# XMC4000 / XMC1000

Microcontroller Series
for Industrial Applications

## Universal Serial Interface Channel (USIC)

Device Guide

V1.0 2014-04

Microcontrollers

**Edition 2014-04**

**Published by**
**Infineon Technologies AG**
**81726 Munich, Germany**



### Legal Disclaimer

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation, warranties of non-infringement of intellectual property rights of any third party.

### Information

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (**www.infineon.com**).

### Warnings

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office.

Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

## Trademarks of Infineon Technologies AG

AURIX™, C166™, CanPAK™, CIPOS™, CIPURSE™, EconoPACK™, CoolMOS™, CoolSET™, CORECONTROL™, CROSSAVE™, DAVE™, DI-POL™, EasyPIM™, EconoBRIDGE™, EconoDUAL™, EconoPIM™, EconoPACK™, EiceDRIVER™, eupec™, FCOS™, HITFET™, HybridPACK™, I²RF™, ISOFACE™, IsoPACK™, MIPAQ™, ModSTACK™, my-d™, NovalithIC™, OptiMOS™, ORIGA™, POWERCODE™; PRIMARION™, PrimePACK™, PrimeSTACK™, PRO-SIL™, PROFET™, RASIC™, ReverSave™, SatRIC™, SIEGET™, SINDRION™, SIPMOS™, SmartLEWIS™, SOLID FLASH™, TEMPFET™, thinQ!™, TRENCHSTOP™, TriCore™.

## Other Trademarks

Advance Design System™ (ADS) of Agilent Technologies, AMBA™, ARM™, MULTI-ICE™, KEIL™, PRIMECELL™, REALVIEW™, THUMB™, µVision™ of ARM Limited, UK. AUTOSAR™ is licensed by AUTOSAR development partnership. Bluetooth™ of Bluetooth SIG Inc. CAT-iq™ of DECT Forum. COLOSSUS™, FirstGPS™ of Trimble Navigation Ltd. EMV™ of EMVCo, LLC (Visa Holdings Inc.). EPCOS™ of Epcos AG. FLEXGO™ of Microsoft Corporation. FlexRay™ is licensed by FlexRay Consortium. HYPERTERMINAL™ of Hilgraeve Incorporated. IEC™ of Commission Electrotechnique Internationale. IrDA™ of Infrared Data Association Corporation. ISO™ of INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. MATLAB™ of MathWorks, Inc. MAXIM™ of Maxim Integrated Products, Inc. MICROTEC™, NUCLEUS™ of Mentor Graphics Corporation. MIPI™ of MIPI Alliance, Inc. MIPS™ of MIPS Technologies, Inc., USA. muRata™ of MURATA MANUFACTURING CO., MICROWAVE OFFICE™ (MWO) of Applied Wave Research Inc., OmniVision™ of OmniVision Technologies, Inc. Openwave™ Openwave Systems Inc. RED HAT™ Red Hat, Inc. RFMD™ RF Micro Devices, Inc. SIRIUS™ of Sirius Satellite Radio Inc. SOLARIS™ of Sun Microsystems, Inc. SPANSION™ of Spansion LLC Ltd. Symbian™ of Symbian Software Limited. TAIYO YUDEN™ of Taiyo Yuden Co. TEAKLITE™ of CEVA, Inc. TEKTRONIX™ of Tektronix Inc. TOKO™ of TOKO KABUSHIKI KAISHA TA. UNIX™ of X/Open Company Limited. VERILOG™, PALLADIUM™ of Cadence Design Systems, Inc. VLYNQ™ of Texas Instruments Incorporated. VXWORKS™, WIND RIVER™ of WIND RIVER SYSTEMS, INC. ZETEX™ of Diodes Zetex Limited.

Last Trademarks Update 2011-11-11

# Revision History

| Page or Item | Subjects (major changes since previous revision) |
|---|---|
| **V1.0, 2014-04** | |
| | |
| | |
| | |

# Table of Contents

# 1 About this document

## 1.1 Scope and purpose

The Infineon Universal Serial Interface Channel (USIC) is a flexible interface module that covers several serial communication protocols.

Originally used in the Infineon XC2000/XE16x microcontroller family, the module has been extended to support the XMC product family with for example, Multi-IO SPI, a wider application field and easier implementation.

This document combines a brief overview of the module structure with a more detailed operation description of different USIC features.

### 1.1.1 Pre-requisites

The example code in this document can run on both the XMC4500 and XMC4400 starter kits, making use of the Infineon freeware tool DAVE™3, which integrates code generation, flash programming and debugging.

## 1.2 References

- For starter kits and DAVE™3, please visit: **www.infineon.com/XMC**.
- **AP1612900** describes initialization routines of UART applications such as LIN and IrDA mode of USIC module in XC2000/XE166.

# 2 USIC Structure

Each USIC module has 2 channels and each channel has the same structure, consisting of:

- **Input stages** (data/clock/control input stage): DX0…DX5
- **Output signals** (data/clock/control signals): DOUT0...DOUT3, SCLKOUT, SELO[0..7], MCLKOUT
- **Baud rate generator**
- **Data shift unit** for data shifting and handling
- **Channel events and interrupt generation unit**
- **FIFO structure** for data transmission and reception

## 1.1 Input stages

Each channel contains:

- Data input stages (DX0, DX3, DX4 and DX5)
- 1 clock input stage (DX1)
- 1 control input stage (DX2)

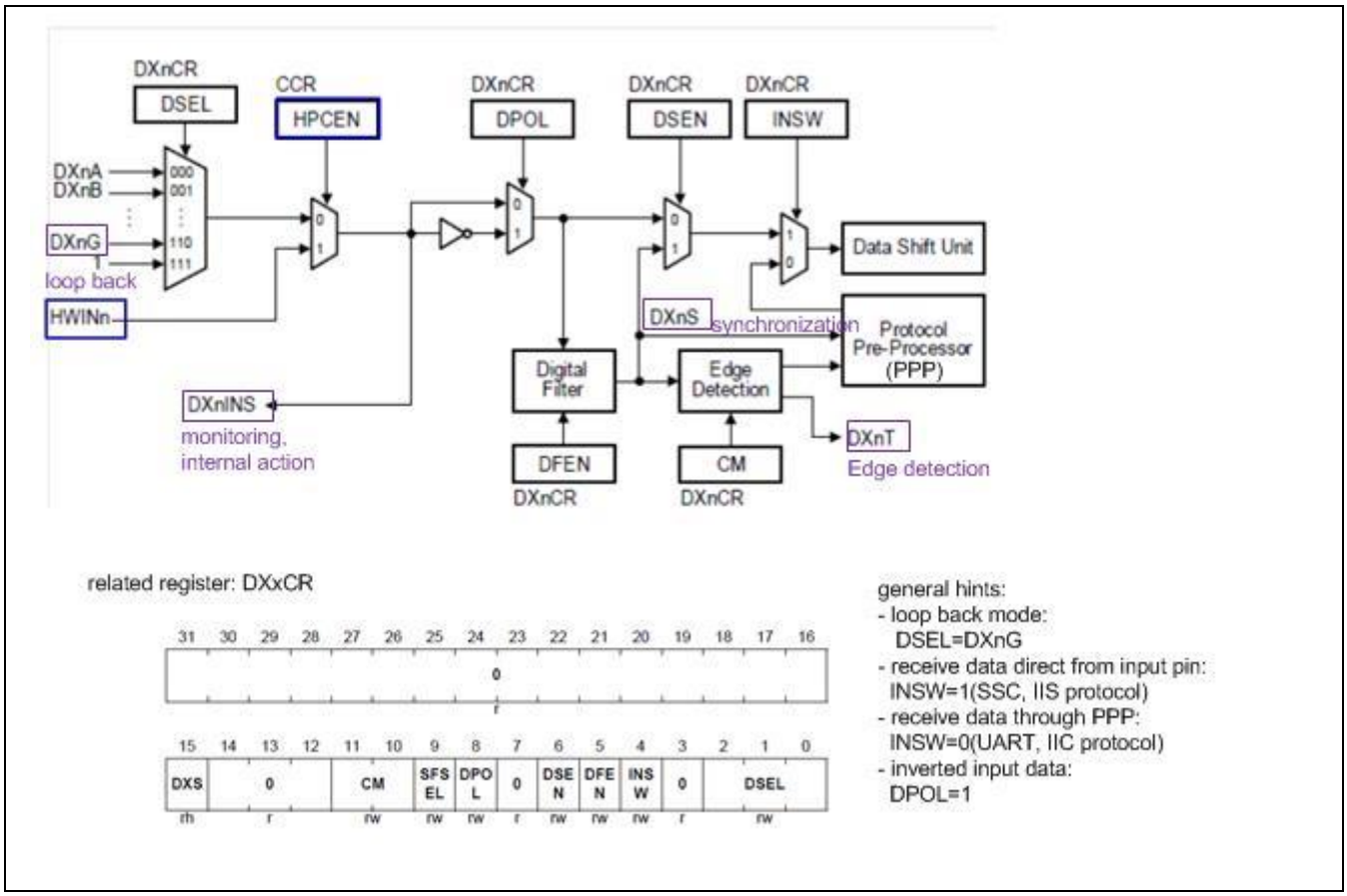


**Figure 1 Input stage for DX0 and DX[5:3] (Data Input Stage)**

### Notes

- In contrast to the data input stage DX0, DX3...DX5, there are no hardware pins (HWINn) on the clock input (DX1) and control input stage (DX2).
- HWINn and HPCEN are only available on the 4 data input stages.
- DX3, DX4, DX5 support multiple data input/output SSC applications, such as the dual and quad-SSC.

- HWINn and HPCEN are implemented in the XMC family of products.
- CCR.HPCEN enables the hardware port control for quick data exchange, and it also allows the USIC pins to directly drive complex control and communication patterns without further software interaction with the ports. CCR.HPCEN is not installed on the clock (DX1) and the control (DX2) input stage.
- The number of input signals used depends on the selected protocol and application mode. For example, UART only uses DX0 (as Rx) line, while DX1 can be optionally used for collision detection.

### 2.1.1 Typical application use cases

#### Loop-back mode

In ASC protocol loop-back mode (only DX0 is used):

- DX0CR.DSEL= DX0G and DX0CR.INSW=0

In SSC protocol loop-back mode (master mode):

- XMC1000 family
  - the data line loop-back mode: DX0CR.DSEL= DX0G and DX3CR.DSEL= DX3G
  - the clock line loop-back mode: DX1CR.DSEL= DX1G and DX4CR.DSEL= DX4G
  - the CS line loop-back mode: DX2CR.DSEL= DX2G and DX5CR.DSEL= DX5G
- XMC4000 family
  - the data line loop-back mode: DX0CR.DSEL= DX0G
  - the clock line loop-back mode: DX1CR.DSEL= DX1G
  - the CS line loop-back mode: DX2CR.DSEL= DX2G

*Note: There is no loop-back mode for the IIC protocol*

#### PPP or Input signal direct modes

Selects the input data direct mode (DXxCR.INSW=$1_b$) or the output of the Protocol Pre-Processor (PPP) (DXxCR.INSW=0b).

- UART mode: DX0CR.INSW=$0_b$ (PPP is used)
- SSC master mode: DX0CR.INSW=$1_b$ (data input), DX1 and DX2 are not used
- SSC slave mode: DX0CR.INSW=$1_b$ (data input), DX1CR.INSW=$1_b$ (clock input), DX2CR.INSW=$1_b$ (CS input)
- IIC master/slave mode: DX0CR.INSW=$0_b$, DX1CR.INSW=$0_b$ (both input stages use the PPP)
- IIS master mode: like SSC mode
- IIS slave mode: like SSC mode

If the input signal is used (INSW=$1_b$), then:

- the edge can be defined as a trigger signal (via DXxCR.CM)
- a digital filter can be used (via DXxCR.DFEN)
- data synchronization can be enabled (via DXxCR.DSEN)

#### Invert the input signal (via bit DPOL)

In applications a ‘low’ active Chip Select (CS) line is normally used as the input signal for the slave SSC device, therefore its polarity must be inverted.

## 2.2 Output signals

For each protocol up to 14 output signals are available:

- Data output: DOUT0...DOUT3,
- Clock output: SCLKOUT, MCLKOUT
- Control output: SELO[7..0]

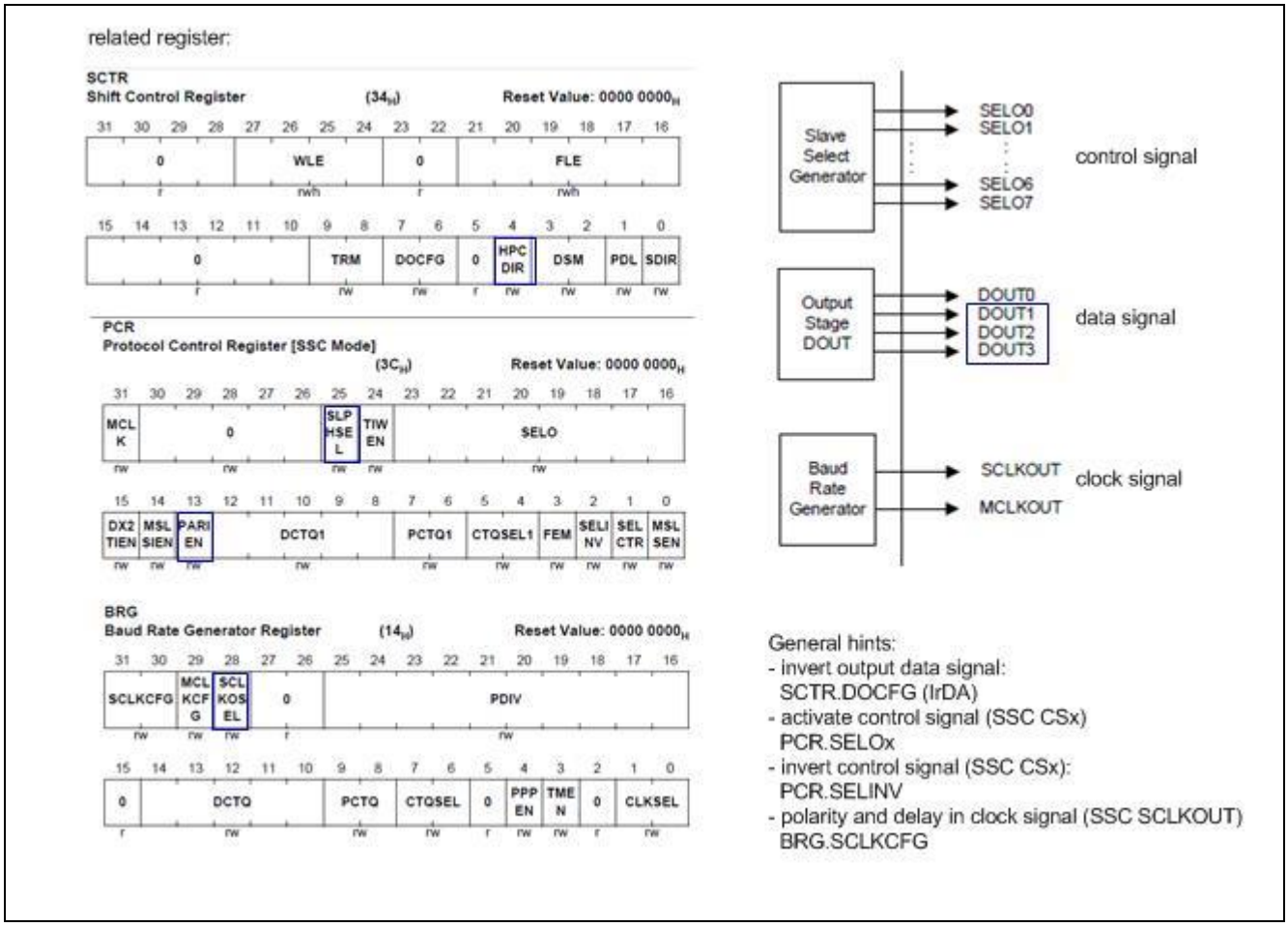


**Figure 2 Output Stage**

The number of outputs actually used depends on the selected protocol.

- UART mode: DOUT0 data output
- SSC master mode: DOUT0…DOUT3, SCLKOUT, SELO[7:0], CLKOUT optional
- SSC slave mode: DOUT0...DOUT3,
- IIC master/slave mode: DOUT0, SCLKOUT
- IIS master mode: DOUT0, SCLKOUT, SELO[7:0]
- IIS slave mode: DOUT0

*Note: Data output line DOUT1...DOUT3 is implemented in the XMC family of products to support multiple data input/output SSC applications, such as the dual and quad-SSC.*

### Output stage configuration options

The polarity of the MCLKOUT can be configured via BRG.MCLKCFG.

- MCLKOUT has a fixed phase relation to the SCLKOUT. It is usually used in IIS communication as the master base clock in order to get a communication network with synchronized connections.

The polarity of the output signals can be inverted.

- Data output.
  - To generate a data signal for IrDA mode its polarity can be inverted via SCTR.DOCFG
- Clock output.
  - The polarity of the shift clock output signal SCLKOUT can be configured and a delay of one period of $f_{PDIV}$ (half SCLK period) can be created (BRG.SCLKCFG). Usually 4 different SSC shift clock output signals (SCLKOUT) are generated.
- Control output.
  - The polarity of the control signal SELOx. The output pin CS for the SSC device normally has an active 'low' level. In this instance the polarity of the SELO signal has been inverted by setting pin SELINV in the register PCR.

## 2.3 Baud Rate Generator

Baud rate generation is divided into the following parts:

- **Clock Input DX1 (optional)**
  - Usually in slave mode for baud rate generation, based on the external signal.
- **Fractional divider**
  - Generates baud rate based on system clock $f_{PB.}$
- **Protocol-related counters**
  - Time mode: contains PDIV divider and provides SCLK in SSC and generates $f_{CTQIN}$
  - Capture mode: counter for time interval measurement. For example, baud rate detection in LIN slave mode (BRG.TMEN=1)
- **Protocol Pre-Processor** (PPP)
  - Generate time quanta counter for one bit in UART / IIC (standard setting: $f_{CTQIN} = f_{FD}$ with CTQSEL = $00_b$)
  - Delay time configuration in SSC mode (standard setting: $f_{CTQIN} = f_{SCLK}$ with CTQSEL = $10_b$)
  - The system word length in IIS mode (standard setting: $f_{CTQIN} = f_{SCLK}$ with CTQSEL = $10_b$)

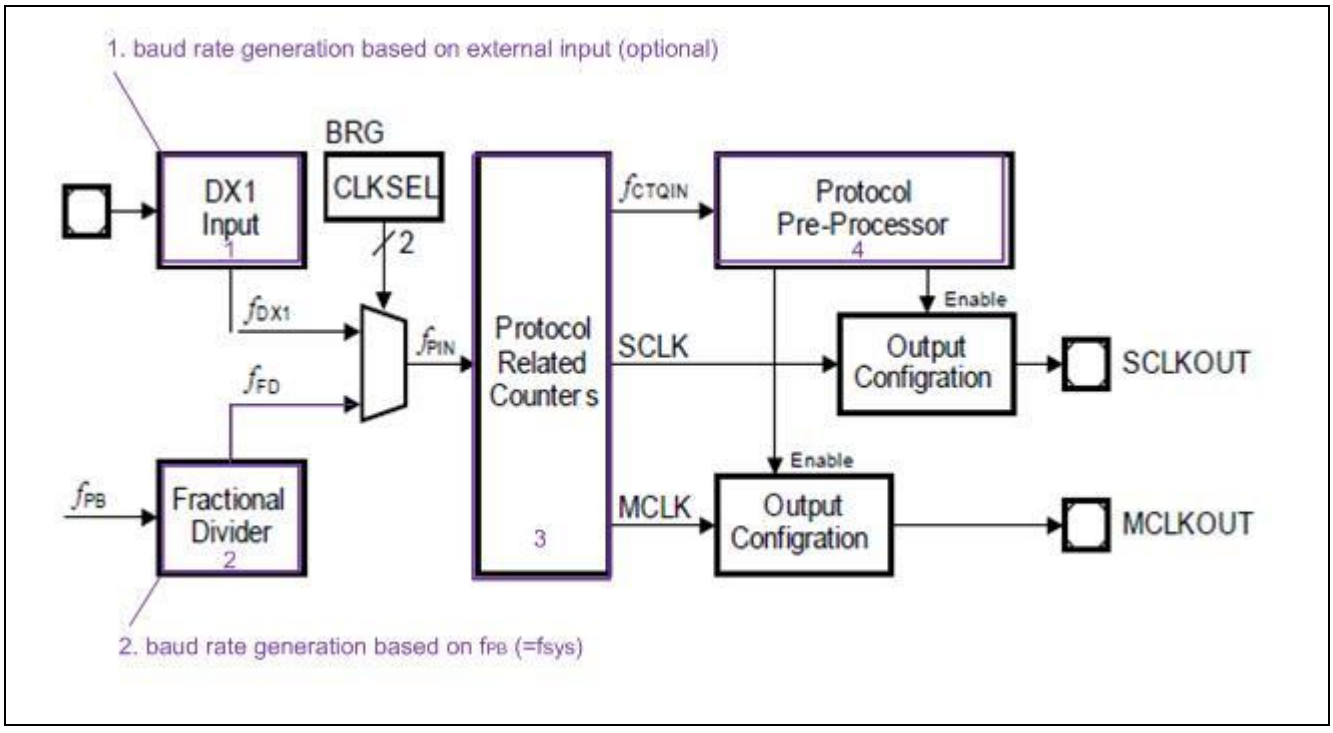


**Figure 3 Baud Rate Generator**

### 2.3.1 Clock Input DX1 (Optional)

The DX1 input stage is used for baud rate generation based on an external signal. It is normally used for slave mode.

An external input signal at the DX1 input stage can be optionally filtered and synchronized with $f_{PB}$($f_{sys}$).

- If BRG.CLKSEL=$10_b$, signal MCLK toggles with $f_{PIN}$.
  - The trigger signal DX1T determines $f_{DX1}$.
  - Both rising/falling edges of the input signal can be used for baud rate generation. The active edge is selected by bit field DX1CR.CM.
- If BRG.CLKSEL=$11_b$, $f_{PIN}$ is derived from the rising edges of DX1S.
  - The rising edges of the input signal can be used for baud rate generation.
  - The external signal is synchronized.
  - The rising edge of DX1S is used for the synchronization.

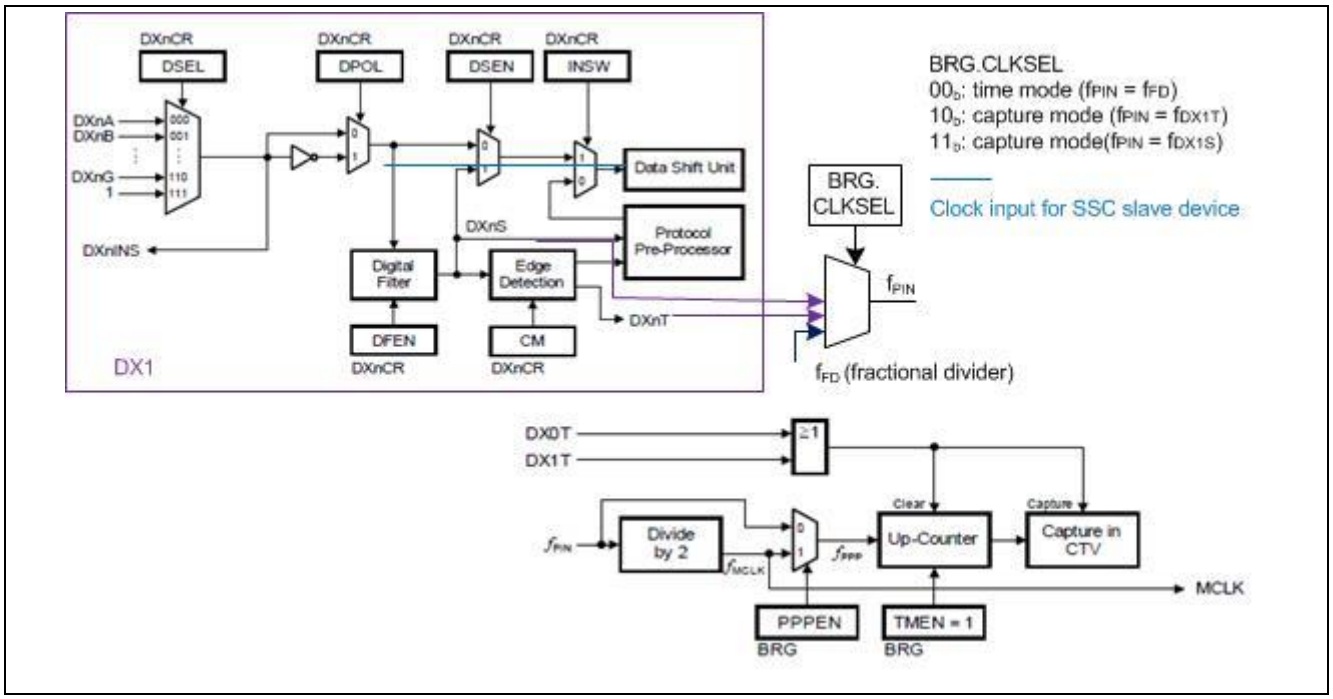


**Figure 4 Clock Input DX1**

### 2.3.2 Fractional Divider

If the fractional divider is used, then it holds $f_{PIN}=f_{PB}$ for baud rate generation based on $f_{PB}$.

There are two operation modes:

- Normal divider mode
  - In this mode (FDR.DM = $01_b$) it behaves like a reload counter (addition of +1) that generates an output clock on the transition from $3FF_H$ to $000_H$.
  - The bitfield RESULT represents the counter value, and STEP defines the reload value.
- Fractional divider mode
  - An output clock pulse at $f_{FD}$ is generated dependent on the result of the addition FDR.RESULT + FDR.STEP. If the addition leads to an overflow over $3FF_H$ a pulse is generated at $f_{FD}$.

#### Comparison of modes

The fractional divider mode provides the average output clock frequency with a higher accuracy than in normal divider mode, but $f_{FD}$ can have a maximum period jitter of one $f_{PB}(=f_{SYS})$ period.

The preference is to use normal divider mode for a higher baud rate.

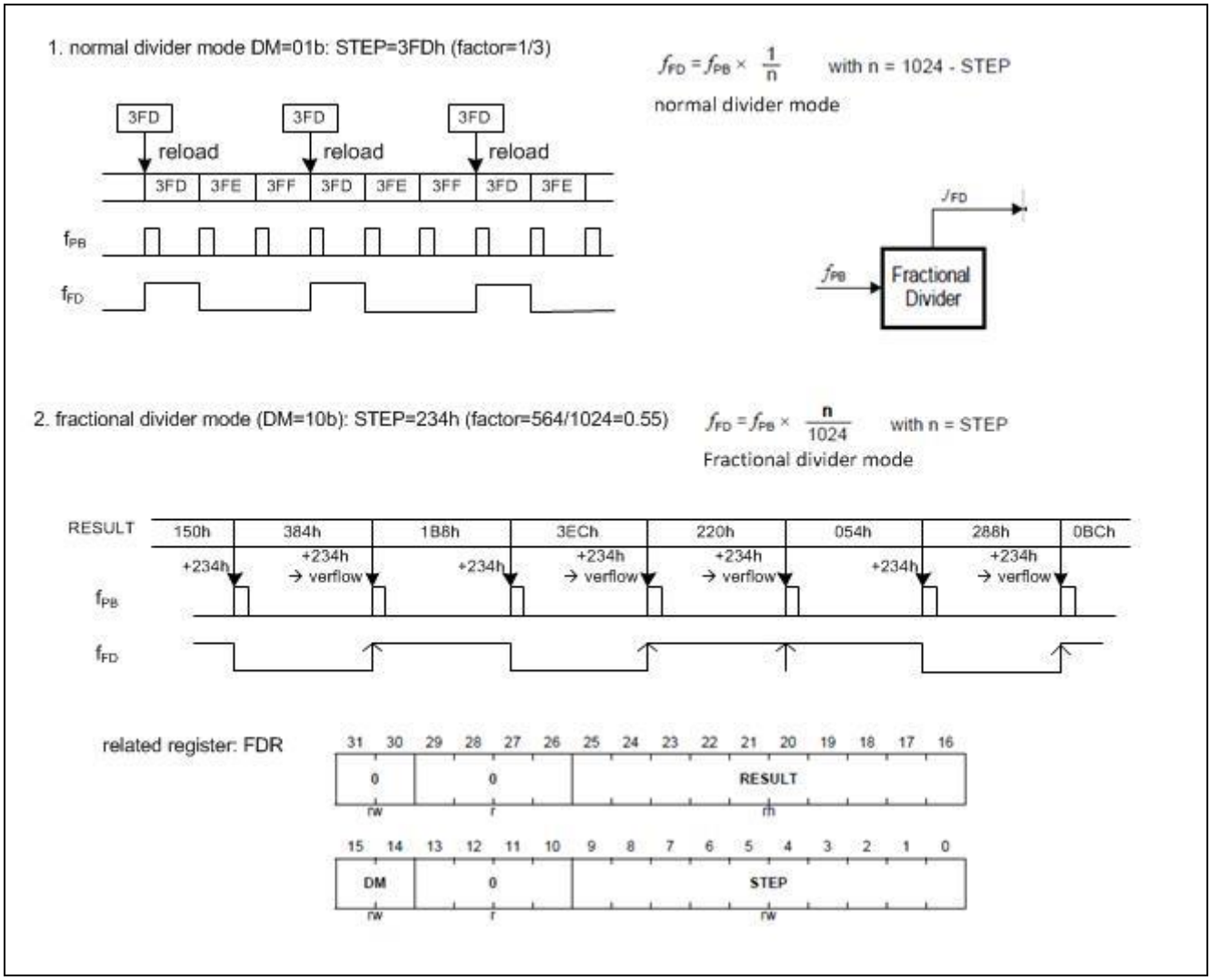


**Figure 5 Fractional Divider**

### 2.3.3 Protocol Related Counter

Protocol-related counters can be used in divider or capture mode. The counter contains a PDIV divider and generates $f_{CTQIN}$.

#### Divider mode

- PDIV divider
  - provides for example the shift clock SCLK, and MCLK in SSC (signal MCLK and SCLK have 50% duty cycle)
- $f_{CTQIN}$ generator, used for:
  - UART / IIC baud rate generation
  - Delay time configuration in SSC

The following figure illustrates divider mode being used to generate baud rate.

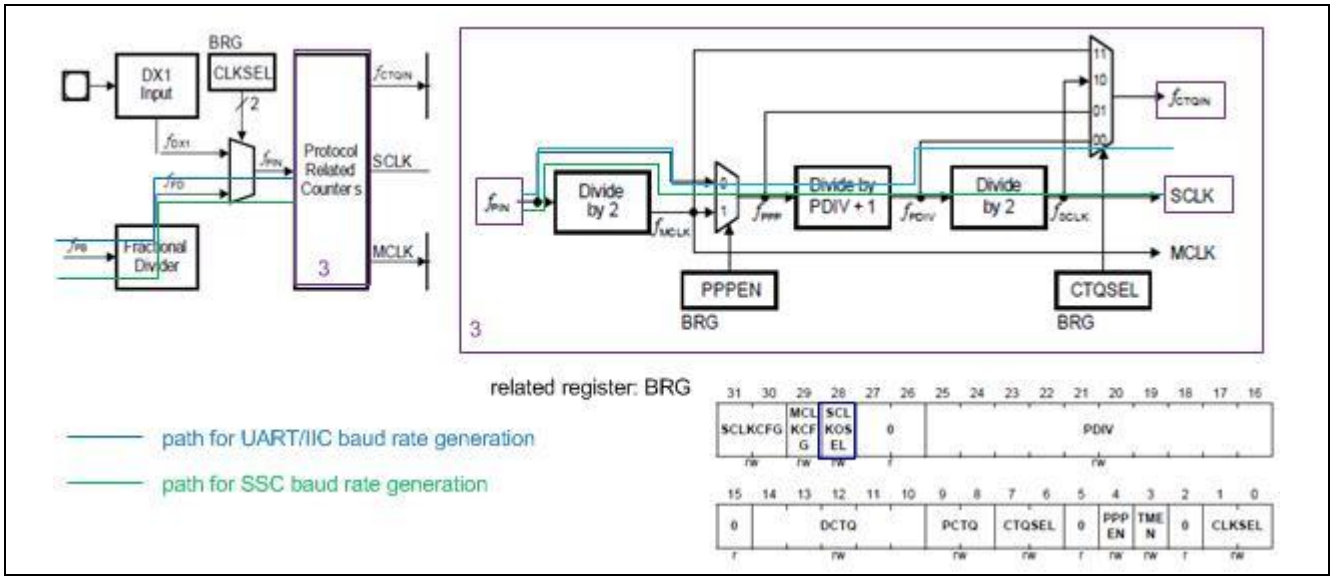


**Figure 6 Protocol-Related Counter in Divider Mode**

#### Software configuration for baud rate generation based on $f_{SYS}$

- $f_{FD}$=f($f_{SYS}$) via bits filed DM and STEP in register FDR
- $f_{PIN}$=$f_{FD}$ via bits CLKSEL=00b in register BRG
- $f_{PPP}$=$f_{PIN}$ or $f_{MCLK}$ via bit PPPEN in register BRG
- $f_{PDIV}$=f($f_{PPP}$) via bits field PDIV in register BRG
- select $f_{CTQIN}$ via bits field CTQSEL in register BRG
  - CTQSEL=$00_b$ -> $f_{CTRQ}$=$f_{PDIV}$
  - CTQSEL=$01_b$ -> $f_{CTRQ}$=$f_{PPP}$
  - CTQSEL=$10_b$ -> $f_{CTRQ}$=$f_{SCLK}$
  - CTQSEL=$11_b$ -> $f_{CTRQ}$=$f_{MCLK}$

SCLKOUT can take the transmit shift clock from the input stage DX1.

Selection is made through BRG.SCLKOSEL.

The slave has to setup the SCLKOUT pin function to output the shift clock by setting bit BRG.SCLKOSEL to 1, while the master has to set the DX1 pin function to receive the shift clock from the slave and enable the delay compensation with DX1CR.DCEN = 1 and DX1CR.INSW = 0 (see the errata issue “USIC_AI.008”).

### Capture mode

The protocol-related counter is used for internal time measurement (BRF.TMEN=1). For example, to measure the baud rate in slave mode before starting data transfers (the time between two edges of DX0T or DX1T), for example, baud rate detection in LIN slave mode.

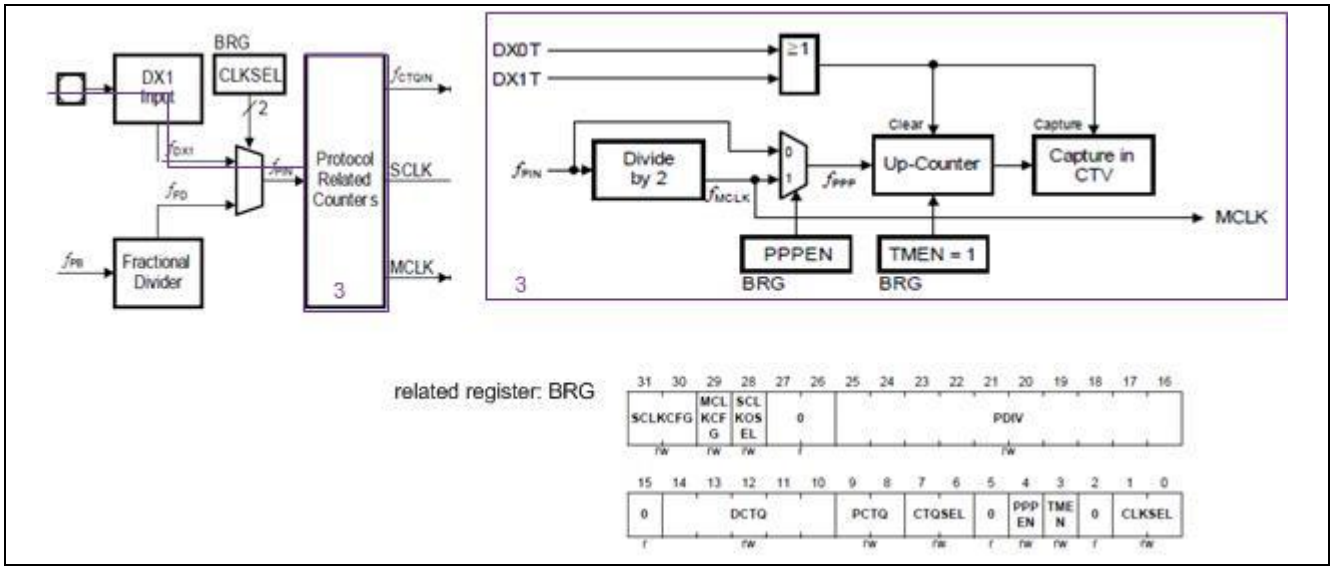


**Figure 7 Protocol-Related Counter in Capture Mode**

## 2.3.4 Protocol Pre-Processor

The Protocol Pre-Processor (PPP) is used to generate time intervals for protocol-specific purposes. It has a time quanta counter and is used for bit timing control. For example:

- PCTQ: pre-divider for time quanta counter (division of $f_{CTQIN}$ by 1,2,3 or 4)
- DCTQ: denominator for time quanta counter

Usually it generates:

- Time quanta counter for one bit in UART / IIC (normal setting: $f_{CTQIN} = f_{PDIV}$ with CTQSEL = $00_b$)
- Delay time configuration in SSC mode (normally $f_{CTQIN} = f_{SCLK}$ with CTQSEL = $10_b$)
- System word length in IIS mode (normally $f_{CTQIN} = f_{SCLK}$ with CTQSEL = $10_b$)

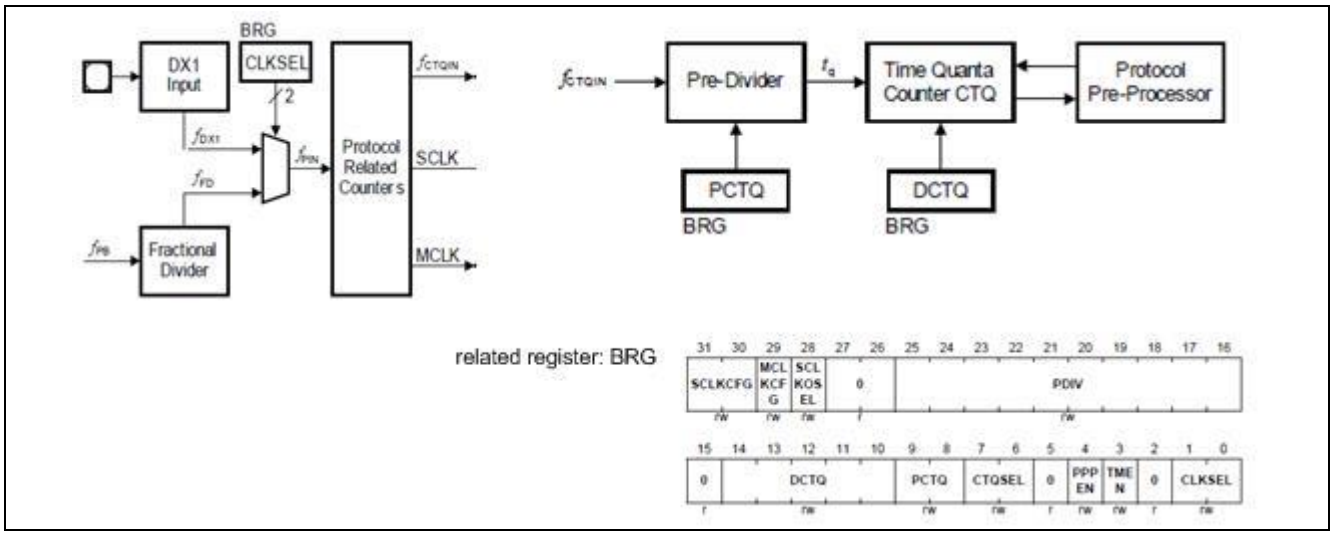


**Figure 8 Time Quanta Counter**

The PPP supports UART, SPI, IIC and IIS communication protocols:

**UART**

- Maximum frequency is $f_{SYS}$/4 (maximum module capability: DCTQ>=3)
- Number of data bits: 1 to 63
- PCTQ: the length of a time quantum (division of $f_{CTQIN}$ by 1, 2, 3, or 4)
- DCTQ: the number of time quanta per bit time. A standard setting is DCTQ+1=16 (sample point SP=8 or 9 (SP<DCTQ, recommended: DCTQ>=4)

**SPI**

- Module capability: maximum $f_{SYS}$/2
- Application target baud rate: ~15MBaud for both transmission and reception (Please refer to the appropriate data sheet for further details)
- Number of data bits: 1 to 63, >63 bits using explicit stop condition
- PCTQ: define the length of a time quantum for delay $T_{ld}$ and $T_{td}$
- DCTQ: the number of time quanta for the delay generation for $T_{ld}$ and $T_{td}$
- $T_{ld} = T_{td} = (PCTQ+1)x(DCTQ+1)/f_{CTQIN}$

**IIC**

- 7bit and 10bit addressing mode
- PCTQ: the length of a time quantum (division of $f_{CTQIN}$ by 1, 2, 3, or 4)
- DCTQ: the number of time quanta per bit time
- 100kBaud (PCR.STIM=0b): $f_{SYS}$>=2MHz, 1 symbol timing=10 tq (DCTQ=9)
- 400kBaud (PCR.STIM=1b): $f_{SYS}$>=10MHz, 1 symbol timing =25 tq (DCTQ=24)

**IIS**

- Module capability: maximum $f_{SYS}$/2
- Application target baud rate: ~30MBaud for transmission (Please refer to the appropriate data sheet for further details)

*Note: The module capability is considered only as transmission. The real baud rates that can be achieved in an application depend on the operating frequency of the device and the timing parameters (for example the setup and edge falling/rising time). If the filter structure is selected in the input stage of USIC, it will have an additional delay. Refer to the appropriate data sheet for further details.*

## 2.4 Data Shifting and Handling

Data shift and handling is based on:

- Tx/Rx buffer structure.
- The data shift mode control (single/dual/quad data shift mode, data/frame length, passive level, and so on) for Tx/Rx process.
- The transmit control and status information (start/end of frame control, TCI info, dynamic control…).
- Transmit handling (data valid control, transfer trigger logic, transfer gating logic, data transfer functionality like single-shot mode, for example valid data will be sent only one time).

### 2.4.1 Transmit and Receive Buffering

#### Transmit

TBUF is the internal shift register. It cannot be directly accessed by software. Data words can be written into one of the transmit buffer input locations TBUFx (x = 00…31).

TBUFx has a total of 32 consecutive addresses. It contains 5-bit wide TCI information (see section 2.4.3) and can be used for control mode. If transmit FIFO is enabled, then you can also write data words into Inx (x = 00…31).

#### Receive

For the receive process in the data shift unit, a double receive buffer structure (RBUF0, RBUF1) is implemented in USIC. This supports the reception of data streams longer than 16-bit word and USIC handles the reception sequence of both internal receive buffers. To read data out, always use register RBUF except when receive FIFO is used, when register OUTR should be used instead.

*Note:*

1. *To enable Tx/Rx FIFO bits TBCTR/RBCTR.SIZE (buffer size) must be set to >0.*
2. *During the initialization phase, the start entry of a FIFO buffer has to be defined by writing the number of the first FIFO buffer entry in the FIFO buffer to the corresponding bit field DPTR in register RBCTR/TBCTR, with the related bitfields RBCTR.SIZE=0 and TBCTR.SIZE=0.*
3. *DO NOT initialize bitfield DPTR by SIZE!=0.*

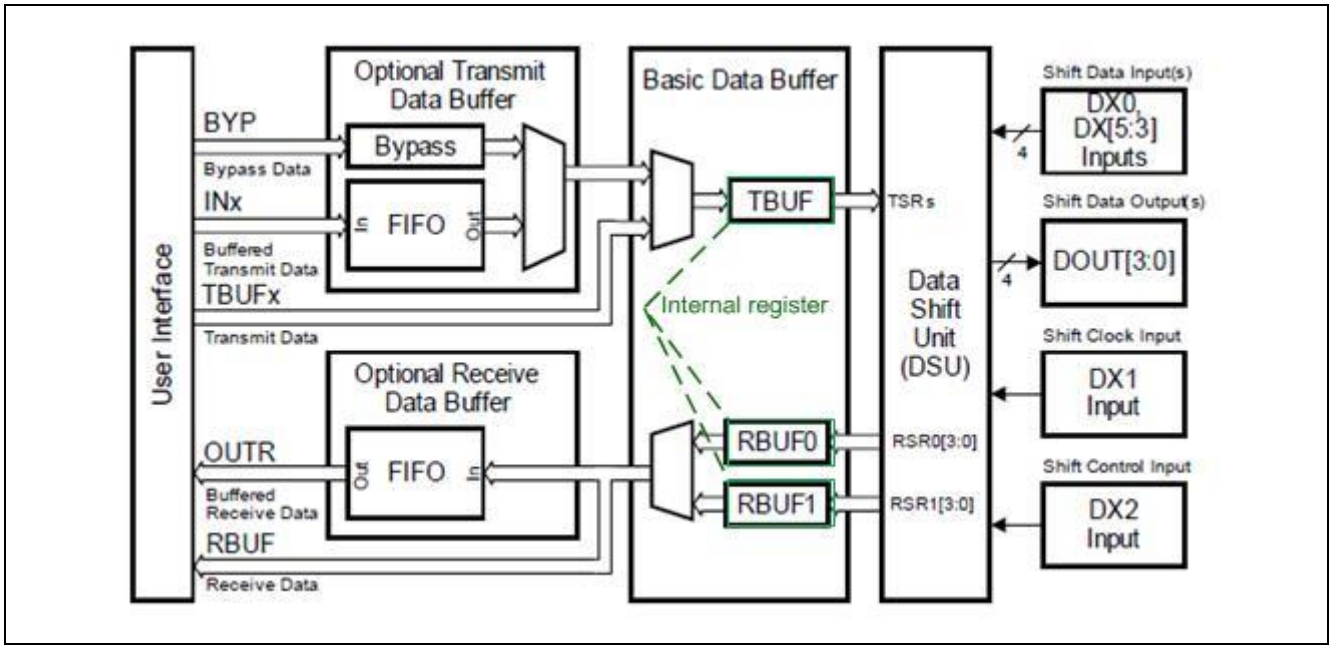


**Figure 9 Data Shift Unit**

### 2.4.2 Data Shift Control: Transmission/Receive Process (SCTR)

The default setting SCTR.DSM=00b means that TSR is used for the transmission path, and RSR00 (for RBUF0) and RSR10 (for RBUF1) are used for the receive path for all data bits.

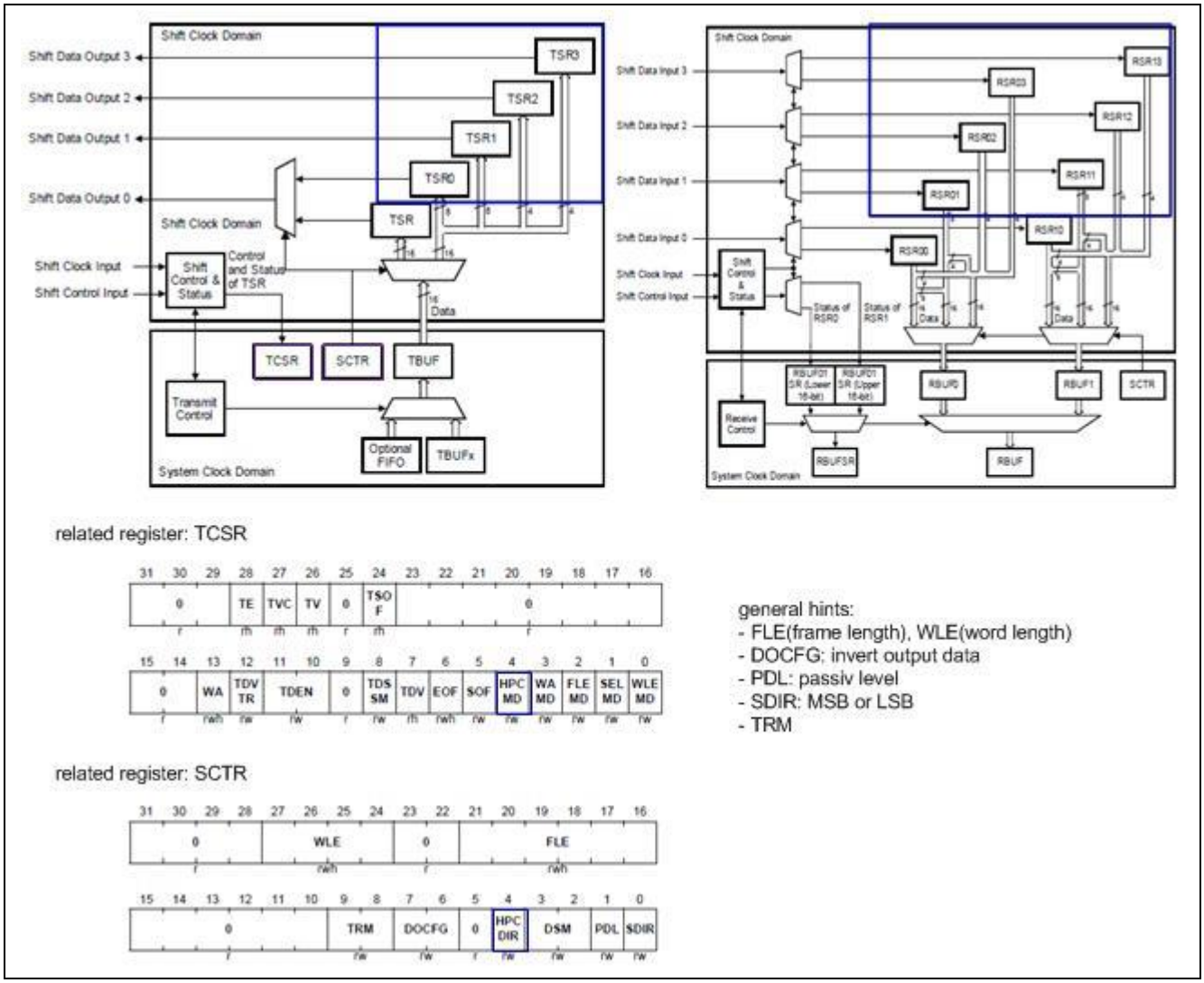


**Figure 10 Data Shift Control: Tx/Rx Process Control/Status Information**

In the XMC family the data shift unit has 4 internal transmit shift registers (TSR0...TSR3) for operating the transmit data path and 6 internal receive shift registers (RSR0[1…3] for RBUF0; RSR1[1...3] for RBUF1) for operating the receive data path.

The transmit shift data can be selected (SCTR.DSM) to be shifted out one, two or four bits at a time through the corresponding number of output lines. This option allows the USIC to support protocols such as the dual and quad-SSC. Selection is made through the TDSM bitfield in the shift control register. This configuration is also available in the receive process.

### Frame and Word length

- ASC
  - FLE=0...62 (63 is not allowed), parity bit can be enabled via bitfield CCR.PM
- SSC
  - FLE=0…63 (FLE=63 for frames with more than 63 data bits, see section 3.3.1), parity bit can be enabled via bitfield CCR.PM
- IIC
  - for 7-bit addressing: WLE=7, unlimited data flow (SCTRH.FLE=$3FF_H$)
- IIS
  - frame length <= system word length

### Shift control signal (TRM)

- ASC
  - TRM= $01_b$, the shift control signal is active if it is at 1-level
- SSC
  - TRM=$01_b$, the shift control signal is active if it is at 1-level
- IIC
  - TRM=$11_b$, active without referring to the actual signal level
- IIS
  - TRM=$11_b$, active without referring to the actual signal level

### Data Output Configuration (DOCFG)

- ASC
  - DOCFG=$00_b$ (DOCFG=$01_b$ for IrDA signal, the DOUT value is then inverted)
- SSC
  - DOCFG=$00_b$, the DOUT value not inverted
- IIC
  - DOCFG=$00_b$, the DOUT value not inverted
- IIS
  - DOCFG=$00_b$

### Passive Data Level (PDL)

- ASC
  - PDL=$1_b$, the passive data level=1
- SSC
  - PDL=$1_b$, the passive data level=1
- IIC
  - PDL=$0_b$, the passive data level=0
- IIS
  - PDL=$1_b$, the passive data level=1

### Shift Direction control (SDIR)

- ASC
  - SDIR=$0_b$, the LSB first
- SSC
  - SDIR=$1_b/0_b$, the MSB/LSB first
- IIC
  - SDIR=$1_b$, the MSB first
- IIS
  - SDIR=$1_b$, the MSB first

## 2.4.3 Transmit Shift Control Information (for Tx Process)

The control bit in the TCSR register defines data control in the transmission process. For example, if SOF is set, then the content of TBUF is transferred as the first Word of a new frame.

The 5-bit TCI value derived from the address of TBUFx or INx (x=0…31) can be used as an additional control parameter in data transfers.

- CSx control mode: TCSR. SELMD = 1. See section **3.6.**
- Word length control mode: TCSR. WLEMD = 1

**Table 1 Word length control: TCSR.WLEMD = 1**

| Write to TBUFx/INx | TCI[4]-[3..0] | TCSR.EOF- SCTR.WLE | |
|---|---|---|---|
| TBUF31/IN31 | 1-1111b | 1-1111b | EOF=1, 16 bits WORD |
| TBUF15/IN15 | 0-1111b | 0-1111b | EOF=0, 16 bits WORD |
| TBUF23/IN23 | 1-0111b | 1-0111b | EOF=1, 8 bits WORD |
| TBUF07/IN07 | 0-0111b | 0-0111b | EOF=0, 8 bits WORD |
| …… | …….. | …….. | ……. |

- Frame length control mode: TCSR. FLEMD = 1

**Table 2 Frame length control: TCSR. FLEMD= 1**

| Write to TBUFx/INx | TCI[4..0] | SCTR.FLE | |
|---|---|---|---|
| TBUF31/IN31 | 11111b | 31 | Frame length=32 bits |
| TBUF15/IN15 | 01111b | 15 | Frame length=16 bits |
| TBUF07/IN07 | 00111b | 7 | Frame length=8 bits |
| …… | …….. | …….. | ……. |

- Word access control mode: TCSR.WAMD = 1

**Table 3 Word access control: TCSR. WAMD = 1 (IIS)**

| Write to TBUFx/INx | TCI[4] | TCSR.WA | |
|---|---|---|---|
| TBUF00/IN00…BUF15/IN15 | 1b | 1 | Right channel |
| TBUF16/IN16…BUF31/IN31 | 0b | 0 | Left channel |

- Hardware port control mode: TCSR.HPCMD=1

**Table 4 Hardware Port control: TCSR. HPCMD = 1 (See Example_5)**

| Write to TBUFx/INx | TCI[2]-TCI[1:0] | TCSR.HPDIR-SCTR.DSM | |
|---|---|---|---|
| TBUF07/IN07 | 1-11b | 1-11b | output, 4x data lines (DOUT0/1/2/3) |
| TBUF06/IN06 | 1-10b | 1-10b | output, 2x data lines (DOUT0/1) |
| TBUF04/IN04 | 1-00b | 1-00b | output, 1x data line (DOUT0) |
| TBUF03/IN03 | 0-11b | 0-11b | input, 4x data lines (DIN0/DIN3/4/5) |
| TBUF02/IN02 | 0-10b | 0-10b | input, 2x data lines (DIN0/DIN3) |
| TBUF00/IN00 | 0-00b | 0-00b | input, 1x data line (DIN0) |

*Note: To enable hardware port control the selected hardware pin of DX0/DOUT0, DX3/DOUT1, DX4/DOUT2 and DX5/DOUT3 must be switched on via CCR.HPCEN.*

### 2.4.4 Transmit Data Validation information (for Tx Process)

If TBUF data is set to the single short mode (TDSSM=1b), the data in TBUF is considered as invalid after it has been loaded into the shift register. TDEN must be set to 01b to allow data to be sent out from TBUF if TDV=1.

Bit TDV is hardware controlled. It is automatically set when data is moved to TBUF, by writing to one of the transmit buffers.

- ASC and IIC
  - TCSR.TVD is cleared in single short mode with the transmit buffer interrupt event (bit TBIF in register PSR)
- SSC and IIS
  - TCSR.TVD is cleared in single short mode with the receive start interrupt event (bit RSIF in register PSR)

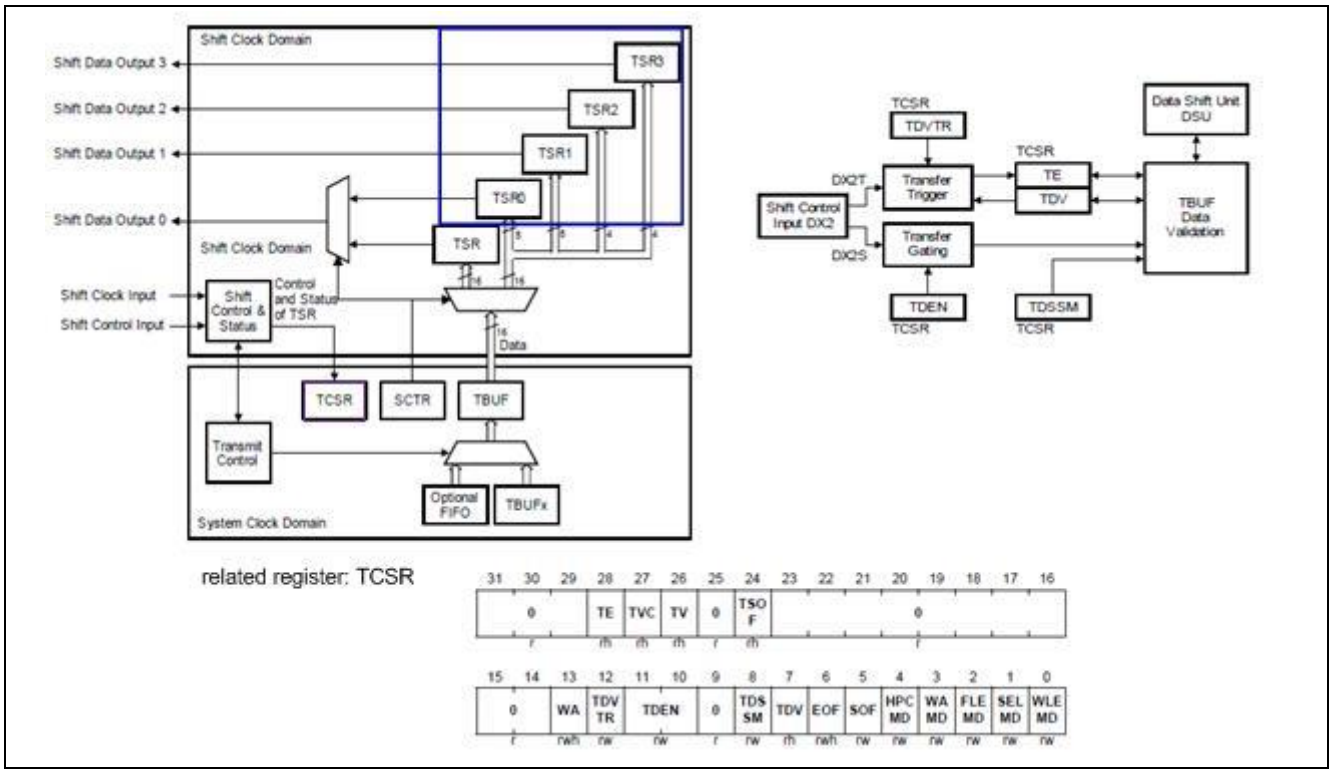


**Figure 11 Transmit Data validation**

## 2.5 Channel Events and Interrupt Generation Unit

Each USIC module provides 6 service request outputs, SRx (x = 0 to 5), which can be shared between its 2 channels.

In XMC4500 there are a total of 18 NVICs defined for 3 x USIC modules. For each channel there are 3 types of interrupt event available.

### 2.5.1 Data Transfer Events Related to Transmission / Reception

The interrupts listed in the following table are independent of the selected protocol.

The following sequences should be executed for initialization:

- Register CCR defines the general interrupt generation
- Register PSR contains indication flags
- Write a 1 to the corresponding bit position in register PSCR to clear its status bit in PSR
- The bitfields of register INPR define which SRx will be activated if the corresponding event occurs, for each USIC module a total of 6 interrupt service request output are defined.
- The interrupt priority level and enable/disable are controlled by Nested Vectored Interrupt Control (NVIC) unit in XMC, please see RM for more details about CMSIS functions to access ARM Cortex-M4 NVIC.

**Table 5 Transmit / Receive interrupts**

| Flag | Indication | Enable / SRx selected by | Remark |
|---|---|---|---|
| PSR.TBIF | Transmit buffer event | CCR.TBIEN / INPR.TBINP | ASC/I2C: TCSR.TDV is cleared with this event |
| PSR.RSIF | Receive start event | CCR.RSIEN / INPR.*TBINP* | SRx for RSIF interrupt is shared with TBIF |
| PSR.TSIF | Transmit shift interrupt | CCR.TSIEN / INPR.TSINP | SSC/I2S: TCSR.TDV is cleared with this event |
| PSR.RIF | Standard receive event | CCR.RIEN / INPR.RINP | |
| PSR.AIF | Alternative receive event | CCR.AIEN / INPR.AINP | |
| PSR.DLIF | Data lost event | CCR.DLIEN / INPR.*PINP* | SRx for DLIF interrupt is shared with Protocol-Specific Interrupt |
| PSR.BRGIF | Baud Rate generator Indication | CCR.BRGIEN / INPR.*PINP* | SRx for BRGIF interrupt is shared with Protocol-Specific Interrupt |

### 2.5.2 Protocol-Specific Interrupts

Register PCR defines protocol-specific interrupts.

- Register bit ield INPR.PINP defines which SRx will be activated if the corresponding event occurs
- Register PSR contains indication flags
- Write a 1 to the corresponding bit position in register PSCR to clear its status bit in PSR
- The interrupt priority level and enable/disable are controlled by Nested Vectored Interrupt Control (NVIC) unit in XMC, please see RM for more details about CMSIS functions to access ARM Cortex-M4 NVIC.

**Table 6 Protocol-specific interrupts**

| Flag | Indication | Enable / SRx selected by | Remark |
|---|---|---|---|
| **UART** | | | |
| PSR.TFF | Transmitter frame finished | PCR.FFIEN | |
| PSR.RFF | Receiver frame finished | | |
| PSR.COL | Collision detection | PCR.CDEN | Can be used in Half-Duplex mode |
| PSR.SBD | Synchronization break detection | PCR.SBIEN | Used in LIN |
| PSR.RNS | Receiver noise detection | PCR.RNIEN | Independent of 1-bit or 3-bits sample mode (via PCR.SMD) |
| PSR.FER0 | Format error 0 | PCR.FEIEN | |
| PSR.FER1 | Format error 1 | | In ‘the two stop bits mode” and a ‘0’ has been latched at 2nd stop bit |
| **SSC** | | | |
| PSR. MSLSEV | start and/or stop of MSLS(CS output) | PCR.MSLSEN | In master mode<br>PSR.MSLS: MSLS current status<br>(polarity of SELOx via PCR.SELINV) |
| PSR. DX2TEV | Rising and/or falling edge of DX2 (SELIN input) | PCR.DX2TIEN | In slave mode<br>PSR.DX2S: DX2S current status<br>(polarity of DX2 via DX2CR.DPOL) |
| RBUFSR.PAR | Parity error | PCR.PARIEN | |
| **IIC** | | | |
| PSR.SCR | Start condition received | PCR.SCRIEN | |
| PSR.PCR | Stop condition received | PCR.PCRIEN | |
| PSR.RSCR | Repeated start condition | PCR.RSCRIEN | |
| PSR.SRR | Slave read requested | PCR.SRRIEN | Only in master read - slave transmit mode (slave device) |
| PSR.ARL | Master arbitration lost | PCR.ARLIEN | For each bit during data and address transmission |
| PSR.ACK | Acknowledge received | PCR.ACKIEN | Only in master device, after address has been acknowledged or data has been received |
| PSR.NACK | Non-acknowledge received | PCR.NACKIEN | Only in master device with wrong address |
| PSR.ERR | Start/Stop condition in wrong position | PCR.ERRIEN | |
| PSR.TDF | TDF error | PCR.ERRIEN | Wrong TDF, undefined TDF |
| | | | |

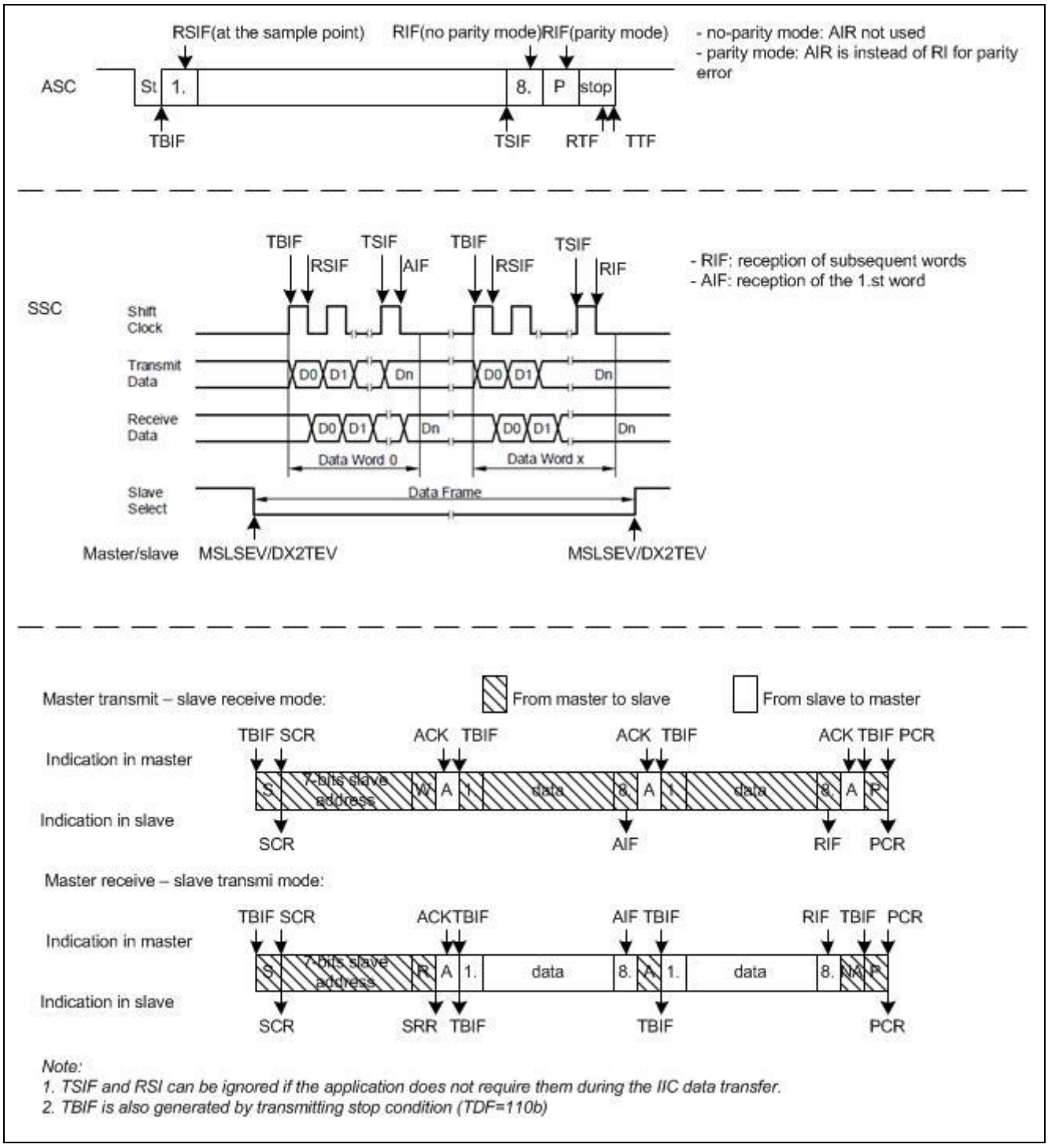


**Figure 12 Channel Events and Interrupt Flags**

## 2.6 FIFO Data Buffer and Interrupt Events

The interrupts listed here are independent of the selected protocol. The following sequences should be executed for initialization:

- Bitfields xxINP of registers TBCTR and RBCTR define which SRx will be activated if the corresponding event occurs.
- Register TRBSR contains indication flags.
- Write a 1 to the corresponding bit position in register TRBSCR to clear its status bit in TRBSR.
- The interrupt priority level and enable/disable are controlled by Nested Vectored Interrupt Control (NVIC) unit in XMC, please see RM for more details about CMSIS functions to access ARM Cortex-M4 NVIC.

**Table 7 FIFO Data buffer interrupts**

| Flag | Indication | Interrupt Enabled / SRx selected by | Cleared by |
|---|---|---|---|
| TRBSR.STBI | Standard TxFIFO event | TBCTR.STBIEN / TBCTR.STBINP | TRBSCR.CSTBI |
| TRBSR.STBT | Standard TxFIFO event trigger (activated via TBCTR.STBTEN=1) | | by HW |
| TRBSR.TBERI | TxFIFO error event | TBCTR.TBERIEN / TBCTR.ATBINP | TBCTR.TBERIEN |
| TRBSR.SRBI | Standard RxFIFO event | RBCTR.SRBIEN / RBCTR.SRBINP | TRBSCR.CSRBI |
| TRBSR.SRBT | Standard RxFIFO event trigger (activated via RBCTR.SRBTEN=1) | | by HW |
| TRBSR.ARBI | Alternative RxFIFO event | RBCTR.ARBIEN / RBCTR.ARBINP | TRBSCR.CARBI |
| TRBSR.RBERI | RxFIFO error event | RBCTR.RBERIEN / RBCTR.ARBINP | TRBSCR.CRBERI |

### Tx FIFO Buffer initialization bitfields

- TBCTR.SIZE = 1, 2, 3, 4, 5 selects FIFO size of 2, 4, 8, 16, 32
- TBCTRL.LIMIT = target FIFO filling level
- TBCTR.LOF = **0** (STBI interrupt occurs when FIFO level is lower than LIMIT, which means TxFIFO should be filled again)
- TBCTR.DPTR = pointer to the first FIFO number

*Note: Flag TRBSR.STBI is only set when the transmit buffer fill level falls below the programmed limit (TBCTR.LOF=0).*

### Rx FIFO Buffer initialization bitfields

- RBCTR.SIZE= 1, 2, 3, 4, 5 selects FIFO size of 2, 4, 8, 16, 32.
- RBCTRL.LIMIT = target FIFO filling level
- RBCTR.LOF = **1** (SRBI interrupt occurs when FIFO level gets bigger than LIMIT, means RxFIFO should be read out)
- RBCTR.DPTR = pointer to the first FIFO number
- RBCTR.RNM (optional)

*Note: Flag TRBSR.SRBI is only set when the receive buffer fill level exceeds the programmed limit (TBCTR.LOF=1).*

# 3 Synchronous Serial Channel (SSC) Mode

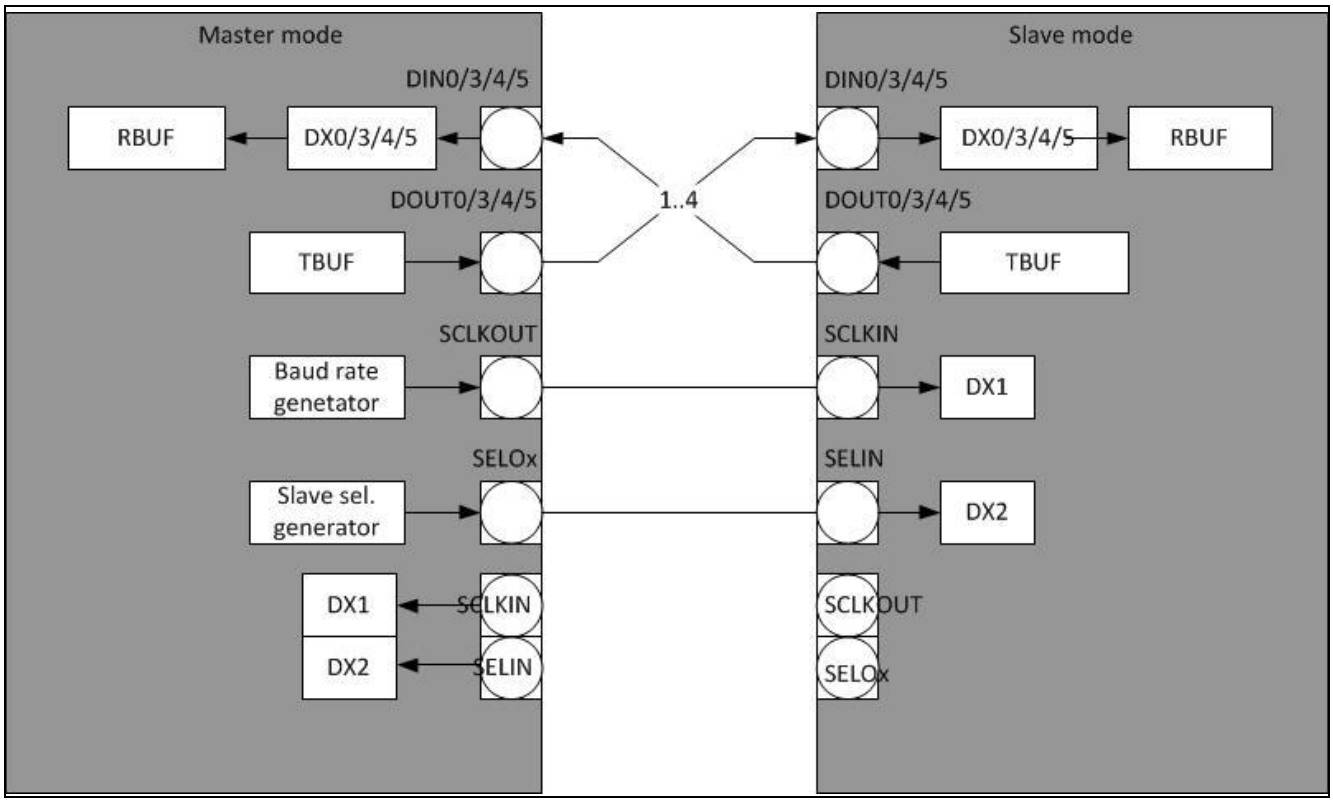


**Figure 13 SSC Signal Connection in Full_Duplex Mode**

This figure shows the standard SSC protocol, consisting of one input and one output data line. The XMC family of products also support two (dual-SSC) or four (quad-SSC) input/output data lines.

The SSC mode is selected when CCR.MODE = $0001_b$.

## 3.1 Input Stages, Output Signals and the Protocol Pre-Process

### Master mode

At DX1 the PPP uses the baud rate generator output SCLK directly as input for the data shift unit and gives the signal SCLKOUT on the shift clock output pin.

At DX2 the PPP provides the output MSLS with the SSC specific delay ($T_{ld}$, $T_{iw}$, $T_{td}$ and $T_{nf}$), and uses it as input for the data shift unit.

In SSC master mode, setting DX1 or DX2 is optionally used for delay compensation. The data input DX0/3/4/5 leads to the data shift unit and they are linked directly from the pins DINx. The PPP is not used and DX0CR.INSW must be set to ‘1’.

*Note: In a given application, the output pin SELO[7:0] is usually used as the Chip Select line (CS) for the SSC device, and it normally has an active ’low’ level. In this case the polarity of the SELO signal has been inverted by setting pin SELINV in the register PCR.*

### Slave mode

In SSC slave mode the PPP is not used in the input stages.

DX0/3/4/5, DX1, DX2 signals are linked directly from the pins DINx, SCLKIN and SELIN (set bit DXxCR.INSW to '1').

*Note: If a 'low' active Chip Select line (CS) is used as input signal for the slave SSC device, its polarity must be inverted (via DX2CR.DPOL).*

In USIC SSC operation mode a re-synchronization will automatically be performed by the CS signal.

In slave mode the DX2 signal is also used as a reset signal for its internal data shift unit. For example, after an error during SSC communication it is possible to reset the state machine by generating an active DX2, single input signal. Both master and slave use the USIC module, and the SPI master is switched off after 10 bits rather than 16 bits have been transmitted, re-sending the Word again.

- If the CS line is used (the 4-line SSC mode) then the slave device does not need special action to reset the SSC channel. With the new CS edge sent by the master, the content of the internal shift register of the slave will automatically be reset and a new data word can be completely received by the slave. Only if the FIFO is used do we need to flush the FIFO via register TRBSCR (bit FLUSHTB/ FLUSHRB).
- Some applications use the 3-line SSC mode (CS is not used). If the USIC module is used as SSC slave then we need to simulate a CS input signal (a falling edge for input DX2) to reset the internal data shift unit or to reset the USIC module completely via PRSETx/PRCLRx.

```
U1C0_DX2CR |= 0x0100;         // set bit DPOL
U1C0_DX2CR &= (~(0x0100));    // clear bit DPOL
```

- Since the data shift units internal SCLK signal has an active 'high' level, if no SELIN (non-CS) is used, the DX2 stage has to deliver a (permanent) 1-level to the data shift unit. This is achieved by programming bitfield DX2CR.DSEL = $111_b$.

*Note: In a multiplex CSx system, if a slave device is not selected (DX2 stage delivers a 0 to the data shift unit) a shift clock pulse is received. In this case the shift clock pulses are ignored, the incoming data is not received, and the DOUT0/3/4/5 outputs the passive level (SCTR.PDL) (see the errata issue "USIC_AI.020").*

## 3.2 Baud rate Generation

The baud rate of the SSC is defined by the frequency of the SCLK signal (one period of $f_{SCLK}$ represents one data bit) and is only required in the master mode.

SSC baud rate generation is based on $f_{PB}$ ($f_{sys}$) via BRG.CLKSEL=$00_b$.

In a standard SSC application, the phase relation between MCLK and SCLK is not relevant, so the 2:1 divider can be switched OFF (PPPEN=0).

Baud rate calculation (fractional divider mode):

$$fsclk = fsys \times \frac{STEP}{1024} \times \frac{1}{2} \times \frac{1}{PDIV + 1}$$

If the phase relation is requested (using MCLK as the clock reference for external devices for example), then the 2:1 divider must be switched ON (PPPEN=1).

## 3.3 Data Shifting and Handling

### 3.3.1 Data Transmission and Reception

Frame length (FLE)

- the number of bits per frame

Word length (WLE)

- for each data word control

#### USIC SSC master mode

In SPI master mode, the CSx (its internal signal is MSLS) is generated automatically by the PPP(PCR.MSLSEN must be set to '1'). This signal indicates the start and the end of a data transfer.

There are two ways to control the end of frame:

- Configuration of the data frame length FLE < 63.
  - The frame is considered as finished and the remaining data bits in the last data word are not transferred if the programmed number of bits per fame is reached within a data word.
- Configuration of the data frame length FLE = 63.
  - The frame is considered as finished and the remaining data bits in the last data word are not transferred if a de-activation of MSLS is detected within a data word.

In master mode, frame transmission/reception can be started when the data in the transmit buffer TBUF is valid (TCSR.TDV is set).

The internal signal MSLS is set together with the corresponding event flag (PSR.MSLS) and enters the first leading delay state.

After the delay ($T_{ld}$) generated by PPP has elapsed, the internal shift clock SCLK is issued, and the data is shifted out at the rising edge of SCLK.

For every processed data bit when the falling edge of the shift clock SCLK is reached, the level of the input signal is latched.

- If FLE < 63 then a CS signal is generated automatically by the first data bit and deactivated at the end of the last bit including delay $T_{ld}$.
- If FLE = 63 then the user should give the start/end information of a data frame to create the desired length of the CS signal. Bit TCSR.SOF/EOF is for this software-based control method.

#### USIC SSC slave mode

In the case that the SELIN input signal (CSx) is used in slave mode, the data frame start/end detection is based on the edge detection of input DX2 in both transmission and reception process. Data frame length (SCTR.FLE) should be set to its maximum value (FLE=63).

In the case that the SELIN input signal is not used in slave mode, the data frame length must be programmed to the known value (FLE<63).

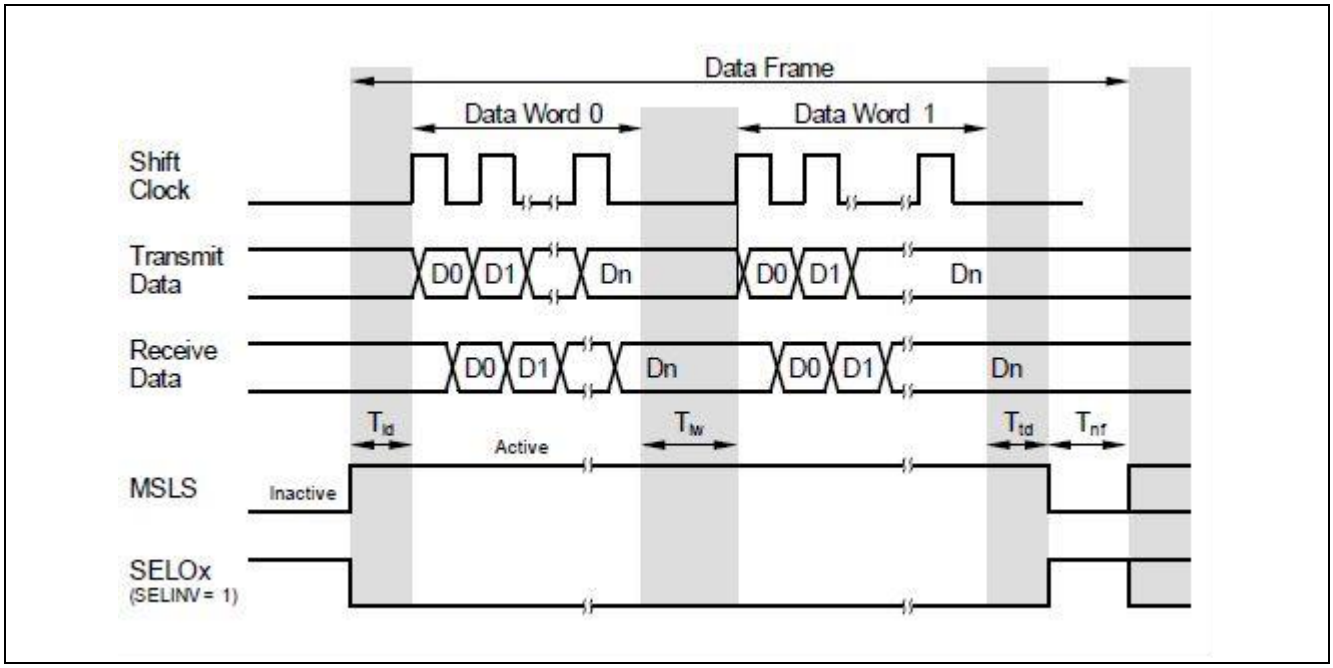


**Figure 14 Standard SSC Frame Format with SCLKCFG=00b**

### 3.3.1.1 SSC Frame Delay Control

SSC frame delay is generated automatically by the PPP in the SSC master mode based on $f_{CTQIN}$.

- $T_{ld}$ (leading delay)
  - Starts if data is valid for transmission. The first shift clock edge of SCLKis generated after the leading delay. The data shift unit always uses the rising edge for data shifting and the latch edge for data receiving.
- $T_{td}$ (trailing delay)
  - Starts at the end of the last SCLK cycle of a data frame. At the end point of the trailing delay the MSLS becomes inactive. It corresponds to the slave hold-time requirements.
- $T_{nf}$ (next-frame delay)
  - After the next-frame delay has elapsed, the frame is considered as finished.
- $T_{iw}$ (inter-word delay)
  - Can be optionally enabled/disabled by PCR.TIWEN. It is used if a data frame consists of more than one data word.

In a standard SSC application, such as the setup and hold time, the $T_{ld}$ and $T_{td}$ are mainly used to ensure stability on the input/output lines.

Normally $f_{CTQIN} = f_{SCLK}$ via CTQSEL=$10_b$.

Delay time calculation:

$$Tld = Ttd = \frac{(PCTQ + 1) \times (DCTQ + 1)}{fsclk}$$

$$Tiw = Tnf = \frac{(PCTQ1 + 1) \times (DTCQ1 + 1)}{fsclk}$$

*Note: Delay $T_{iw}$ can be disabled via bit PCR.TIWEN.*

### 3.3.1.2 Shift Clock (SCLK) and CS

In master mode, the shift clock is generated by the internal baud rate generator.

In slave mode, the signal SCLKIN is received from an external master.

#### CS generation

If the SSC module is in master mode, the slave select signal (the internal signal MSLS) is generated automatically by PPP.

SSC interfaces have 4 different configurations regarding the shift and latch edge for data transmission and reception process.

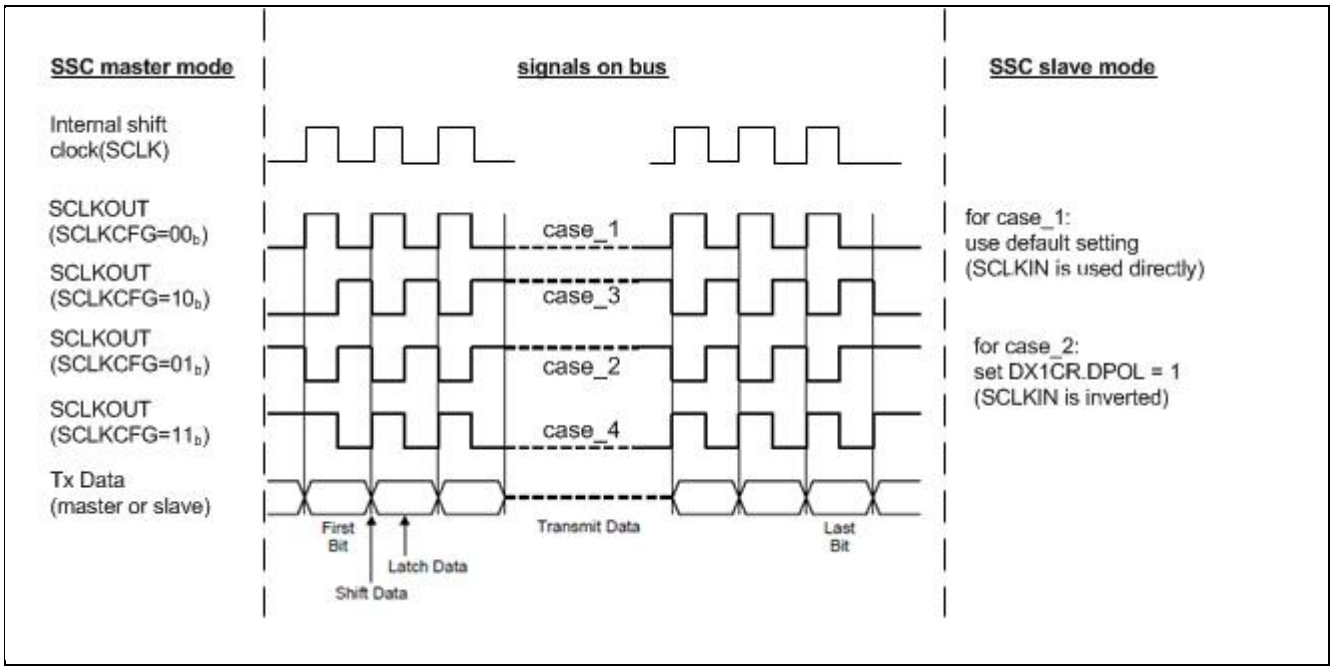


**Figure 15 Shift Clock in SSC Communication**

#### USIC SSC master mode support

- case_1 (SCLKCFG = $00_b$): No delay, no polarity inversion (SCLKOUT equals SCLK)
- case_2 (SCLKCFG = $01_b$): No delay, polarity inversion
- case_3 (SCLKCFG = $10_b$): SCLKOUT is delayed by 1/2 shift clock period, no polarity inversion
- case_4 (SCLKCFG = $11_b$): is delayed by 1/2 shift clock period, polarity inversion

#### USIC SSC slave mode support (XMC4500)

- case_1: no delay, no polarity inversion (SCLKOUT equals SCLK)
- case_2: no delay, polarity inversion (SCLKOUT equals inverted SCLK): set DX1CR.DPOL to 1 to adapt
- case_3/case_4: software workaround can be used.

*Note: In slave mode bitfield SCLKCFG is ignored.*

In slave mode the shift clock signal is handled by the input stage DX1 (signal SCLKIN is received from an external master), so the DX1 stage has to be connected to an input pin. For case_1 the input signal on DX1 pin can be directly forwarded to the internal data shift unit. For case_2 the DX1 stage must invert (set DX1CR.DPOL to 1) the received signal to adapt the SCLKIN polarity, because the internal data shift unit always takes data transmission on the rising edge, and data reception on the falling edge.

*Note: In the XMC4400 and XMC1000 product families, the bit PCR.SLPHSEL is implemented to handle the shift clock of the data shift unit to support case_3 and case_4 in SSC slave mode.*

### 3.3.2 Parity Mode

The XMC products support parity generation for transmission and parity check for reception on frame base in master and slave mode. For consistency reasons, all communication partners have to be programmed to the same parity mode.

- CCR.PM: define the type of parity. In SSC parity mode the clock extends by one cycle after the last data word of the data frameindependent of SDIR setting (MSB or LSB).
- RBUFSR.PAR: the monitored parity bit value.
- PSR.PARERR: the result of the parity check
- PCR.PARIEN: Parity Error Interrupt Enable

*Note: For dual and quad SSC protocols, the parity bit will be transmitted and received only on DOUT0 and DX0 respectively, in the extended clock cycle.*

*Note: Parity bit generation or detection is not supported for a frame length > 64 data bits; i.e. setting FLE=0x3F.*

## 3.4 SSC Software Configuration

### 3.4.1 SSC Full-Duplex Communication (Example_1)

A full-duplex system allows communication in both directions at the same time. A synchronous data transfer is characterized by a simultaneous transfer of a shift clock signal together with transmit and receive data signals.

*Note: In XM1000 family, only one USIC module is available. The FIFO and interrupt SRx are shared.*

- Input stages

**Table 8 Input stages (Example_1 for XMC4000)**

| | INSW | DPOL | DSEL (DXxA…G): G=Loopback | Used as… |
|---|---|---|---|---|
| **Master mode** | | | | |
| DX0,3,4,5 | 1 (PPP not used) | 0 (not inverted) | P0.4=data input (DX0CR.DX0A) | Data input |
| DX1 | Not used | | | |
| DX2 | Not used | | | |
| **Slave mode** | | | | |
| DX0,3,4,5 | 1 (PPP not used) | 0 (not inverted) | P2.2=data input (DX0CR.DX0A) | Data input |
| DX1 | 1 (PPP not used) | 0 | P2.4=clock input (DX1CR.DX1A) | SCLKIN |
| DX2 | 1 (PPP not used) | 1 (see Note) | P2.3=CS input (DX2CR.DX2A) | CS input |

*Note: In an application, the output pin SELO[7:0] is usually used as the Chip Select line (CS) for the SSC device and it normally has an active 'low' level. In this case the polarity of the SELO signal has been inverted by set pin SELINV in the register PCR.*

- Output signals

**Table 9 Output signals (Example_1 for XMC4000)**

| | | |
|---|---|---|
| **Master mode** | | |
| DOUTx (x=0,1,2,3) | P0.5=DOUT0<br>DOUT0→single data (DOUT1…3: not used) | Data output |
| SCLKOUT | P0.11=clock output,<br>Generated by the baud rate generator based on *f*sys.<br>4 possible settings (via BRG.SCLKCFG) | Clock output |
| SELOx (x=0…7) | P0.6=CS output (SELO0)<br>Set bit PCRL.MSLSEN to enable MSLS<br>Set bit PCR.SELCTR to use CS direct select mode<br>Set bit field PCR.SELO to active the corresponding SELOx output line<br>Set bit PCR.SELIN to invert MSLS for an active 'low' CS output signal | CS output |
| **Slave mode** | | |
| DOUTx (x=0,1,2,3) | P2.5=DOUT0<br>DOUT0→single data (DOUT1…3: not used) | Data output |
| SCLKOUT | Not used | |
| SELO[7:0] | Not used | |

*Note:*

1. *MSLS is an internal signal, which is active 'high'. To generate the MSLS the bit PCR.MSLSEN must be set.*
2. *Direct Select Mode: a SELOx output becomes active while the internal signal MSLS is active and bit x in bit field SELO is 1. Several external slave devices can be addressed in parallel if more than one bit in bit field SELO is set.*
3. *The output pin SELO[7:0] is usually used as the CS for SSC device and it normally has an active 'low' level. In this case the polarity of the SELO signal has been inverted by set pin SELINV in the register PCR.*

- Data shift control (SCTR): TRM=01b, PDL=1, SDIR=1(MSB first), FLE=WLE=15
- Data transmission control (TCSR): no trigger, no gating, single shot mode
- Parity (CCR.PM): not used
- Protocol-related information (PCR): SELO=1(P0.6=U1C0_CS0), FEM=1, SELINV=1, SELCTR=1, MSLSEN=1
- Interrupts point (INPR) and enable control (CCR): INPR.AINP/RINP=SR2; CCR.AIEN/RIEN=1
- Interrupt: enable AIR/RI interrupt via CMSIS functions: NVIC_SetPriority(..); NVIC_EnableIRQ(..)
- Input/output pins configuration:
  - Output: PORTx->IOCRx.PC(alternate output function)

**Table 10 Input / Output pins (Example_1 for XMC4000)**

| Input / output pins | Function | pin | Port driver (IOCRxPC) |
|---|---|---|---|
| **Master mode** | | | |
| data out | U1C0_DOUT | P0.5 | ALT2 (push pull) |
| clock output: | U1C0_SCLKOUT | P0.11 | ALT2 (push pull) |
| CS output | U1C0_SEL0 | P0.6 | ALT2 (push pull) |
| data input | U1C0_DX0A | P0.4 | input |
| **Slave mode** | | | |
| data input | U0C1_DX0A | P2.2 | input |
| clock input | U0C1_DX1A | P2.4 | input |
| CS input | U0C1_DX2A | P2.3 | input |
| data output | U0C1_DOUT | P2.5 | ALT2 (push pull) |

### 3.4.2 Software in Loop-back Mode (Example_2)

*Note: This is only applicable in master mode.*

We use the same initialization routine as that in Example_1, until DX0CR.DSEL which must be set to “G” ($110_b$).

*Note: In loop-back mode the data output pins are not required to be switched on, but if they are then the signal can be monitored on an oscilloscope.*

### 3.4.3 SSC for Half-Duplex Communication

In half-duplex mode only one data line is shared between the communication partners and it is used for both transmission and reception of data (MRST and MTSR are connected together).

The user software must ensure that only one transmitter is active at a time.

There are two ways to avoid collisions on the data exchange line:

- Only the transmitting channel may enable its transmit pin driver (enable/disable push/pull drivers)
- Devices use open-drain outputs to allow the wired-AND connection in a multi-transmitter communication

Since the SSC data transfer is synchronized by a simultaneous transfer of a shift clock signal together with the transmission and reception of the data signal, the dummy data of the register TBUF in an inactive partner should be set to all 1’s.

### 3.4.3.1 Standard SSC Half-Duplex System (Example_3)

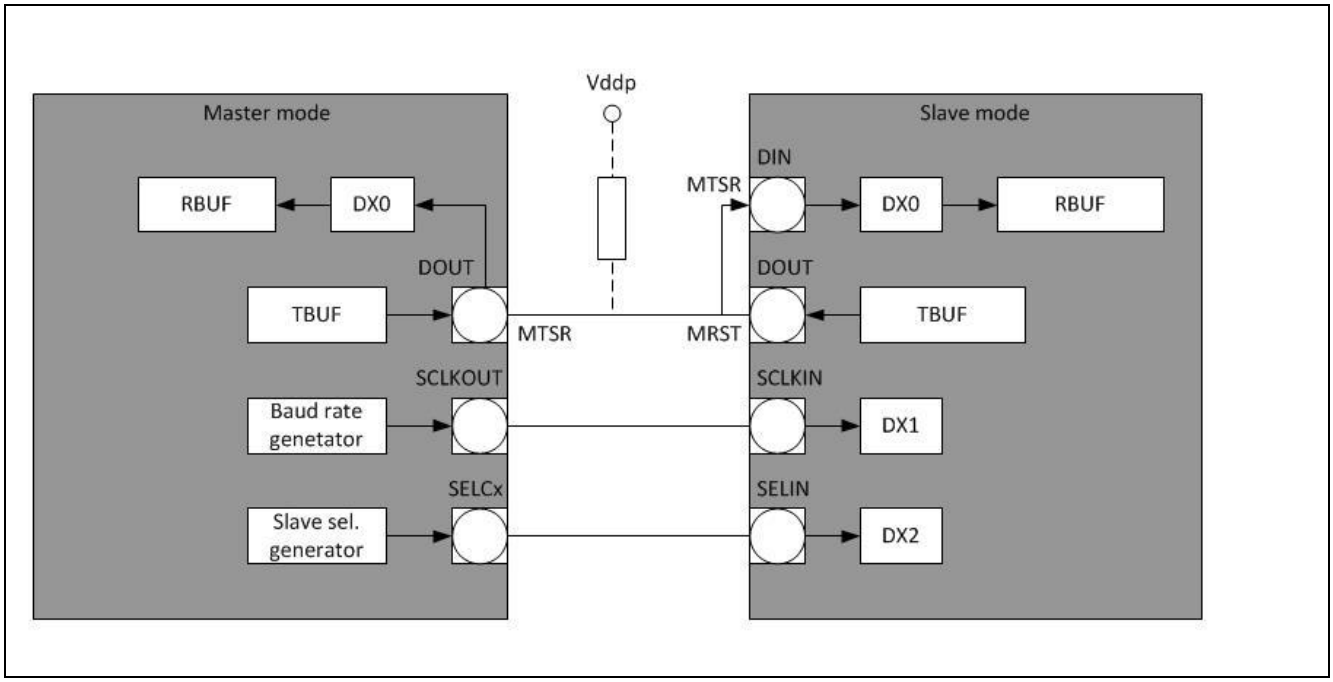


**Figure 16 Signal Connection for SSC Standard Half-Duplex System**

In this example figure the master mode uses an internal connection with the output pin DOUT. The slave mode uses an external connection between DOUT and DIN pins. Internal connection means DX0x points to DOUT. For example, XMC1100, P1.0 => U0C0_DX0C => U0C0_DOUT.

*Note:*

1. *Not all the data output DOUTx pins contain an internal connection. For example, P1.5 (as U0C0_DOUT0) has an internal connection to DX0 (U0C0_DX0A), but P1.7 (U0C0_DOUT) has no such connection in XMC4000 family*

#### Initialization routine

This example uses the same initialization routine as for Example_1, but with the changes indicated in the following table.

**Table 11 Standard SSC half-duplex system initialization (Example_3 for XMC4000)**

| | Function | pin | Port driver (IOCRxPC) |
|---|---|---|---|
| **Master mode** | Internal connection mode is used | | |
| Data out | U1C0_DOUT | P0.5 | ALT2 (open drain) |
| Clock output | U1C0_SCLKOUT | P0.11 | ALT2 (push pull) |
| CS output | U1C0_SEL0 | P0.6 | ALT2 (push pull) |
| Data input | U1C0_DX0A → U1C0_DX0B (see note) | P0.4 → P0.5 (see note) | Input |
| **Slave mode** | External connection mode is used | | |
| Data input | U0C1_DX0A | P2.2 | Input |
| Clock input | U0C1_DX1A | P2.4 | Input |
| CS input | U0C1_DX2A | P2.3 | Input |
| Data output | U0C1_DOUT | P2.5 | ALT2 (open drain) |

*Note: Instead of P0.4, we use P0.5 as data input U1C0_DX0 (the internal connection mode is used).*

### 3.4.3.2 Hardware-controlled SSC Half-Duplex System

Hardware-port pin control is implemented for the XMC family of products. One, two or four port pins can be selected with the hardware port control to support SSC protocols with multiple bi-directional data lines, such as dual and quad-SSC. This selection, and the enable/disable of the hardware port control, is made through CCR.HPCEN.

USIC is usually used as master mode. For data transmission direction hardware pins must be switched as input or output. The direction of all selected pins is controlled through a single bit, SCTR.HPCDIR.

SCTR.HPCDIR is automatically shadowed with the start of each data word to prevent the pin changing direction in the middle of a data word transfer (see the errata issue “USIC_AI.013”).

In the XMC family several peripherals have hardware controlled pins. Because multiple peripheral I/Os are mapped on some pins, the register Pn_HWSEL is used to select which peripheral has control over the pin.

In XMC4000 products, all USIC hardware-controlled pins use the HW0 control path (Pn_HWSEL.HWx=01b).

In XMC1000 products, all USIC hardware-controlled pins use the HW1 control path (Pn_HWSEL.HWx=10b),

*Note: In the XMC4500 144 pin package, each channel (2 channels per module) has hardware-control pins, but in XMC1100 only U0C0 can use this feature and U0C1 does not have any hardware-control pins.*

**Example_5** gives a detailed initialization routine for using this feature.

## 3.4.4 Multi-IO SSC Support

This XMC family feature supports multiple data input/output SSC protocols, such as dual and quad-SSC. The relevant register bitfields are:

- SCTR.DSM
- SCTR.HPCDIR
- CR.HPCEN

*Note: See section **2.4.3** for additional information.*

For a quick data exchange between master and slave devices (usually a memory device), the hardware port pins should be used as this allows the USIC pins to directly drive complex control and communication patterns without further software interaction with the ports for a half-duplex system.

On the XMC4500 Easy Kit V3, the QSPI flash device (MICRON N25Q032A) is mounted. In the DAVE™ example project “SPI001_Example1”, the standard full-duplex mode is used to program this external flash device. In “SPI001_Example2”, in the phase flash command sequence, the full-duplex mode with alternate port pin function is used as standard, while in phase data write/read sequence the Multi_IO feature with HW_controlled port pin function is used.

*Note: Download the DAVE™ code from **DAVE™3 at Infineon,** or find it in Example_4 available for download alongside with this document.*

**Example_5** shows how to realize the data sequence as well as the command sequence by using HW_controlled pins.

**Table 12 Exampel_5 for XMC4000**

**XMC4500 CPU_45A_V3 kit**
**QSPI flash device=MICRON N25Q032A (Manufacturer=0x20)**

| | U1C1 in master | Pin (HW_control) | Register | N25Q032A |
|---|---|---|---|---|
| Data out/in | DOUT0/DX0 | P3.15 | PORT3_HWSEL(HW15=01b) | SI/IO0 |
| | DOUT1/DX1 | P3.14 | PORT3_HWSEL(HW14=01b) | SO/IO1 |
| | DOUT2/DX2 | P0.15 | PORT0_HWSEL(HW15=01b) | W#/ACC/IO2 |
| | DOUT3/DX3 | P0.14 | PORT0_HWSEL(HW14=01b) | HOLD#/IO3 |
| Clock output | SCLKOUT | P0.13(ALT2) | PORT0_IOCR12(PC13=10010b) | SCK |
| CS output | SELO**1** | P3.3(ALT2) | PCR.SELO=0000000**1**0b | CS# |

**XMC4500 CPU_45A_V3 kit**
**QSPI flash device=MICRON N25Q032A (Manufacturer=0x20)**

| | | | | |
|---|---|---|---|---|
| | | | PORT3_IOCR0(PC3=10010b) | |

*Note: QSPI flash device is not available on XMC4400 CPU_44A_V2 kit.*

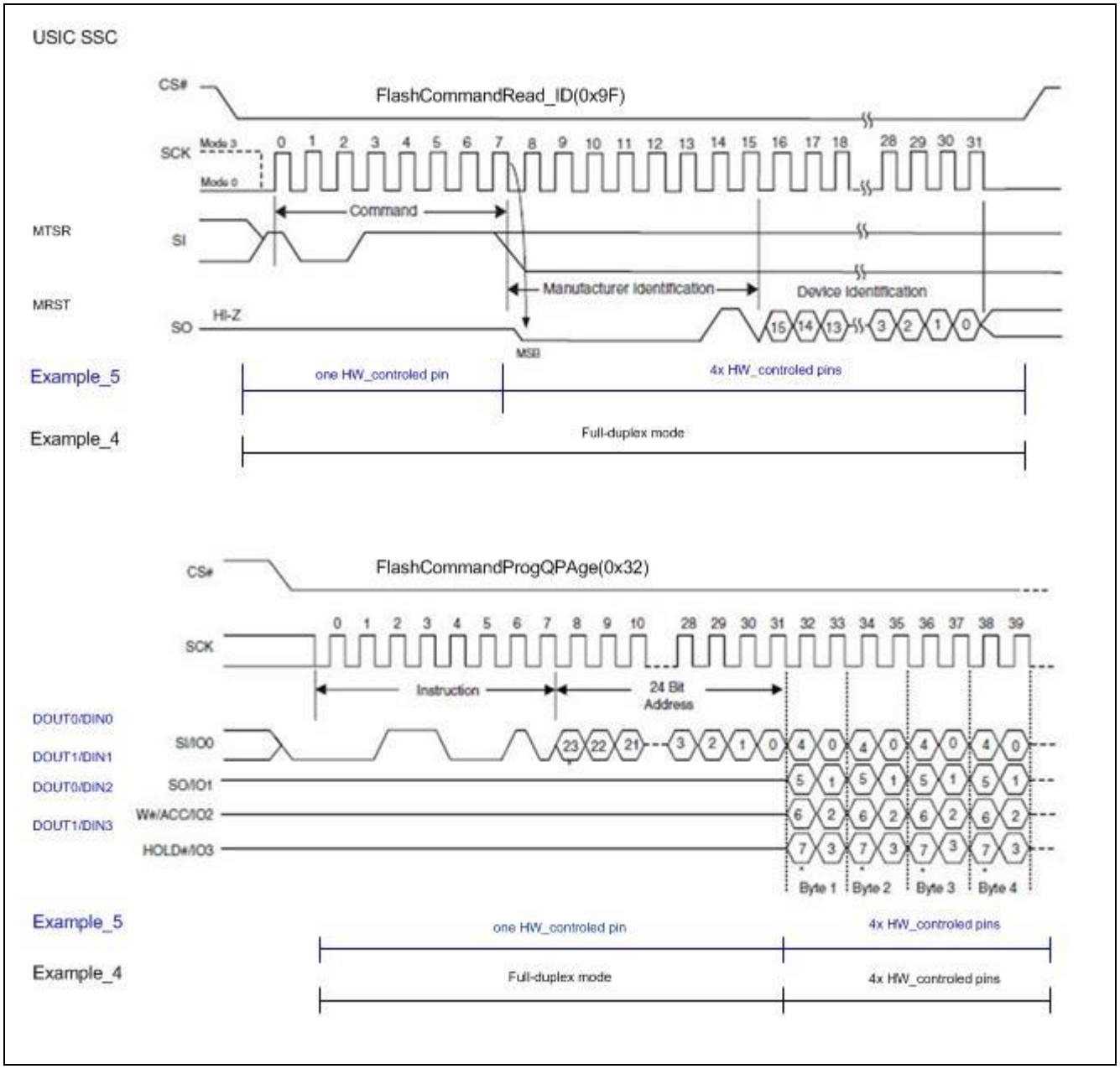


**Figure 17 QSPI Flash Command “Read Identification” and Sequence “QUAD Page Program”**

On execution of read commands some information must be read out. In this example figure for instance, for “read identification” the manufacturer and device will be driven on the bus to USIC DX1 pin.

The USIC module can only select one shift mode at a time for one, two or four data bits. Therefore at the command “read device code” phase on the ‘SI’ line, a minimum of 2x HW_control pins (SCTR.DSM = 10b) must be switched on to get read data. Received data must be calculated via software in order to get the right code.

*Note: In Example_5 the 4 data lines shift mode (SCTR.DSM = 11b) is used.*

## 3.5 Delay Compensation

For the SSC protocol, USIC works with fsys/2 (40Mbaud/fsys=80MHz). This maximum baud rate is based on module capability. In the application environment it is limited by several factors, including driver delays, signal propagation times, synchronization and filter delay, and so on. In the data receive process the minimum required setup time must also be considered.

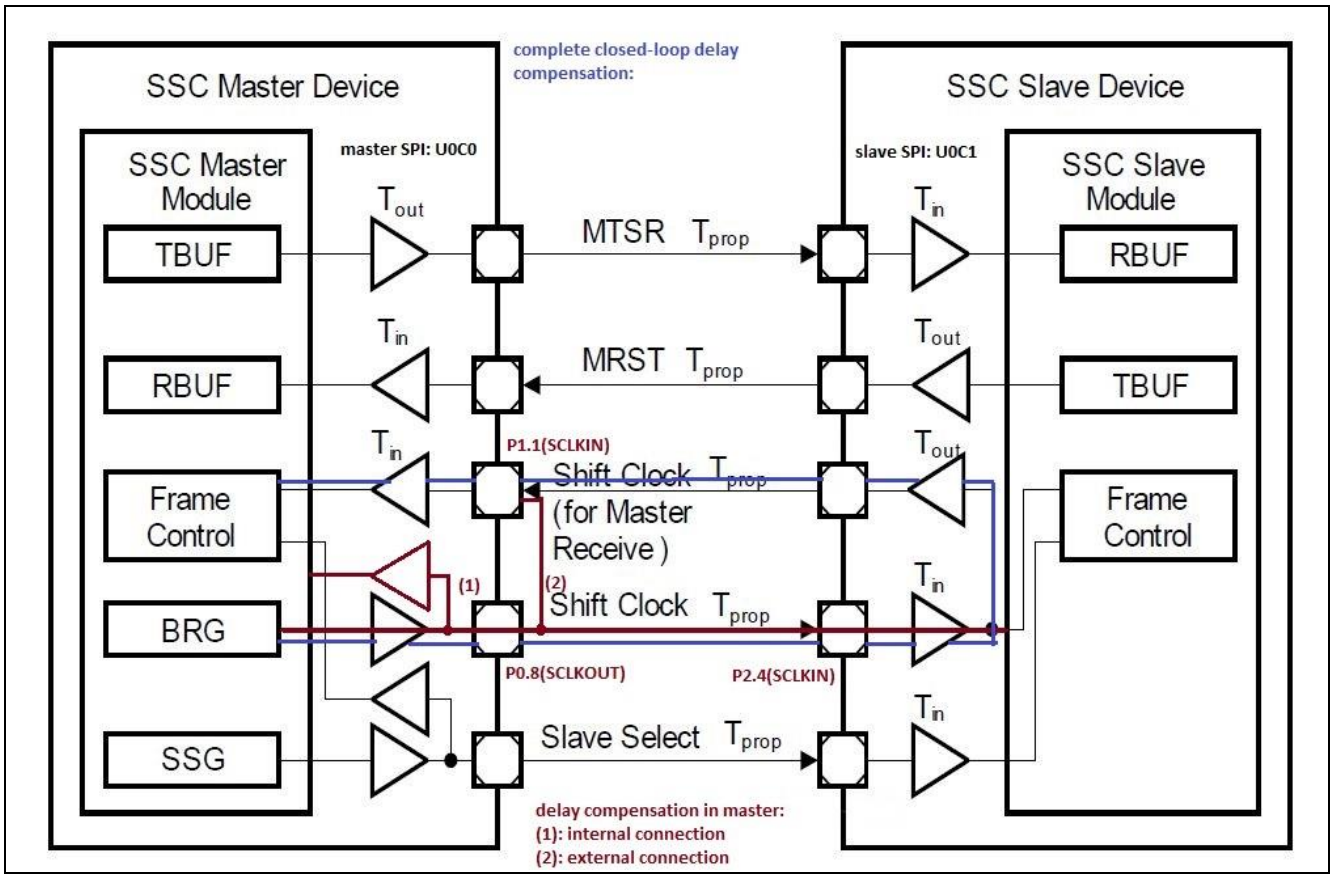


**Figure 18 Compensation in master mode and complete closed-loop compensation**

The closed-loop delay is a system-inherent factor. The delay time between the generation of the shift clock signal and the evaluation of the receive data by the master SSC module is given by the sum = $T_{out\_master}$ + 2 x $T_{prop}$ + $T_{in\_slave}$ + $T_{out\_slave}$ + $T_{in\_master}$ + module reaction times, where:

- $T_{out\_master}$/$T_{out\_slave}$
  - Delay time through the output driver stage to the pin (default setting A1+/A2 pin: falling/rising time<16ns. Please refer to the appropriate data sheet)
- $T_{prop}$
  - Delay time on the wires.
- $T_{in\_slave}$/$T_{in\_master}$:
  - Delay time through the input pin to the module input stage.
- Module reaction times
  - Delay time due to digital filter, synchronization, setup/hold time (Please refer to the appropriate data sheet)

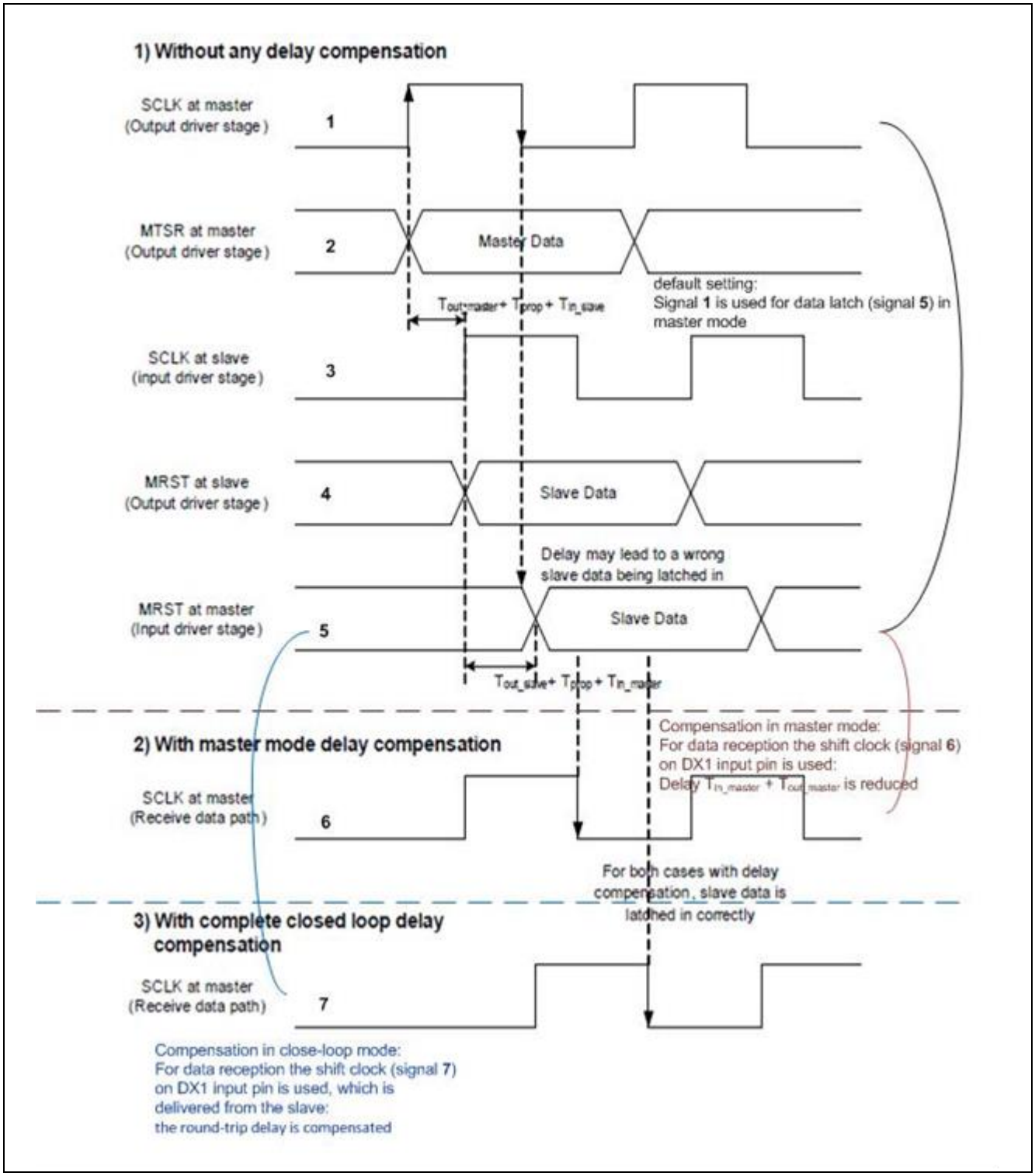


**Figure 19 SSC Signals Delay Timing Waveform**

Using the default standard setting, as data is received in the master SPI process, the clock signal (signal 1 in this example figure) generated from the baud rate generator in the master device, is used to latch a data signal (signal 5). For a higher baud rate, this may lead to incorrect data being latched. A higher baud rate can be reached by using a delay compensation feature.

In XMC there are two compensation methods: delay compensation and complete closed-loop delay compensation.

### Delay compensation in master mode

This method uses the input clock signal at the DX1 pin for data latching, instead of the SCLKOUT generated by the baud rate generator (signal 6 and 5 in Figure 19).

With this method the clock output driver delay in master mode is compensated; i.e. the delay between the evaluated clock signal and Rx data by master is reduced by $T_{in\_master} + T_{out\_master}$.

**Example_6** demonstrates the initialization routine using delay compensation in master mode.

An external or an internal connection can be used:

- External connection
  - P0.8 → U0C0_SCLKOUT, P1.1 → U0C0_DX1A

```
USIC0_CH0->DX1CR |= (0<<0);    // DX1CR.DSEL=A
USIC0_CH0->DX1CR |= (1<<4);    // DX1CR.INSW=1
```

  - P1.1 (U0C0_DX1A) has to be connected with P0.8 (U0C0_SCLKOUT) externally.
- Internal connection
  - P0.8 → U1C0_SCLKOUT, P0.8 → U1C0_DX1B

```
USIC0_CH0->DX1CR |= (1<<0);    // DX1CR.DSEL=B
USIC0_CH0->DX1CR |= (1<<4);    // DX1CR.INSW=1
```

*Note:*

1. *The internal connection can only be used for a bi-directional clock pin and it does not lead to additional pins for the SSC communication.*
2. *This method is compatible with the USIC module in the XC2000 / XE166 product families. In this setting the clock input (DX1) signal is used for both data reception and transmission, but only the clock used for receiving data has to be compensated, not the clock used for data transmission. Therefore with this method the maximum baud rate is limited.*
3. *Bit DCEN is implemented in XMC to allow the Rx shift clock to be controlled independently from the Rx shift clock. When DCEN=1, the Tx shift clock is taken from the baud rate generator directly (see the errata issue "USIC_AI.008").*

### Complete closed-loop delay compensation

*Note: This is implemented in the XMC4400 and XMC1000 product families, but not in the XMC4500 family.*

The principle behind this method is to feedback the clock signal to the master mode, so that the master can use this clock signal to latch data from the slave. Because the clock signal is through the complete closed-loop signal path, the delay between the clock used in the master and data (from the slave) signal is therefore fully compensated.

This method can only be realized when both master and slave use the USIC module. In slave mode the CLKOUT pin should be enabled via set BRG.SCLKOSEL to 1, while the USIC master mode must be 'delay compensation in master'(see the errata issue "USIC_AI.008").

**Table 13 Example_6 for XMC4000**

| | | | | |
|---|---|---|---|---|
| **U0C0 in master** | | | | |
| Data output | DOUT0 | P1.5 | ALT2 (A1+) | |
| Clock output | SCLKOUT | P0.8 | ALT2 (A2) | |
| CS output | SELO0 | P0.7 | ALT2 (A2) | |
| Data input | DIN (DX0) | P1.4 | DX0B, input, (A1+) | |
| Clock input | SCLKIN (DX1) | P1.1 | DX1A, input, (A1+) | external connection for delay compensation in master mode |
| **U0C1 in slave** | | | | |
| Data input | DIN (DX0) | P2.2 | DX0A, input, (A2) | |
| Clock input | SCLKIN (DX1) | P2.4 | DX1A, input, (A2) | |

| | | | | |
|---|---|---|---|---|
| **U0C0 in master** | | | | |
| CS input | CS input (DX2) | P2.3 | DX2A, input, (A2) | |
| Data output | DOUT0 | P2.5 | ALT2, (A2) | |

## 3.6 Multiple MSLS Output Signals

The SSC module supports up to 8 different SELOx output signals for master mode operation in one USIC module. USIC provides two configuration modes to select the MSLS signal:

- Direct control mode
  - Write PCR.SELO[7:0] as individual values for each SELOx line
- Automatic update mode
  - Enabled by TCSR.SELMD=1.
  - PCR.SELO[4:0] is updated with TCI[4:0] and PCR.SELO[7:5] is always ’0’

Each USIC module has the transmit buffer input locations TBUFx (x=00-31), addressed by using 32 consecutive addresses.

If TCSR.SELMD = 1, data written to one of these locations will appear in a common TBUF register, and the 5-bit TCI[4:0] coding will be updated accordingly.

The relationship between TBUFx, TCI[x] and MSELx is listed in following table.

**Table 14 Select output control: TCSR.SELMD = 1**

| Write to TBUFx | TCI[4:0] | PCRH.SELO[7:0] | SELOx signals |
|---|---|---|---|
| TBUF01 | 00001b | 0000,0001b | SELO0 active |
| TBUF02 | 00010b | 0000,0010b | SELO1 active |
| TBUF04 | 00100b | 0000,0100b | SELO2 active |
| TBUF08 | 01000b | 0000,1000b | SELO3 active |
| TBUF16 | 10000b | 0001,0000b | SELO4 active |
| TBUF03 | 00011b | 0000,0011b | SELO0/1 active |
| TBUF07 | 00111b | 0000,0111b | SELO0/1/2 active |
| TBUF00 | 00000b | 0000,0000b | no SELOx |
| …… | …….. | …….. | ……. |

w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG