position control

#### 4.9.7 Number of bits for reference position comparison

Name:

CfO\_Reference0Range to CfO\_Reference1Range

"Reference #1 range" to "Reference #2 range" in the Automation Studio I/O configuration.

The number of bits used for the reference position comparison are configured in this register. This makes it possible to generate a reference pulse every 2<sup>n</sup> increments.

##### Information:

The number of bits set in this register is not permitted to exceed the number of bits of data points **"MovReferenceStart"** on page 43 and **"MovReferenceStopMargin"** on page 44.

Data type	Values	Information
SINT	4 to 32	Number of bits for position comparison

#### 4.9.8 Timestamp delay

Name:

CfO\_TimeStampDelay

"Default time delay" in the Automation Studio I/O configuration.

All [timestamps](#) are delayed by the value set in this register in microseconds.

##### Information:

When setting to "Local time" in register ["CfO\\_TimeStampRange" on page 41](#), a value at least 2x the X2X cycle time in microseconds must be entered here.

Data type	Values	Information
DINT	0 to 1,000,000	Timestamp delay in microseconds

#### 4.9.9 Control period for mode "Speed control"

Name:

CfO\_SpeedCycleTime\_32Bit

"Control period" in the Automation Studio I/O configuration.

The control period for mode "Speed control" can be configured in 1/8 µs steps in this register.

##### Information:

The value set in the Automation Studio I/O configuration under "Control period" is automatically multiplied by 8 and then used as [CfO\\_SpeedCycleTime\\_32bit](#).

Data type	Values	Information
UDINT	400 to 40000	Control period for mode "Speed control"

#### 4.9.10 Minimizing jitter for the position

Name:

CfO\_ResolPosition

"Position resolution" in the Automation Studio I/O configuration.

This register contains the number of bits as decimal place for jitter reduction. Within the module, a  $2^n$  ( $n$  = number of decimal places) higher frequency is used for calculation, which results in edge switching times with a higher resolution. The output switching frequency is not increased from a hardware perspective, but the edge moment is more precise.

Data type	Values	Information
SINT	0	Default, no decimal places
	1 to 14	Selection of bits as decimal places

##### Information:

It is important to note that each configured decimal place also limits the maximum number range by that number of bits.

For example: 0 decimal places → Maximum position range = 29-bit

3 decimal places → Maximum position range = 26-bit

It is also important to note that register ["CfO\\_SpeedLimit" on page 40](#) must be adjusted for these higher computing frequencies according to the number of configured decimal places.

#### 4.9.11 Minimizing jitter for the speed

Name:

CfO\_ResolSpeed

"Speed resolution" in the Automation Studio I/O configuration.

This register contains the number of bits for minimizing the jitter of the speed value as decimal places. Within the module, a  $2^n$  ( $n$  = number of decimal places) higher frequency is calculated, resulting in speed values with higher resolution.

A 16-bit or 32-bit speed value is basically configured in the Automation Studio I/O configuration due to the bit limitation. Since the internal calculation is always based on a 32-bit configuration, offset 16 must always be added to the desired decimal places for 16-bit configurations.

Data type	Values	Information
SINT	0 to 31	Selection of bits as decimal places. Default value in Automation Studio = 24

#### Information:

It is important to note that each configured decimal place also limits the maximum number range by that number of bits.

#### 4.9.12 Acceleration value

Name:

CfO\_AccelDataInit

MovAcceleration

"Acceleration value" in the Automation Studio I/O configuration.

The acceleration value in increments per [controller period](#)<sup>2</sup> is contained in this register:

- 32-bit: 16,777,216 (0x01000000) corresponds to 1 increment per control period<sup>2</sup>
- 16-bit: 256 (0x0100) corresponds to 1 increment per control period<sup>2</sup>

Data type	Values	Information
UINT	0 to 65535	Acceleration value (16-bit)
UDINT	0 to 4,294,967,296	Acceleration value (32-bit)

#### 4.9.13 Starting position of the reference pulses

Name:

CfO\_Reference0Start to CfO\_Reference1Start

MovReference1Start to MovReference2Start

"Starting position" in the Automation Studio I/O configuration.

The starting position for the reference pulses is contained in these registers.

If the direction is positive, the output (R) is set when the starting position is reached. In the negative direction, the output is reset as soon as the starting position is undershot.

Data type	Values	Information
INT	-32768 to 32767	Starting position (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Starting position (32-bit)

#### 4.9.14 End position or range of the reference pulse output

Name:

CfO\_Reference0StopMargin to CfO\_Reference1StopMargin

MovReference1StopMargin to MovReference2StopMargin

"End position or range" in the Automation Studio I/O configuration.

The end position or range in which the reference pulse is output can be configured in these registers.

If setting "Reference mode x = Starting/End position" is used in register "CfO\_Mode" on page 40, the output (R) is reset when the end position is reached if the direction is positive. In the negative direction, the output is set as soon as the end position is undershot.

When using "Reference mode x = Starting position and span", the contents of this register are added to the starting position and the resulting sum is used as the end position.

Data type	Values	Information
INT	-32768 to 32767	End position (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	End position (32-bit)

#### 4.9.15 Enabling position and speed control

Name:

MovEnable (position control)

MovEnable (speed control)

MovReset

This register can be used to enable position and speed control.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	MovEnable - For position control	0	Position control disabled
		1	Position control enabled
1	MovEnable - For speed control	0	Speed control disabled
		1	Speed control enabled
2 - 6	Reserved	-	
7	MovReset - Movement reset (immediate stop)	0	No function
		1	Reset active

#### 4.9.16 Timestamp data of the target position

Name:

MovTargetTime

The **timestamp data** is contained in this register. Each time this register changes, the new position data ("MovTargetPosition" on page 44) and timestamp data are transferred to the FIFO buffer. If bit 1 for speed control "MovEnable = True", the module calculates the output speed (frequency) so that "MovTargetPosition" is reached based on "MovTargetTime".

Data type	Values	Information
INT	-32768 to 32767	Timestamp in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Timestamp in microseconds (32-bit)

### Information:

Only 29 bits are processed internally by this register.

#### 4.9.17 Data of the target position

Name:

MovTargetPosition

The position data is contained in this register.

Data type	Values	Information
INT	-32768 to 32767	Position (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Position (32-bit)

#### 4.9.18 Velocity setpoint

Name:

MovSpeed

The speed setpoint for mode "Speed control" in increments per [control period](#) is contained in this register.

- 32-bit: 16,777,216 (0x01000000) corresponds to 1 increment per control period
- 16-bit: 256 (0x0100) corresponds to 1 increment per control period

Data type	Values	Information
INT	-32768 to 32767	Speed setpoint (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Speed setpoint (32-bit)

#### 4.9.19 NetTime of the current position

Name:

MovTimeValid

The NetTime of the current position is contained in this register.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime of the current position in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime of the current position in microseconds (32-bit)

#### 4.9.20 Current position

Name:

MovPosition

The current position is contained in this register.

Data type	Values	Information
INT	-32768 to 32767	Current position (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Current position (32-bit)

### 4.10 Synchronous Serial Interface (SSI)

Synchronous Serial Interface makes it possible to receive data from SSI absolute encoders.

#### 4.10.1 Update cycle - Starting SSI transfer

Name:

CfO\_CycleSelect

"Update cycle" in the Automation Studio I/O configuration.

The update cycle for starting an SSI transfer can be configured in this register. For details, see ["Reference cycle" on page 8](#).

Data type	Values	Information
USINT	2	System timer
	3	Prescaled system timer
	10	AOAI
	14	SOSI

#### 4.10.2 Configuring the SSI interface

Name:

CfO\_PhysicalMode

"Parity bit" in the Automation Studio I/O configuration

"Monostable multivibrator testing" in the Automation Studio I/O configuration

"Data format" in the Automation Studio I/O configuration

"Clock frequency" in the Automation Studio I/O configuration

The SSI interface is configured in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	"Parity bit" <sup>(1)</sup>	00	Disabled
		01	Even parity
		10	Odd parity
		11	Ignore (the parity bit is transferred but not evaluated)
2 - 3	"Monostable multivibrator testing" <sup>(2)</sup>	00	Disabled
		01	Low level (the data signal is checked for a low level after the monostable multivibrator has returned to its stable state)
		10	High level (data signal is checked for high level after the monostable multivibrator has returned to its stable state)
		11	Ignore (the necessary clock is triggered but not evaluated)
4	"Data format"	0	Encoder with binary data output
		1	Encoder with Gray code. The position data is converted into binary format by the module.
5	Reserved	-	
6 - 7	"Clock frequency"	00 to 10	Not permitted
		11	125 kHz

1) If the parity bit is not correct, [SSIParityError](#) is generated and the position data is not applied to register ["SSIPosition"](#) on page 47.

2) As long as the data signal has not reached the level defined for monostable multivibrator testing after the transfer, no new SSI transfer is started. This will trigger error [SSICycleTimeViolation](#).

#### 4.10.3 Valid number of SSI data bits

Name:

CfO\_DataBits

"Valid number of SSI data bits" in the Automation Studio I/O configuration.

Determines the number of valid data bits to be transferred via SSI. The valid data bits are used for ["SSIPosition"](#) on page 47.

Data type	Values	Information
USINT	1 to 32	Number of valid data bits

#### 4.10.4 Number of leading zero bits

Name:

CfO\_NullBits

"Number of leading zero bits" in the Automation Studio I/O configuration.

The number of leading zero bits can be configured in this register. The leading zero bits may be required before the valid data bits.

Data type	Values	Information
USINT	0 to 31	Number of leading zero bits

#### 4.10.5 NetTime of the current position

Name:

SSITimeValid

The NetTime of the current position is contained in this register.

For additional information about NetTime and timestamps, see ["NetTime Technology"](#) on page 23.

Data type	Values	Information
INT	-32768 to 32767	NetTime of the current position in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime of the current position in microseconds (32-bit)

#### 4.10.6 NetTime of the last position change

Name:

SSITimeChanged

The NetTime at which the last position change took place is contained in this register.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime of the last position change in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime of the last position change in microseconds (32-bit)

#### 4.10.7 Current position

Name:

SSIPosition

The current position transferred via the SSI interface is contained in this register.

Data type	Values	Information
INT	-32768 to 32767	Current position (16-bit)
UDINT	0 to 4,294,967,295	Current position (32-bit)
DINT	-2,147,483,648 to 2,147,483,647	

### 4.11 Counters

#### 4.11.1 Update cycle for counter values

Name:

CfO\_CounterCycleSelect

"Update cycle" in the Automation Studio I/O configuration.

The update cycle for the counter values is configured in this register.

#### Information:

The maximum counting frequency depends on this cycle. The module can process a maximum of 200 increments (edges) within one counter cycle.

Data type	Values	Information
USINT	2	System timer
	3	Prescaled system timer
	10	AOAI moment from X2X cycle
	14	SOSI moment from X2X cycle

#### 4.11.2 Counter mode

Name:

CfO\_CounterMode

"Counter mode" in the Automation Studio I/O configuration.

The counter mode is configured (see ["Counter modes" on page 21](#)) in this register.

Data type	Values	Information
USINT	0	Edge counter
	2	Up/Down counter
	3	Incremental encoder (AB counter)

### 4.11.3 Latch mode

Name:

CfO\_LatchMode

"Latch mode" in the Automation Studio I/O configuration.

The latch mode is configured in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	LatchMode	0	One-time
		1	Continuous
1 - 7	Reserved	-	

### 4.11.4 Latch comparators for counter inputs

Name:

CfO\_LatchComparator

"Latch level of channel 0x" in the Automation Studio I/O configuration.

The latch comparators for the counter inputs are configured in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Comparison level for the latch comparator on input 1	0	LOW
		1	HIGH
1	Comparison level for the latch comparator on input 2	0	LOW
		1	HIGH
2	Comparison level for the latch comparator on input 5	0	LOW
		1	HIGH
3	Comparison level for the latch comparator on input 6	0	LOW
		1	HIGH
4	Enables the latch comparator on input 1	0	Disabled
		1	Enabled
5	Enables the latch comparator on input 2	0	Disabled
		1	Enabled
6	Enables the latch comparator on input 5	0	Disabled
		1	Enabled
7	Enables the latch comparator on input 6	0	Disabled
		1	Enabled

### 4.11.5 Clearing counter values and enabling latching

Name:

CounterReset

LatchEnable

Counter values can be deleted or the latch enabled using this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	CounterReset	0	No action
		1	Delete counter
1	LatchEnable	0	Disabled
		1	Enabled
2 - 7	Reserved	-	



#### 4.11.6 Counter for latch events

Name:

LatchCount

Latch events that occur are counted in this register. This counter can be used to detect whether a new value has been latched, for example.

Data type	Values	Information
SINT	-128 to 127	Latch counter

#### 4.11.7 NetTime of the current counter value

Name:

CounterTimeValid

The X2X NetTime of the current counter value is contained in this register.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime of the current counter value in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime of the current counter value in microseconds (32-bit)

#### 4.11.8 NetTime of the last counter value change

Name:

Counter01TimeChanged to Counter02TimeChanged

The NetTime at which the last change of the respective counter took place is contained in this register.

For additional information about NetTime and timestamps, see ["NetTime Technology" on page 23](#).

Data type	Values	Information
INT	-32768 to 32767	NetTime of the last change of the respective counter in microseconds (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	NetTime of the last change of the respective counter in microseconds (32-bit)

#### 4.11.9 Current counter value

Name:

CounterValue01 to CounterValue02

The current counter value of the respective counter is contained in this register.

Data type	Values	Information
INT	-32768 to 32767	Counter value of the respective counter (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Counter value of the respective counter (32-bit)

#### 4.11.10 Latch counter

Name:

CounterLatch01 to CounterLatch02

As soon as the latch conditions set in register ["CfO\\_LatchComparator" on page 48](#) are met, the content of the relevant ["CounterValue register" on page 49](#) is copied to this register.

Data type	Values	Information
INT	-32768 to 32767	Latch counter (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Latch counter (32-bit)

#### 4.11.11 Counter value relative to the last latch

Name:

CounterRel01 to CounterRel02

In this register, the counter value of the respective counter is calculated relative to the last latch of the respective counter.

Data type	Values	Information
INT	-32768 to 32767	Counter value relative to the last latch (16-bit)
DINT	-2,147,483,648 to 2,147,483,647	Counter value relative to the last latch (32-bit)

## 4.12 Error handling

### 4.12.1 Error state - Output data and edge detection

Name:

OutputControlError

OutputCopyError

EdgeDetectError

Data output errors and cycle time setting errors are indicated in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Reserved	-	
4	OutputControlError	0	No error
		1	The module did not receive new data in time when "Output control mode = Single", meaning that a bit that has already been output would have been output again by the output control buffer.
5	OutputCopyError	0	No error
		1	Oversampling output data could not be copied to the output control buffer (attempted to write to an address outside the <a href="#">oversample output window</a> , for example).
6	EdgeDetectError	0	No error
		1	Edge detection cycle time violation: "EdgeDetectPollCycle" must be $\leq 255 \mu\text{s}$ . This error occurs if the cycle set in register "CfO_EdgeDetectPollCycleID" on <a href="#">page 36</a> is $> 255 \mu\text{s}$ .
7	Reserved	-	

### 4.12.2 Error state - SSI

Name:

SSICycleTimeViolation

SSIParityError

SSI interface errors are indicated in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	SSICycleTimeViolation	0	No error
		1	An error occurred - Possible causes: <ul style="list-style-type: none"> <li>SSI transfer takes longer than the set "update cycle".</li> <li>Monostable multivibrator testing is enabled, and the SSI data line does not take on the defined level at the end of the transfer.</li> </ul>
1	SSIParityError	0	No error
		1	SSI parity error
2 - 7	Reserved	-	

### 4.12.3 Error state - Motion functions

Name:

MovFifoEmpty

MovFifoFull

MovTargetTimeViolation

MovMaxFrequencyViolation

Motion function errors are indicated in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	MovFifoEmpty	0	No error
		1	The position/timestamp FIFO buffer is empty.
1	MovFifoFull	0	No error
		1	The position/timestamp FIFO buffer has overshot the size set in register "FifoSize" on page 39.
2	MovTargetTimeViolation	0	No error
		1	This occurs if the moment set in register "MovTargetTime" on page 44 is already in the past.
3	MovMaxFrequencyViolation	0	No error
		1	The maximum output frequency setpoint has overshot the maximum frequency set in register "CfO_SpeedLimit" on page 40.
4 - 7	Reserved	-	

### 4.12.4 Acknowledging error messages - Output data and edge detection

Name:

QuitOutputControlError

QuitOutputCopyError

QuitEdgeDetectError

Error messages from register "Error state - Output data and edge detection" on page 50 can be acknowledged by setting the corresponding bits in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Reserved	-	
4	QuitOutputControlError	0	No change
		1	Acknowledge error
5	QuitOutputCopyError	0	No change
		1	Acknowledge error
6	QuitEdgeDetectError	0	No change
		1	Acknowledge error
7	Reserved	-	

### 4.12.5 Acknowledging error messages - SSI

Name:

SSIQuitCycleTimeViolation

SSIQuitParityError

Error messages from register "Error state - SSI" on page 50 can be acknowledged by setting the corresponding bits in this register.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	SSIQuitCycleTimeViolation	0	No change
		1	Acknowledge error
1	SSIQuitParityError	0	No change
		1	Acknowledge error
2 - 7	Reserved	-	

#### 4.12.6 Acknowledging error messages - Motion functions

Name:

MovQuitFifoEmpty

MovQuitFifoFull

MovQuitTargetTimeViolation

MovQuitMaxFrequencyViolation

In this register, error messages from register ["Error state - Motion functions" on page 51](#) can be acknowledged by setting the corresponding bits.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	MovQuitFifoEmpty	0	No change
		1	Acknowledge error
1	MovQuitFifoFull	0	No change
		1	Acknowledge error
2	MovQuitTargetTimeViolation	0	No change
		1	Acknowledge error
3	MovQuitMaxFrequencyViolation	0	No change
		1	Acknowledge error
4 - 7	Reserved	-	

#### 4.13 Minimum X2X cycle time

The minimum X2X cycle time is strongly dependent on the configured functions and the resulting load on the module. Setting "Fast reaction" and a very short system cycle (<50 µs) generally have a negative effect on the minimum X2X cycle time. This can result in error behavior with short X2X cycle times.