

XHFC - 2SU

**Extended ISDN HDLC FIFO controller
with
two Universal ISDN Ports**





Cologne Chip AG
Eintrachtstrasse 113
D - 50668 Köln

Germany
Tel.: +49 (0) 221 / 91 24-0
Fax: +49 (0) 221 / 91 24-100

<http://www.CologneChip.com>
<http://www.CologneChip.de>
support@CologneChip.com

Copyright 1994 - 2012 Cologne Chip AG
All Rights Reserved

The information presented can not be considered as assured characteristics. Data can change without notice.

Parts of the information presented may be protected by patent or other rights.

Cologne Chip products are not designed, intended, or authorized for use in any application intended to support or sustain life, or for any other application in which the failure of the Cologne Chip product could create a situation where personal injury or death may occur.

Date	Remarks
June 2012	<p>Bitmap width of A_SL_CFG.V_CH_SNUM reduced to 4 bit.</p> <p>Description of bitmaps V_B1_TX_EN, V_B2_TX_EN (both in register A_SU_CTRL0), R_STATUS.V_MISC_IRQSTA and R_IRQ_OVIEW.V_MISC_IRQ slightly changed.</p> <p>Paragraph concerning HFC-channel processing limitation added to section 3.2.3 on page 78.</p> <p>Figure 6.5 (PCM clock synchronization) on page 229 modified concerning C2IO signal path to PCM data controller.</p> <p>Section 9.2.4 (Supply noise requirements for PLL locking) on page 294 added.</p>
May 2010	<p>No changes with regard to contents, only a few typographical corrections have been made.</p>
April 2010	<p>Sections 6.5.6, 6.5.7, 9.1.3.5, 10.3 and 13 added. Figures 9.3 and 10.2 added. Tables 2.3, 10.3, 13.1 and 13.2 added. Register R_SU_LED_CTRL added. Bit V_WD_EN in register R_BERT_WD_MD added.</p> <p>Reset groups and I/O characteristic of the pin list completely reworked. Section “Reset” also reworked.</p> <p>Register descriptions improved for e.g. A_B1_RX, A_B2_RX, A_D_RX, A_E_RX, A_FIFO_CTRL, A_SL_CFG, R_CHIP_ID, R_CIRM, R_CLK_CFG, R_FIFO_BL0_IRQ..R_FIFO_BL3_IRQ, R_IRQ_CTRL, R_IRQ_OVIEW, R_PCM_MD1 and R_RAM_CTRL.</p> <p>Register descriptions clarified and corrected for A_CON_HDLC, A_F1, A_F2, A_SU_CTRL0, R_IRQ_OVIEW, R_PCM_MD1, R_STATUS.</p> <p>Schematics slightly changed for processor interface (recommended capacitor added), component values of the quartz circuitry improved (see also calculation procedure in section 9.1.3).</p> <p>/WAIT signal added to parallel processor interface timing diagrams.</p> <p>Signals /SPICLK, SPICLK_IDLE and /SPICLK_IDLE added to timing diagram 2.20.</p> <p>Internal signal FSC renamed to FSC_RX (and as a consequence of this FSC_0 and FSC_1 renamed to FSC_TX_0 and FSC_TX_1). Internal signal FSC_TX added to figures 5.1, 5.5, 5.13, 5.15 and 5.16 and to the corresponding sections.</p> <p>Note *1 changed in NT state matrices for S/T (Table 5.4) and U_p (Table 5.9) line interfaces.</p> <p>Remarks added for bit scrambler configuration in sections 5.3.4 and 5.3.7.</p> <p>Remark added for monitoring applications in section 5.3.6.</p> <p>Remark concerning internal PCM loop added to section 6.2.</p> <p>Register bit V_FIFO_IRQMSK added to Figure 9.7.</p> <p>Remark concerning unused line interfaces added to section 9.4.4.1.</p> <p>Chapter 11 reworked and supplemented.</p>
October 2007	<p>Minor changes were made in this data sheet revision: Information added to Sections 5.2.6 and 9.4.4.1 concerning state machine behaviour and line interface frequency slip interrupt.</p>
July 2007	<p>C1 and C2 in external S/T receive circuitry moved from L_A /L_B pins to R_A /R_B pins, <i>t_{AHD}</i> added for bus interface write access in mode 2 (Motorola) and mode 3 (Intel), name V_CH_IRQ changed to V_STUP_IRQ in register R_IRQ_OVIEW, name V_FR_IRQSTA changed to V_FIFO_IRQSTA in register R_STATUS, description of many interrupt registers improved.</p>
March 2007	<p>I/O type and input/output characteristics revised in pin list, pins 6 and 7 changed from NC to GND in SPI operating mode (refer to pin list and SPI connection circuitry), SPI timing diagrams added, section “PCM clock synchronization” restructured and supplemented, application hints for expanding an existing system with ISDN ports added.</p>

Contents

About this data sheet and Cologne Chip technical support	27
1 General description	31
1.1 System overview	32
1.2 Features	33
1.3 Pin description	34
1.3.1 Pinout diagram	34
1.3.2 Pin list	36
2 Microprocessor bus interface	41
2.1 Mode selection	42
2.2 Parallel processor interface	43
2.2.1 Overview	43
2.2.2 Interface signals	43
2.2.3 Register access	44
2.2.3.1 Non-multiplexed / multiplexed access	44
2.2.3.2 Read* access	45
2.2.3.3 Register address read-back capability	45
2.2.3.4 Problems with interrupts between address and data accesses	45
2.2.4 Signal and timing characteristics	45
2.2.4.1 Bus interface in mode 2 and mode 3 (non-multiplexed)	45
2.2.4.2 Bus interface in mode 2m and mode 3m (multiplexed)	51
2.2.5 Microprocessor connection circuitries	55
2.3 Serial processor interface (SPI)	57
2.3.1 SPI control byte	57
2.3.2 SPI transactions	58
2.3.3 Transaction duration	59
2.3.4 Register write access	61
2.3.5 Register read access	63
2.3.6 Register access duration	64

2.3.7	Register address read-back capability	64
2.3.8	Problems with interrupts during transaction sequences	65
2.3.9	SPI timing diagrams	65
2.3.10	SPI connection circuitry	68
2.4	Auto-EEPROM mode	69
2.5	Register description	70
2.5.1	Write only registers	70
2.5.2	Read only registers	72
2.5.3	Read / write register	74
3	XHFC-2SU data flow	75
3.1	Data flow concept	76
3.1.1	Overview	76
3.1.2	Term definitions	76
3.2	Flow controller	77
3.2.1	Overview	77
3.2.2	Switching buffers	78
3.2.3	Timed sequence	78
3.2.4	Transmit operation (FIFO in transmit data direction)	79
3.2.5	Receive operation (FIFO in receive data direction)	79
3.2.6	Connection summary	80
3.3	Assigners	82
3.3.1	HFC-channel assigner	82
3.3.2	PCM slot assigner	82
3.3.3	Line interface assigner	82
3.3.4	Assigner summary	83
3.4	Data flow modes	85
3.4.1	Simple Mode (SM)	85
3.4.1.1	Mode description	85
3.4.1.2	Subchannel processing	87
3.4.1.3	Example for SM	87
3.4.2	Channel Select Mode (CSM)	93
3.4.2.1	Mode description	93
3.4.2.2	HFC-channel assigner	93
3.4.2.3	Subchannel Processing	93
3.4.2.4	Example for CSM	94
3.4.3	FIFO Sequence Mode (FSM)	99
3.4.3.1	Mode description	99

3.4.3.2	FIFO sequence	99
3.4.3.3	FSM programming	101
3.4.3.4	Example for FSM	102
3.5	Subchannel Processing	108
3.5.1	Overview	108
3.5.2	Subchannel registers	109
3.5.3	Details of the FIFO oriented part of the subchannel processor (part A)	109
3.5.3.1	FIFO transmit operation in transparent mode	109
3.5.3.2	FIFO transmit operation in HDLC mode	111
3.5.3.3	FIFO receive operation in transparent mode	111
3.5.3.4	FIFO receive operation in HDLC mode	111
3.5.4	Details of the HFC-channel oriented part of the subchannel processor (part B)	111
3.5.4.1	FIFO transmit operation in SM	111
3.5.4.2	FIFO transmit operation in CSM and FSM	112
3.5.4.3	FIFO receive operation in SM	112
3.5.4.4	FIFO receive operation in CSM and FSM	112
3.5.5	Subchannel example for SM	113
3.5.6	Subchannel example for CSM	118
4	FIFO handling and HDLC controller	125
4.1	Overview	126
4.2	FIFO counters	126
4.3	FIFO size setup	128
4.4	FIFO operation	128
4.4.1	HDLC transmit FIFOs	128
4.4.2	Automatic D-channel frame repetition (for S/T in TE mode only)	129
4.4.3	HDLC receive FIFOs	130
4.4.4	Transparent mode of XHFC-2SU	131
4.5	Register description	133
4.5.1	Write only registers	133
4.5.2	Read only registers	138
4.5.3	Read / write registers	145
5	Universal ISDN Port	155
5.1	General overview of the S/T and U _p interfaces	156
5.1.1	Array registers and multiregisters	156
5.1.2	Block diagram of the Universal ISDN Port module	156
5.2	S/T interface description	158
5.2.1	Overview	158

5.2.2	Frame Structure	158
5.2.3	Multiframe structure	159
5.2.4	Data transmission	159
5.2.5	INFO signals	161
5.2.6	State machine	163
5.2.7	Clock synchronization	166
5.2.8	External circuitries	168
5.2.8.1	External receive circuitry	168
5.2.8.2	External transmit circuitry	169
5.2.8.3	Transformer and ISDN jack connection	170
5.2.9	S/T transformers	171
5.3	U _p interface description	175
5.3.1	Overview	175
5.3.2	Frame Structure	175
5.3.3	Superframe structure	175
5.3.4	Data transmission	176
5.3.5	INFO signals	177
5.3.6	State machine	179
5.3.7	Clock synchronization	182
5.3.8	External circuitry	184
5.3.9	U _p transformers	185
5.4	Common features of the S/T and U _p interfaces	186
5.4.1	Direct data access for test purposes	186
5.4.2	Clock synchr. with several TEs connected to different central office switches	187
5.4.3	Combined S/T and U _p circuitry	192
5.5	Register description	193
5.5.1	Write only registers	193
5.5.2	Read only registers	209
6	PCM interface	219
6.1	PCM interface function	220
6.2	PCM data flow	221
6.3	PCM initialization	223
6.4	PCM timing	223
6.4.1	Mode selection	223
6.4.2	Master mode	223
6.4.3	Slave mode	223
6.5	PCM clock synchronization	228

6.5.1	Overview	228
6.5.2	Manual or automatic synchronization source selection	228
6.5.3	PLL programming for F0IO generation	228
6.5.4	Manual PLL adjustment	228
6.5.5	C2IO signal	230
6.5.6	SYNC_O and FSC_RX synchronization signals	230
6.5.7	Synchronous 512 kHz output signal	230
6.5.8	Application examples for XHFC-2SU synchronization schemes	231
6.5.8.1	System with internal PCM bus expanded by ISDN interfaces	231
6.5.8.2	Application with multiple XHFC-2SU	232
6.6	Multiframe / superframe synchronization to the PCM interface	235
6.6.1	Overview	235
6.6.2	Frame counter	235
6.6.3	Synchronization input signal	235
6.6.4	Synchronization output signal (F0IO pulse modification)	238
6.6.5	Line interface synchronization	239
6.7	External CODECs	240
6.8	GCI / IOM-2 mode	242
6.8.1	Overview	242
6.8.2	GCI frame structure	242
6.8.3	GCI register programming	243
6.8.3.1	Enable CGI functionality	243
6.8.3.2	Monitor channel	244
6.8.3.3	Command / indication bits (C/I-channel)	244
6.8.3.4	Examples for GCI frame embedding in the PCM data structure	244
6.8.4	GCI protocol	249
6.8.4.1	XHFC-2SU transmit procedure	249
6.8.4.2	XHFC-2SU receive procedure	250
6.8.4.3	Receiver abort	252
6.9	Register description	253
6.9.1	Write only registers	253
6.9.2	Read only registers	271
6.9.3	Read / write register	274
7	Pulse width modulation (PWM) outputs	275
7.1	Overview	276
7.2	Standard PWM usage	276
7.3	Alternative PWM usage	276

7.4	Register description	277
8	Bit Error Rate Test (BERT)	279
8.1	BERT functionality	280
8.2	BERT transmitter	280
8.3	BERT receiver	280
8.4	Register description	284
8.4.1	Write only registers	284
8.4.2	Read only registers	285
9	Clock, PLL, reset, interrupt, timer and watchdog	287
9.1	Clock	288
9.1.1	Clock output	288
9.1.2	Clock distribution	288
9.1.3	Clock oscillator circuitry	290
9.1.3.1	Frequency accuracy	290
9.1.3.2	Pierce oscillator	290
9.1.3.3	3rd overtone oscillator	291
9.1.3.4	Crystal oscillator circuitry	291
9.1.3.5	Several XHFC-2SU with a single oscillator circuitry	291
9.2	Phase locked loop (PLL)	293
9.2.1	Overview	293
9.2.2	PLL structure	293
9.2.3	PLL operation	294
9.2.4	Supply noise requirements for PLL locking	294
9.2.5	PLL configuration	295
9.3	Reset	298
9.4	Interrupt	299
9.4.1	Common features	299
9.4.2	ST/U _p interface interrupt	299
9.4.3	FIFO interrupt	299
9.4.4	Miscellaneous interrupts	301
9.4.4.1	Line interface frequency slip interrupt	301
9.4.4.2	Timer interrupt	302
9.4.4.3	Processing / non-processing interrupt	302
9.4.4.4	Command / indication interrupt (GCI interface)	303
9.4.4.5	Wakeup interrupt	303
9.4.4.6	Interrupt for GCI monitor byte transmission	303
9.4.4.7	Interrupt after GCI monitor byte received	303

9.5	Watchdog reset	304
9.6	Register description	305
9.6.1	Write only registers	305
9.6.2	Read only registers	313
9.6.3	Read / write registers	322
10	General purpose I/O pins (GPIO)	323
10.1	GPIO functionality	324
10.2	GPIO output voltage	324
10.3	Activation state F7 / G3 signalling	327
10.4	Register description	328
10.4.1	Write only registers	328
10.4.2	Read only registers	335
11	Electrical characteristics	337
12	XHFC-2SU package dimensions	341
13	XHFC-2SU address decoding erratum	343
13.1	Fault description	344
13.2	Work-Around	344
	References	349
	List of register and bitmap abbreviations	351

List of Figures

1.1	XHFC-2SU block diagram	31
1.2	XHFC-2SU pinout in parallel processor interface mode	34
1.3	XHFC-2SU pinout in serial processor interface mode	35
2.1	Bus interface read access in mode 2 (Motorola) and mode 3 (Intel)	46
2.2	Bus interface write access in mode 2 (Motorola) and mode 3 (Intel)	49
2.3	Bus interface read access in modes 2m and 3m (multiplexed)	52
2.4	Bus interface write access in mode 2m and 3m (multiplexed)	54
2.5	8 bit Motorola processor circuitry example (mode 2)	55
2.6	8 bit Motorola processor circuitry example (mode 2m)	55
2.7	8 bit Intel processor circuitry example (mode 3)	56
2.8	8 bit Intel processor circuitry example (mode 3m)	56
2.9	16 bit SPI write transaction (R = '0', one data byte)	59
2.10	16 bit SPI read transaction (R = '1', one data byte)	59
2.11	40 bit SPI write transaction (R = '0', four data bytes)	60
2.12	40 bit SPI read transaction (R = '1', four data bytes)	60
2.13	Split 40 bit SPI read transaction (R = '1')	60
2.14	Register write access (transaction sequence)	62
2.15	Multiple write accesses to the same register	62
2.16	Register write access with a 40 bit transaction	62
2.17	Register read access (transaction sequence)	63
2.18	Multiple read accesses to the same register	64
2.19	Register read access with a 40 bit transaction	64
2.20	SPI timing diagram for data write transactions	65
2.21	SPI timing diagram for data read transactions	66
2.22	SPI connection circuitry	68
3.1	Data flow block diagram	75
3.2	Areas of FIFO oriented, HFC-channel oriented and PCM time slot oriented numbering	77
3.3	The flow controller in transmit operation	79

3.4	The flow controller in receive FIFO operation	80
3.5	Overview of the assigner programming	84
3.6	SM example	88
3.7	HFC-channel assigner in CSM	93
3.8	CSM example	94
3.9	HFC-channel assigner in FSM	99
3.10	FSM list processing	100
3.11	FSM list programming	101
3.12	FSM example	102
3.13	General structure of the subchannel processor	108
3.14	Location of the subchannel parts A and B in the data flow diagram	108
3.15	Part A of the subchannel processor	110
3.16	Part B of the subchannel processor	112
3.17	SM example with subchannel processor	113
3.18	CSM example with subchannel processor	118
4.1	FIFO organization	129
4.2	FIFO data organization in HDLC mode	130
5.1	Overview of the Universal ISDN Port module	157
5.2	Frame structure at reference point S and T	158
5.3	S/T frame composition for B1-, B2-, D-channel and multiframe bits	161
5.4	S/T frame decomposition for B1-, B2-, D-, E-channel and multiframe bits	162
5.5	S/T clock synchronization	167
5.6	External S/T receive circuitry for TE and NT mode	168
5.7	External S/T transmit circuitry for TE and NT mode	169
5.8	Transformer and connector circuitry in TE mode	170
5.9	Transformer and connector circuitry in NT mode	170
5.10	U _p interface frame structure	175
5.11	U _p frame composition for B1-, B2-, D-channel and superframe bits	177
5.12	U _p frame decomposition for B1-, B2-, D-channel and superframe bits	178
5.13	U _p clock synchronization	183
5.14	External U _p circuitry for TE and LT mode	184
5.15	Synchronization scenario with TEs connected to several central office switches	187
5.16	Synchronization registers (detail of Figure 5.15)	188
5.17	Timing example of one transmit and one receive transmission	189
5.18	Data transmission with $f_{FSC_TX_1} > f_{FSC_TX_0}$	190
5.19	Data transmission with $f_{FSC_TX_1} < f_{FSC_TX_0}$	191

6.1	PCM data flow for transmit and receive time slots	221
6.2	Global setting for all PCM time slots (detail of Figure 6.1)	222
6.3	PCM timing for master mode	224
6.4	PCM timing for slave mode	226
6.5	PCM clock synchronization	229
6.6	Expanding an existing system with ISDN ports (XHFC-2SU as PCM slave)	231
6.7	Expanding an existing system with ISDN ports (XHFC-2SU as PCM master)	232
6.8	Multiple XHFC-2SU synchronized with an open loop of SYNC_O/SYNC_I	233
6.9	Multiple XHFC-2SU synchronized with a closed loop of SYNC_O/SYNC_I	234
6.10	Multiframe / superframe synchronization to the PCM interface	236
6.11	Timing specification of the SYNC_IN change detection	237
6.12	FOIO pulse modification	238
6.13	Example for two CODEC enable signal shapes	240
6.14	Single channel GCI format	243
6.15	GCI application example 1 (XHFC-2SU is PCM master and GCI master)	245
6.16	GCI application example 2 (XHFC-2SU is PCM slave and GCI master)	247
6.17	GCI application example 3 (XHFC-2SU is PCM slave and GCI slave)	247
6.18	GCI protocol for monitor bytes transmission	249
6.19	GCI protocol for monitor bytes receiving	251
8.1	BERT transmitter block diagram	281
8.2	BERT receiver block diagram	282
9.1	CLK_OUT generation and clock distribution	289
9.2	Standard XHFC-2SU quartz circuitry	290
9.3	Clock distribution with only one quartz circuitry	292
9.4	PLL block diagram	294
9.5	Interrupt output	299
9.6	ST/U _p interface interrupt	300
9.7	Enable FIFO interrupt condition with V_FIFO_IRQ_EN	300
9.8	FIFO interrupt	301
9.9	Miscellaneous interrupts	302
9.10	Wakeup interrupt	303
10.1	GPIO block diagram	325
10.2	Activation state signalling exemplarily shown with GPIO0	327
12.1	XHFC-2SU package dimensions	342

List of Tables

2.1	Overview of the XHFC-2SU bus interface registers	41
2.2	Microprocessor access types	42
2.3	Overview of the parallel processor interface pins	43
2.4	Parallel processor interface mode selection	44
2.5	Pins and signal names of the parallel processor interface modes	44
2.6	Overview of accesses in parallel microprocessor interface mode	47
2.7	Timing diagrams of the parallel microprocessor interface	47
2.8	Symbols of read accesses in Figure 2.1	48
2.9	Symbols of write accesses in Figure 2.2	50
2.10	Symbols of read accesses in Figure 2.3	51
2.11	Symbols of write accesses in Figures 2.4	53
2.12	Overview of the SPI interface pins	57
2.13	SPI control byte with A = '0'	58
2.14	SPI control byte with A = '1'	58
2.15	SPI control commands for register read and write operations	58
2.16	SPI data rates	64
2.17	Symbols of write access in Figure 2.20	66
2.18	Symbols of read access in Figure 2.21	67
3.1	Flow controller connectivity	81
3.2	V_DATA_FLOW programming values for bidirectional connections	81
3.3	Line interface assigner	83
3.4	Index registers of the FIFO array registers (sorted by address)	86
3.5	List specification of the example in Figure 3.12	103
3.6	Subchannel processing according to Figure 3.17 (SM ① TX, transparent mode) . . .	115
3.7	Subchannel processing according to Figure 3.17 (SM ① RX, transparent mode) . . .	115
3.8	Subchannel processing according to Figure 3.17 (SM ② TX, HDLC mode)	117
3.9	Subchannel processing according to Figure 3.17 (SM ② RX, HDLC mode)	117
3.10	Subchannel processing according to Figure 3.18 (CSM ① TX, transparent mode) . .	120
3.11	Subchannel processing according to Figure 3.18 (CSM ① RX, transparent mode) . .	121
3.12	Subchannel processing according to Figure 3.18 (CSM ② TX, HDLC mode)	124

3.13	Subchannel processing according to Figure 3.18 (CSM ② RX, HDLC mode)	124
4.1	Overview of the XHFC-2SU FIFO registers	125
4.2	FIFO size setup	128
5.1	Overview of the ST/U _p interface registers	155
5.2	Legend for Figure 5.2	159
5.3	Multiframe structure of the Q- and S-bits	160
5.4	S/T interface activation / deactivation layer 1 matrix for NT mode	164
5.5	S/T interface activation / deactivation layer 1 matrix for TE mode	165
5.6	S/T transformer part numbers and manufacturers	171
5.7	Legend for Figure 5.10	176
5.8	Superframe construction	176
5.9	U _p interface activation / deactivation layer 1 matrix for LT mode	180
5.10	U _p interface activation / deactivation layer 1 matrix for TE mode	181
5.11	Up transformer part numbers and manufacturers	185
5.12	Symbols of Figures 5.17	188
6.1	Overview of the XHFC-2SU PCM interface registers	219
6.2	PCM master mode	220
6.3	PCM data flow programming	222
6.4	Symbols of PCM timing for master mode in Figure 6.3	225
6.5	Symbols of PCM timing for slave mode in Figure 6.4	227
6.6	Symbols of the SYNC_IN change detection timing specification in Figure 6.11	238
6.7	Selection of the F0IO pulse characteristic (see also Figure 6.12)	239
6.8	Legend of figure 6.14	243
6.9	Programming procedure for application example 1 according to Figure 6.15	246
6.10	Programming procedure for application example 3 according to Figure 6.17	248
6.11	Rules for the handshake signals MX _{TX} and MR _{TX}	250
6.12	Rules for the handshake signals MX _{RX} and MR _{RX}	252
7.1	Overview of the XHFC-2SU PWM registers	275
8.1	Overview of the XHFC-2SU BERT registers	279
9.1	Overview of clock, PLL, reset, timer and watchdog registers	287
9.2	PLL setup examples (P, M, N, S) with ISDN related frequencies for f _{out}	296
9.3	PLL setup examples (P, M, N, S) with ISDN related frequencies for f _{in}	297
9.4	XHFC-2SU reset groups	298
10.1	Overview of the XHFC-2SU general purpose I/O registers	323

10.2 GPIO pins	326
10.3 State F7/G3 reporting	327
13.1 Involved registers of the address decoding fault	344
13.2 Detailed register list and work-around description	345

List of Registers



Please note !

Register addresses are assigned independently for write and read access; i.e. in many cases there are different registers for write and read access with the same address. Only registers with the same meaning and bitmap structure in both write and read directions are declared to be read / write.

It is important to distinguish between *registers*, *array registers* and *multi-registers*.

Array registers have multiple instances and are indexed by a number. This index is either the FIFO number (R_FIFO with 15 indexed registers), the PCM time slot number (R_SLOT with 1 indexed register) or the ST/U_p interface number (R_SU_SEL with 21 indexed registers). Array registers have equal name, bitmap structure and meaning for every instance.

Multi-registers have multiple instances, too, but they are selected by a bitmap value. With this value, different registers can be selected with the same address. Multi-register addresses are 0x15 (11 instances selected by R_PCM_MD0), 0x0F (2 instances selected by R_FIFO_MD) and 0x35 (2 instances selected by A_ST_CTRL3/A_UP_CTRL3) for XHFC-2SU. Multi-registers have different names, bitmap structure and meaning for each instance.

The first letter of array register names is 'A_ ...' whereas all other registers begin with 'R_ ...'. The index of array registers and multi-registers has to be specified in the appropriate register.

Registers sorted by name

Reset group: H = Hardware reset
 0 = Global software reset
 1 = HFC reset
 2 = PCM reset
 3 = Line interface reset



Please note !

See explanation of register types on page 21.

See Table 9.4 on page 298 for a detailed reset group explanation.

Write only registers:

Address	Name	Reset group	Page	Address	Name	Reset group	Page
0x3C	A_B1_TX[ST/Up]	-	206	0x14	R_PCM_MD0	H, 0, 2	254
0x3D	A_B2_TX[ST/Up]	-	206	0x15	R_PCM_MD1	H, 0, 2	259
0x3F	A_BAC_S_TX[ST/Up]	-	208	0x15	R_PCM_MD2	H, 0, 2	261
0x3E	A_D_TX[ST/Up]	-	207	0x50	R_PLL_CTRL	H	312
0x0E	A_INC_RES_FIFO[FIFO]	H, 0, 1, 2, 3	136	0x1E	R_PWM_CFG	H, 0	277
0x36	A_MS_DF[ST/Up]	H, 0, 3	204	0x46	R_PWM_MD	H, 0	278
0x34	A_MS_TX[ST/Up]	H, 0, 3	200	0x38	R_PWM0	H, 0	277
0x35	A_ST_CTRL3[ST/Up]	H, 0, 3	201	0x39	R_PWM1	H, 0	278
0x37	A_SU_CLK_DLY[ST/Up]	H, 0, 3	205	0x08	R_RAM_ADDR	H, 0	71
0x31	A_SU_CTRL0[ST/Up]	H, 0, 3	195	0x09	R_RAM_CTRL	H, 0	71
0x32	A_SU_CTRL1[ST/Up]	H, 0, 3	197	0x15	R_SH0H	H, 0, 2	264
0x33	A_SU_CTRL2[ST/Up]	H, 0, 3	198	0x15	R_SH0L	H, 0, 2	264
0x30	A_SU_WR_STA[ST/Up]	H, 0, 3	194	0x15	R_SH1H	H, 0, 2	265
0x35	A_UP_CTRL3[ST/Up]	H, 0, 3	202	0x15	R_SH1L	H, 0, 2	264
0x1B	R_BERT_WD_MD	H, 0	284	0x15	R_SL_SEL0	H, 0, 2	255
0x28	R_CI_TX	H, 0, 2	267	0x15	R_SL_SEL1	H, 0, 2	256
0x00	R_CIRM	H	305	0x15	R_SL_SEL7	H, 0, 2	256
0x02	R_CLK_CFG	H	307	0x10	R_SLOT	H, 0, 2	253
0x01	R_CTRL	H	70	0x12	R_SU_IRQMSK	H, 0, 3	309
0x0D	R_FIFO_MD	H	135	0x4D	R_SU_LED_CTRL	H, 0	334
0x0C	R_FIFO_THRES	H, 0, 1, 2, 3	134	0x16	R_SU_SEL	H, 0, 3	193
0x0F	R_FIFO	H, 0, 1	137	0x17	R_SU_SYNC	H, 0, 3	266
0x0B	R_FIRST_FIFO	H, 0, 1	133	0x1A	R_TI_WD	H, 0	311
0x0F	R_FSM_IDX	H, 0, 1	137				
0x29	R_GCI_CFG0	H, 0, 2	268				
0x2A	R_GCI_CFG1	H, 0, 2	270				
0x4A	R_GPIO_EN0	H, 0	332				
0x42	R_GPIO_EN1	H, 0	329				
0x47	R_GPIO_EN2	H, 0	330				
0x48	R_GPIO_OUT0	H, 0	331				
0x40	R_GPIO_OUT1	H, 0	328				
0x45	R_GPIO_OUT2	H, 0	330				
0x44	R_GPIO_SEL_BL	H, 0	329				
0x4C	R_GPIO_SEL	H, 0	333				
0x13	R_IRQ_CTRL	H, 0	310				
0x11	R_MISC_IRQMSK	H	308				
0x2B	R_MON_TX	H, 0, 2	270				
0x15	R_MSS0	H, 0, 2	257				
0x15	R_MSS1	H, 0, 2	263				

Read only registers:

Address	Name	Reset group	Page
0x3C	A_B1_RX[ST/Up]	–	215
0x3D	A_B2_RX[ST/Up]	–	216
0x3E	A_D_RX[ST/Up]	–	217
0x3F	A_E_RX[ST/Up]	–	218
0x0C	A_F1[FIFO]	H, 0, 1	138
0x0D	A_F2[FIFO]	H, 0, 1	139
0x0E	A_FIFO_STA[FIFO]	H, 0, 1	140
0x34	A_MS_RX[ST/Up]	H, 0, 3	213
0x32	A_SU_DLYH[ST/Up]	–	212
0x31	A_SU_DLYL[ST/Up]	–	211
0x30	A_SU_RD_STA[ST/Up]	H, 0, 3	210
0x35	A_SU_STA[ST/Up]	H, 0, 3	214
0x14	A_USAGE[FIFO]	H, 0, 1	141
0x04	A_Z1[FIFO]	H, 0, 1	138
0x06	A_Z2[FIFO]	H, 0, 1	138
0x13	R_AF0_OVIEW	H, 0, 3	209
0x1B	R_BERT_ECH	H, 0, 1	286
0x1A	R_BERT_ECL	H, 0, 1	285
0x17	R_BERT_STA	H, 0, 1	285
0x16	R_CHIP_ID	H	72
0x1F	R_CHIP_RV	H	72
0x28	R_CI_RX	–	272
0x19	R_F0_CNTH	H, 0, 1	271
0x18	R_F0_CNTL	H, 0, 1	271
0x20	R_FIFO_BL0_IRQ	H, 0, 1	318
0x21	R_FIFO_BL1_IRQ	H, 0, 1	319
0x22	R_FIFO_BL2_IRQ	H, 0, 1	320
0x23	R_FIFO_BL3_IRQ	H, 0, 1	321
0x24	R_FILL_BL0	H, 0, 1	141
0x25	R_FILL_BL1	H, 0, 1	142
0x26	R_FILL_BL2	H, 0, 1	143
0x27	R_FILL_BL3	H, 0, 1	144
0x29	R_GCI_STA	H, 0, 2	273
0x48	R_GPIO_IN0	–	336
0x40	R_GPIO_IN1	–	335
0x45	R_GPIO_IN2	–	335
0x88	R_INT_DATA	–	73
0x10	R_IRQ_OVIEW	H, 0, 1	313
0x11	R_MISC_IRQ	H, 0, 1	315
0x2A	R_MON_RX	–	273
0x50	R_PLL_STA	H	321
0x15	R_RAM_USE	–	72
0x1D	R_SL_MAX	–	271
0x1C	R_STATUS	H, 0, 3	317
0x12	R_SU_IRQ	H, 0	316

Read / Write registers:

Address	Name	Reset group	Page
0xF4	A_CH_MSK[FIFO]	H, 0, 1	146
0xFC	A_CHANNEL[FIFO]	H, 0, 1	150
0xFA	A_CON_HDLC[FIFO]	H, 0, 1	147
0xFF	A_FIFO_CTRL[FIFO]	H, 0, 1	152
0x84	A_FIFO_DATA_NOINC[FIFO]	–	145
0x80	A_FIFO_DATA[FIFO]	–	145
0xFD	A_FIFO_SEQ[FIFO]	H, 0, 1	151
0xD0	A_SL_CFG[SLOT]	H, 0, 2	274
0xFB	A_SUBCH_CFG[FIFO]	H, 0, 1	149
0x52	R_PLL_N	H	322
0x51	R_PLL_P	H	322
0x53	R_PLL_S	H	322
0xC0	R_RAM_DATA	–	74

Registers sorted by address

Reset group: H = Hardware reset
 0 = Global software reset
 1 = HFC reset
 2 = PCM reset
 3 = Line interface reset



Please note !

See explanation of register types on page 21.

See Table 9.4 on page 298 for a detailed reset group explanation.

Write only registers:

Address	Name	Reset group	Page
0x00	R_CIRM	H	305
0x01	R_CTRL	H	70
0x02	R_CLK_CFG	H	307
0x08	R_RAM_ADDR	H, 0	71
0x09	R_RAM_CTRL	H, 0	71
0x0B	R_FIRST_FIFO	H, 0, 1	133
0x0C	R_FIFO_THRES	H, 0, 1, 2, 3	134
0x0D	R_FIFO_MD	H	135
0x0E	A_INC_RES_FIFO[FIFO]	H, 0, 1, 2, 3	136
0x0F	R_FSM_IDX	H, 0, 1	137
0x0F	R_FIFO	H, 0, 1	137
0x10	R_SLOT	H, 0, 2	253
0x11	R_MISC_IRQMSK	H	308
0x12	R_SU_IRQMSK	H, 0, 3	309
0x13	R_IRQ_CTRL	H, 0	310
0x14	R_PCM_MD0	H, 0, 2	254
0x15	R_MSS0	H, 0, 2	257
0x15	R_MSS1	H, 0, 2	263
0x15	R_PCM_MD1	H, 0, 2	259
0x15	R_PCM_MD2	H, 0, 2	261
0x15	R_SH0H	H, 0, 2	264
0x15	R_SH1H	H, 0, 2	265
0x15	R_SH0L	H, 0, 2	264
0x15	R_SH1L	H, 0, 2	264
0x15	R_SL_SEL0	H, 0, 2	255
0x15	R_SL_SEL1	H, 0, 2	256
0x15	R_SL_SEL7	H, 0, 2	256
0x16	R_SU_SEL	H, 0, 3	193
0x17	R_SU_SYNC	H, 0, 3	266
0x1A	R_TI_WD	H, 0	311
0x1B	R_BERT_WD_MD	H, 0	284
0x1E	R_PWM_CFG	H, 0	277
0x28	R_CI_TX	H, 0, 2	267
0x29	R_GCI_CFG0	H, 0, 2	268
0x2A	R_GCI_CFG1	H, 0, 2	270
0x2B	R_MON_TX	H, 0, 2	270
0x30	A_SU_WR_STA[ST/Up]	H, 0, 3	194
0x31	A_SU_CTRL0[ST/Up]	H, 0, 3	195
0x32	A_SU_CTRL1[ST/Up]	H, 0, 3	197
0x33	A_SU_CTRL2[ST/Up]	H, 0, 3	198
0x34	A_MS_TX[ST/Up]	H, 0, 3	200
0x35	A_ST_CTRL3[ST/Up]	H, 0, 3	201
0x35	A_UP_CTRL3[ST/Up]	H, 0, 3	202
0x36	A_MS_DF[ST/Up]	H, 0, 3	204
0x37	A_SU_CLK_DLY[ST/Up]	H, 0, 3	205
0x38	R_PWM0	H, 0	277
0x39	R_PWM1	H, 0	278
0x3C	A_B1_TX[ST/Up]	-	206
0x3D	A_B2_TX[ST/Up]	-	206
0x3E	A_D_TX[ST/Up]	-	207
0x3F	A_BAC_S_TX[ST/Up]	-	208
0x40	R_GPIO_OUT1	H, 0	328
0x42	R_GPIO_EN1	H, 0	329
0x44	R_GPIO_SEL_BL	H, 0	329
0x45	R_GPIO_OUT2	H, 0	330
0x46	R_PWM_MD	H, 0	278
0x47	R_GPIO_EN2	H, 0	330
0x48	R_GPIO_OUT0	H, 0	331
0x4A	R_GPIO_EN0	H, 0	332
0x4C	R_GPIO_SEL	H, 0	333
0x4D	R_SU_LED_CTRL	H, 0	334
0x50	R_PLL_CTRL	H	312

Read only registers:

Address	Name	Reset group	Page
0x04	A_Z1[FIFO]	H, 0, 1	138
0x06	A_Z2[FIFO]	H, 0, 1	138
0x0C	A_F1[FIFO]	H, 0, 1	138
0x0D	A_F2[FIFO]	H, 0, 1	139
0x0E	A_FIFO_STA[FIFO]	H, 0, 1	140
0x10	R_IRQ_OVIEW	H, 0, 1	313
0x11	R_MISC_IRQ	H, 0, 1	315
0x12	R_SU_IRQ	H, 0	316
0x13	R_AF0_OVIEW	H, 0, 3	209
0x14	A_USAGE[FIFO]	H, 0, 1	141
0x15	R_RAM_USE	-	72
0x16	R_CHIP_ID	H	72
0x17	R_BERT_STA	H, 0, 1	285
0x18	R_F0_CNTL	H, 0, 1	271
0x19	R_F0_CNTH	H, 0, 1	271
0x1A	R_BERT_ECL	H, 0, 1	285
0x1B	R_BERT_ECH	H, 0, 1	286
0x1C	R_STATUS	H, 0, 3	317
0x1D	R_SL_MAX	-	271
0x1F	R_CHIP_RV	H	72
0x20	R_FIFO_BL0_IRQ	H, 0, 1	318
0x21	R_FIFO_BL1_IRQ	H, 0, 1	319
0x22	R_FIFO_BL2_IRQ	H, 0, 1	320
0x23	R_FIFO_BL3_IRQ	H, 0, 1	321
0x24	R_FILL_BL0	H, 0, 1	141
0x25	R_FILL_BL1	H, 0, 1	142
0x26	R_FILL_BL2	H, 0, 1	143
0x27	R_FILL_BL3	H, 0, 1	144
0x28	R_CI_RX	-	272
0x29	R_GCI_STA	H, 0, 2	273
0x2A	R_MON_RX	-	273
0x30	A_SU_RD_STA[ST/Up]	H, 0, 3	210
0x31	A_SU_DLYL[ST/Up]	-	211
0x32	A_SU_DLYH[ST/Up]	-	212
0x34	A_MS_RX[ST/Up]	H, 0, 3	213
0x35	A_SU_STA[ST/Up]	H, 0, 3	214
0x3C	A_B1_RX[ST/Up]	-	215
0x3D	A_B2_RX[ST/Up]	-	216
0x3E	A_D_RX[ST/Up]	-	217
0x3F	A_E_RX[ST/Up]	-	218
0x40	R_GPIO_IN1	-	335
0x45	R_GPIO_IN2	-	335
0x48	R_GPIO_IN0	-	336
0x50	R_PLL_STA	H	321
0x88	R_INT_DATA	-	73

Read / Write registers:

Address	Name	Reset group	Page
0x51	R_PLL_P	H	322
0x52	R_PLL_N	H	322
0x53	R_PLL_S	H	322
0x80	A_FIFO_DATA[FIFO]	-	145
0x84	A_FIFO_DATA_NOINC[FIFO]-	-	145
0xC0	R_RAM_DATA	-	74
0xD0	A_SL_CFG[SLOT]	H, 0, 2	274
0xF4	A_CH_MSK[FIFO]	H, 0, 1	146
0xFA	A_CON_HDLC[FIFO]	H, 0, 1	147
0xFB	A_SUBCH_CFG[FIFO]	H, 0, 1	149
0xFC	A_CHANNEL[FIFO]	H, 0, 1	150
0xFD	A_FIFO_SEQ[FIFO]	H, 0, 1	151
0xFF	A_FIFO_CTRL[FIFO]	H, 0, 1	152

About this data sheet and Cologne Chip technical support

This data sheet covers all the features of XHFC-2SU. The reader who absorbs the information in this data sheet will gain a deep and broad understanding of XHFC-2SU microchip.

However, the hurried reader needs not to read the complete data sheet. Every chapter comprises just one topic. What's not needed in the focus of a target application can be skipped over while reading this data sheet.

Organization of this data sheet

Chapters start with a short overview. They typically contain both the electrical description and the programming features of the corresponding subject. Finally, chapters end with a register description.

Links between chapters are mentioned in the text.

Development tools

Driver software plays an important role in all ISDN projects. For this reason we offer more than the hardware:

- An evaluation board of XHFC-2SU is available. This can be connected to the target microprocessor system via a flat ribbon cable. It is planned to make the evaluation board accessible via a PCI bridge board to a standard PC environment.
- A demo layer 1 driver as source code as well as open-source (GPL) Linux drivers are available.
- There are also header files with all registers and their bitmaps available for programming language C. Please ask the Cologne Chip support team for more information and file delivery.

Visit our web site

Our web site (<http://www.colognechip.com>) contains a download area for all Cologne Chip data sheets. Additional information is given concerning transformers, drivers etc. on the website, too.

By having broad knowledge about ISDN applications, Cologne Chip supports any project individually. Please contact our support team.

Chapter overview

Chapter “General description” (1) begins with an overview to XHFC-2SU, especially general block diagrams and a feature list. Pinout diagrams for the different microprocessor bus interfaces and a detailed list of all pins complete this chapter.

Chapter “Microprocessor bus interface”: (2) XHFC-2SU supports several processor interfaces which are explained in this chapter. This includes signal and timing characteristics as well as register access explanation and typical connection circuitries. (Separated interface modes, read only the section which deals with the interface mode of your interest, prerequisite knowledge of the chosen interface mode is strongly recommended.)

Chapter “XHFC-2SU data flow” (3) starts with the data processing explanation. This chapter deals with the data flow concept which connects all the data interfaces that are explained in the following chapters. (It is recommended to have at least a basic comprehension to this topic, as it connects several important parts of XHFC-2SU.)

Chapter “FIFO handling and HDLC controller” (4) covers the host side of the data flow. This includes both the HDLC controller and the FIFOs. (Should be read because FIFOs and the HDLC controller is typically used in every application.)

Chapter “Universal ISDN Port”: (5) XHFC-2SU has two line interfaces which can be configured either in S/T or U_p mode. This chapter explains the data structures, clock synchronization and external circuitries. (The most important interface, should be read, prerequisite knowledge of ISDN protocol is strongly recommended.)

Chapter “PCM interface”: (6) The last interface which deals with the data flow described in chapter 3 is the PCM interface. Beneath other, an important topic of this chapter are synchronization features of XHFC-2SU. (Read only when used, but don't skip the overview in this chapter even if the PCM interface is not used!)

Chapter “Pulse width modulation (PWM) outputs”: (7) This chapter can be skipped if the PWM interface is not used.

Chapter “Bit Error Rate Test (BERT)”: (8) This chapter can be skipped if the BERT functionality is not used.

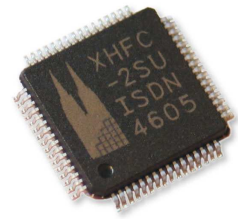
Chapter “Clock, PLL, reset, interrupt, timer and watchdog” (9) explains clock generation and distribution, PLL programming, reset functions and interrupt capabilities. (Must be read.)

Chapter “Electrical characteristics”: (11) Some information about the electrical characteristics of XHFC-2SU are given in this chapter. (Information for hardware design.)

Chapter “XHFC-2SU package dimensions” (12) shows the XHFC-2SU package dimensions. (Information for hardware design.)

General remarks to notations

1. The decimal point is written as a point (e.g. 1.23). Thousands separators are written with thin space.
2. Numerical values have different notations for various number systems; e.g. the hexadecimal value 0xC9 is '1100 1001' in binary and 201 in decimal notation.
3. The prefix 'kilo' is written k for the meaning of 1000 and it is written K for the meaning of 1024.
4. The first letter of register names indicates the type: 'R_ ...' is a register or multi-register, while 'A_ ...' is an array register.



Chapter 1

General description

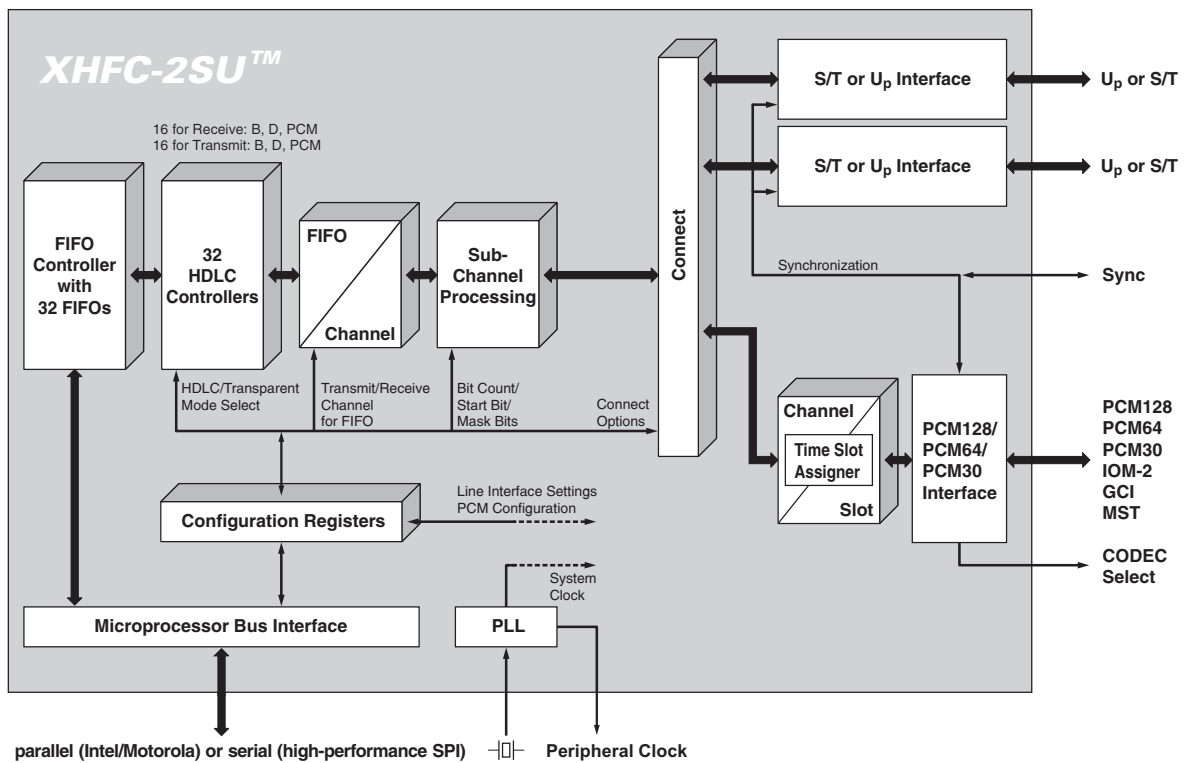


Figure 1.1: XHFC-2SU block diagram

1.1 System overview

XHFC-2SU is a ISDN transceiver chip for two ISDN S/T or U_{pN}/U_{p0} ¹ Basic Rate Interfaces with integrated HDLC controllers for all kinds of BRI equipment, such as

- VoIP gateways / VoIP routers
- Integrated Access Devices (IAD)
- ISDN SOHO PABX
- ISDN phones / system phones
- ISDN least cost routers
- ISDN LAN routers
- ISDN test equipment
- Call recording
- S/T-to- U_p converters (private NTs)
- U_p repeaters

The integrated microprocessor bus interface of XHFC-2SU can be configured to 8 bit parallel microprocessor interface or serial processor interface (SPI). A PCM128 / PCM64 / PCM30 interface for CODEC or inter-chip connection is also integrated. The deep FIFOs of XHFC-2SU are realized with an internal SRAM.

¹ U_{pN}/U_{p0} in the following referred to as U_p .

1.2 Features

Line interfaces

- 2 ISDN interfaces, both selectable as S/T or U_{pN}/U_{p0} interfaces (Universal ISDN Ports)
- S/T ISDN interfaces in TE and NT mode conform to ITU-T I.430 and TBR 3 [9, 4]
- U_p signal range exceeding U_{pN}/U_{p0} specification [3]
- Simple external line interface circuitry

HDLC-controller and FIFO controller

- Universal HDLC controller for all B-, D- and E-channels, can also be used for PCM time slots
- Transparent mode and data rate independently selectable for all FIFOs
- Up to 16 FIFOs for transmit and receive data each, FIFO size configurable from 64 up to 256 bytes per FIFO, maximum 7 HDLC frames per FIFO
- B- and D-channels can be combined for higher data rate to 128 kBit/s (2B) or 144 kBit/s (2B+D) per line interface
- Bit Error Rate Test (BERT) with transmitter and receiver
- Programmable data flow to connect FIFOs, ST/ U_p interfaces and PCM time slots with each other

PCM interface

- PCM128 / PCM64 / PCM30 interface configurable to MST (MVIP)² or Siemens IOMTM-2 and Motorola GCI (monitor and C/I-channel support) for interchip connection or external CODECs
- Programmable PCM time slot assigner for 16 channels in transmit and receive direction each (switch matrix for PCM)
- H.100 data rate supported on PCM bus
- Flexible PCM synchronization options implemented, synchronization input and output signals available

Microprocessor bus interface

- Improved 8 bit parallel microprocessor interface compatible to Motorola bus and Siemens / Intel bus, multiplexed and non-multiplexed modes supported
- High performance serial processor interface (SPI), up to 16 XHFC devices addressable with one /SPISEL signal
- Auto-configure mode for repeater applications without microcontroller (only external EEPROM needed)

Miscellaneous features

- Flexible interrupt controller, timer and watchdog with interrupt capability
- Programmable PLL with big range of clock frequencies for general purpose usage (can also be used to generate the internal system clock)
- 6 GPIO pins can be used instead of every unused line interface, further 8 GPIOs can be enabled separately as second pin function
- 2 general purpose pulse width modulators (PWM) with dedicated output pins

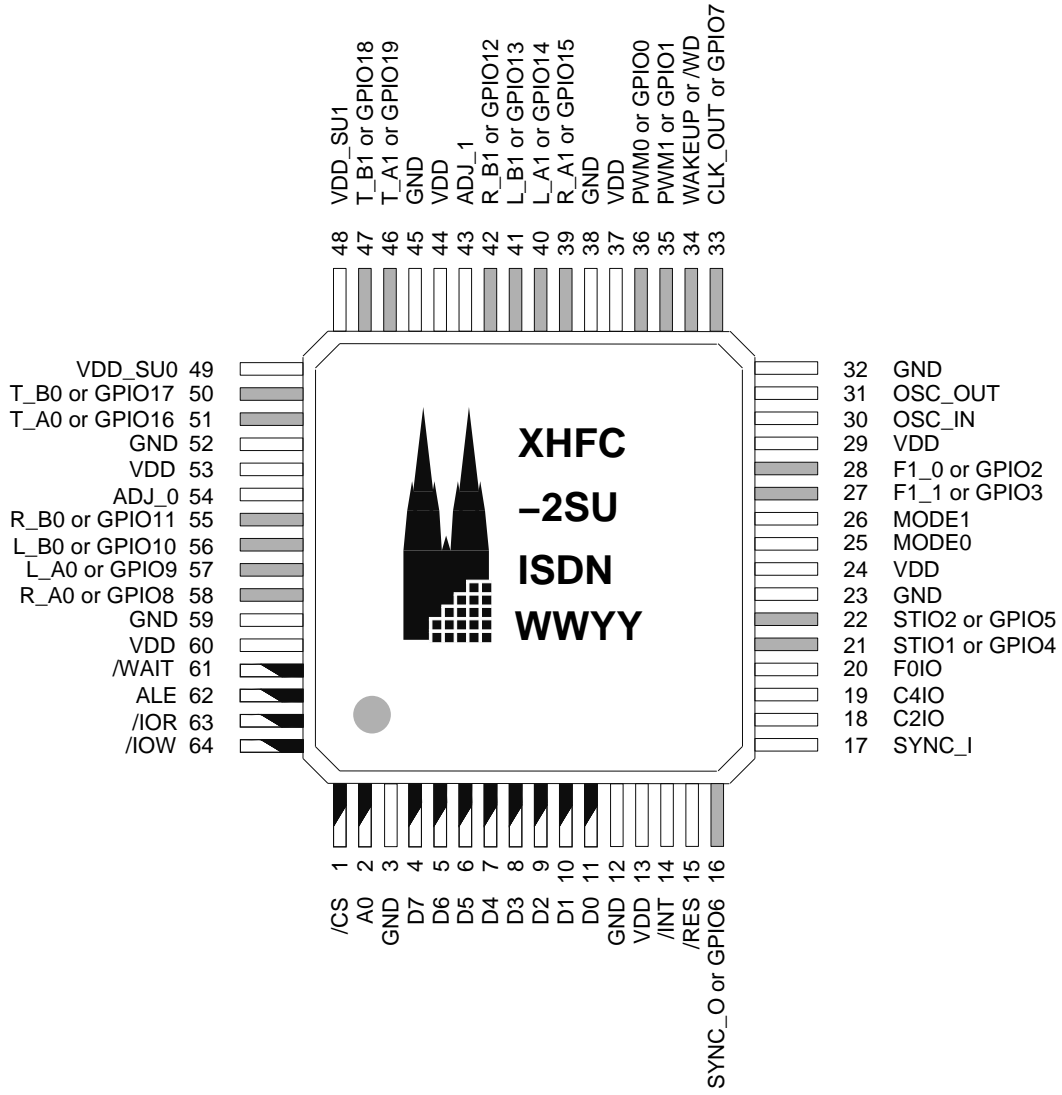
Technology features

- Single 3.3 V power supply, CMOS technology 3.3 V, 5 V tolerant on nearly all inputs
- LQFP 64 package, 0.5 mm pin pitch
- RoHS compliant

²Mitel Serial Telecom bus

1.3 Pin description

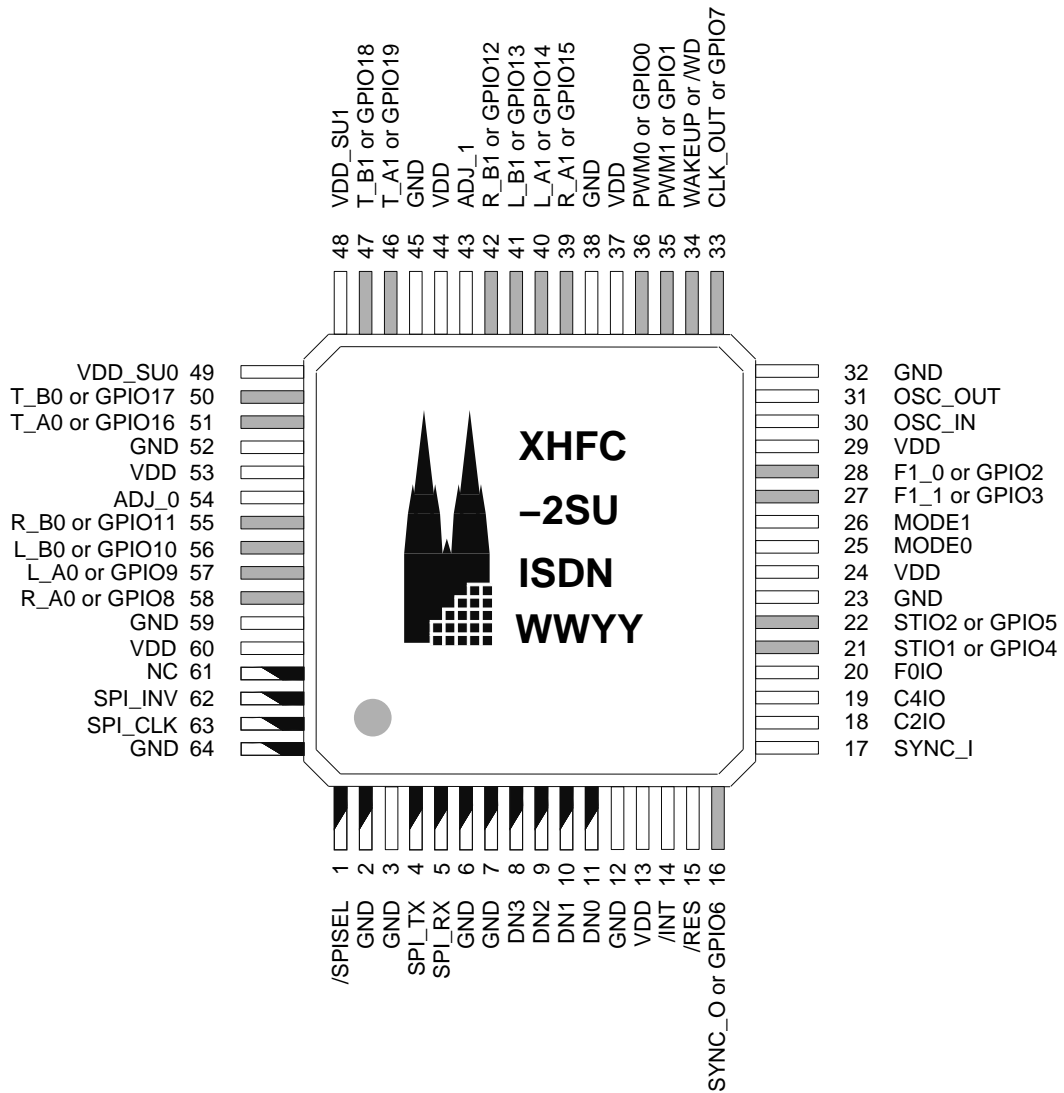
1.3.1 Pinout diagram



- normal function only
- normal and secondary function available
- function depends on the selected processor interface mode

NC pins must not be connected

Figure 1.2: XHFC-2SU pinout in parallel processor interface mode



- normal function only
- normal and secondary function available
- function depends on the selected processor interface mode

NC pins must not be connected

Figure 1.3: XHFC-2SU pinout in serial processor interface mode

1.3.2 Pin list

Pin	Function	Name	I/O (type)	Description	U_{in}/V	I_{out}/mA
Microprocessor bus interface						
1	Processor	/CS	I (2 #)	Chip select, active low	TTL	
	SPI	/SPISEL	I (2 #)	SPI device select, active low	TTL	
2	Processor	A0	I (2 #)	Address	TTL	
	SPI	GND		Ground		
3		GND		Ground		
4	Processor	D7	IO (1 #)	Data bit 7	TTL	8 mA
	SPI	SPI_TX	O (1)	SPI transmit data output		8 mA
5	Processor	D6	IO (1 #)	Data bit 6	TTL	8 mA
	SPI	SPI_RX	I (1 #)	SPI receive data input	TTL	
6	Processor	D5	IO (1 #)	Data bit 5	TTL	8 mA
	SPI	GND		Ground		
7	Processor	D4	IO (1 #)	Data bit 4	TTL	8 mA
	SPI	GND		Ground		
8	Processor	D3	IO (1 #)	Data bit 3	TTL	8 mA
	SPI	DN3	I (1 #)	SPI device number, bit 3	TTL	
9	Processor	D2	IO (1 #)	Data bit 2	TTL	8 mA
	SPI	DN2	I (1 #)	SPI device number, bit 2	TTL	
10	Processor	D1	IO (1 #)	Data bit 1	TTL	8 mA
	SPI	DN1	I (1 #)	SPI device number, bit 1	TTL	
11	Processor	D0	IO (1 #)	Data bit 0	TTL	8 mA
	SPI	DN0	I (1 #)	SPI device number, bit 0	TTL	
Miscellaneous						
12		GND		Ground		
13		VDD		+3.3 V power supply		
14		/INT	Ood #	Interrupt request pin, active low / high programmable		8 mA
15		/RES	Isch #	Reset input pin, active low	TTL	
PCM interface						
16	1st function	SYNC_O	IO (1 #)	Synchronization output	TTL	8 mA
	2nd function	GPIO6	IO (1 #)	General purpose I/O pin 6	TTL	8 mA
17		SYNC_I	I (2 #)	Synchronization input	TTL	

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	U_{in}/V	I_{out}/mA
18		C2IO	IOPu #	PCM bit clock	TTL	8 mA
19		C4IO	IOPu #	PCM double bit clock	TTL	8 mA
20		F0IO	IOPu #	PCM frame clock (8 kHz)	TTL	8 mA
21	1st function	STIO1	IOPu #	PCM data line 1, I or O per time slot	TTL	8 mA
	2nd function	GPIO4	IOPu #	General purpose I/O pin 4	TTL	8 mA
22	1st function	STIO2	IOPu #	PCM data line 2, I or O per time slot	TTL	8 mA
	2nd function	GPIO5	IOPu #	General purpose I/O pin 5	TTL	8 mA
23		GND		Ground		
24		VDD		+3.3 V power supply		
25		MODE0	I (2 #)	Interface mode pin 0	TTL	
26		MODE1	I (2 #)	Interface mode pin 1	TTL	
27	1st function	F1_1	O (1)	Enable signal for external CODEC 1		8 mA
	2nd function	GPIO3	IO (1 #)	General purpose I/O pin 3	TTL	8 mA
28	1st function	F1_0	O (1)	Enable signal for external CODEC 0		8 mA
	2nd function	GPIO2	IO (1 #)	General purpose I/O pin 2	TTL	8 mA
Clock						
29		VDD		+3.3 V power supply		
30		OSC_IN	I (4)	Oscillator input signal	Oscillator	
31		OSC_OUT	O (3)	Oscillator output signal		Oscillator
32		GND		Ground		
33	1st function	CLK_OUT	O (1)	Clock output signal		8 mA
	2nd function	GPIO7	IO (1 #)	General purpose I/O pin 7	TTL	8 mA
Miscellaneous						
34	1st function	WAKEUP	I (1 #)	Wakeup input pin for external awake circuitry	TTL	
	2nd function	/WD	Ood #	Watchdog output signal	TTL	
35	1st function	PWM1	O (1)	Pulse width modulator output 1		8 mA
	2nd function	GPIO1	IO (1 #)	General purpose I/O pin 1	TTL	8 mA
36	1st function	PWM0	O (1)	Pulse width modulator output 0		8 mA
	2nd function	GPIO0	IO (1 #)	General purpose I/O pin 0	TTL	8 mA

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	U_{in}/V	I_{out}/mA
37		VDD		+3.3 V power supply		
38		GND		Ground		
Universal ISDN Ports						
39	1st function	R_A1	I (3)	Combined ST/Up interface no. 1 receive input A	S/T / Up	
	2nd function	GPIO15	IO (3)	General purpose I/O pin 15	TTL	3 mA
40	1st function	L_A1	I (3)	Combined ST/Up interface no. 1 level detect A	S/T / Up	
	2nd function	GPIO14	IO (3)	General purpose I/O pin 14	TTL	3 mA
41	1st function	L_B1	I (3)	Combined ST/Up interface no. 1 level detect B	S/T / Up	
	2nd function	GPIO13	IO (3)	General purpose I/O pin 13	TTL	3 mA
42	1st function	R_B1	I (3)	Combined ST/Up interface no. 1 receive input B	S/T / Up	
	2nd function	GPIO12	IO (3)	General purpose I/O pin 12	TTL	3 mA
43		ADJ_1	Ood #	Combined ST/Up interface no. 1 level generator		2 mA
44		VDD		+3.3 V power supply		
45		GND		Ground		
46	1st function	T_A1	O (2)	Combined ST/Up interface no. 1 transmit output A		S/T / Up
	2nd function	GPIO19	IO (3)	General purpose I/O pin 19	TTL	16 mA
47	1st function	T_B1	O (2)	Combined ST/Up interface no. 1 transmit output B		S/T / Up
	2nd function	GPIO18	IO (3)	General purpose I/O pin 18	TTL	16 mA
48		VDD_SU1		Power supply for ST/Up interface no. 1		
49		VDD_SU0		Power supply for ST/Up interface no. 0		
50	1st function	T_B0	O (2)	Combined ST/Up interface no. 0 transmit output B		S/T / Up
	2nd function	GPIO17	IO (3)	General purpose I/O pin 17	TTL	16 mA
51	1st function	T_A0	O (2)	Combined ST/Up interface no. 0 transmit output A		S/T / Up
	2nd function	GPIO16	IO (3)	General purpose I/O pin 16	TTL	16 mA
52		GND		Ground		
53		VDD		+3.3 V power supply		

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	U_{in} / V	I_{out} / mA
54		ADJ_0	Ood #	Combined ST/Up interface no. 0 level generator		2 mA
55	1st function	R_B0	I (3)	Combined ST/Up interface no. 0 receive input B	S/T / Up	
	2nd function	GPIO11	IO (3)	General purpose I/O pin 11	TTL	3 mA
56	1st function	L_B0	I (3)	Combined ST/Up interface no. 0 level detect B	S/T / Up	
	2nd function	GPIO10	IO (3)	General purpose I/O pin 10	TTL	3 mA
57	1st function	L_A0	I (3)	Combined ST/Up interface no. 0 level detect A	S/T / Up	
	2nd function	GPIO9	IO (3)	General purpose I/O pin 9	TTL	3 mA
58	1st function	R_A0	I (3)	Combined ST/Up interface no. 0 receive input A	S/T / Up	
	2nd function	GPIO8	IO (3)	General purpose I/O pin 8	TTL	3 mA
59		GND		Ground		
60		VDD		+3.3 V power supply		
Microprocessor bus interface						
61	Processor	/WAIT	Ood #	Wait signal for external processor, active low		8 mA
	SPI	NC		Must not be connected		
62	Processor	ALE	I (2 #)	Address latch enable (only in multiplexed modes)	TTL	
	SPI	SPI_INV	I (2 #)	Invert SPI clock	TTL	
63	Processor	/IOR	I (2 #)	Read enable	TTL	
	SPI	SPI_CLK	I (2 #)	SPI clock input	TTL	
64	Processor	/IOW	I (2 #)	Write enable	TTL	
	SPI	GND		Ground		

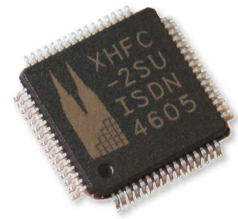
Legend:

IO (1 #)	Bidirectional pin, input 5 V tolerant, 9 pF pin capacitance
IO (3)	Bidirectional pin
IOpu #	Bidirectional pin with internal pull-up resistor of 110 k Ω typical to VDD, input 5 V tolerant, 9 pF pin capacitance
I (1 #)	Input pin, 5 V tolerant, 9 pF pin capacitance
I (2 #)	Input pin, 5 V tolerant, 3 pF pin capacitance
I (3)	Line interface input pin
I (4)	Oscillator input pin, 12 pF pin capacitance
Isch #	Schmitt Trigger input pin, 5 V tolerant, 9 pF pin capacitance
O (1)	Output pin, 9 pF pin capacitance
O (2)	Line interface output pin
O (3)	Oscillator output pin, 10 pF pin capacitance
Ood #	Output pin with open drain, 5 V tolerant, 9 pF pin capacitance
NC	Must not be connected

Pins with 5 V tolerant input are marked with #.

Unused input pins should be connected to ground.

Unused I/O pins should be connected with pull-down resistor to ground. It is recommended to use a maximum of 22 k Ω for pins with internal pull-up resistor and a maximum of 100 k Ω for pins without internal pull-up resistor.



Chapter 2

Microprocessor bus interface

Table 2.1: Overview of the XHFC-2SU bus interface registers

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x01	R_CTRL	70	0x15	R_RAM_USE	72
0x08	R_RAM_ADDR	71	0x16	R_CHIP_ID	72
0x09	R_RAM_CTRL	71	0x1F	R_CHIP_RV	72
			0x88	R_INT_DATA	73
			Read / write registers:		
			Address	Name	Page
			0xC0	R_RAM_DATA	74

2.1 Mode selection

XHFC-2SU has an integrated microprocessor bus interface which can be configured as parallel 8 bit microprocessor interface and serial processor interface (SPI). Table 2.2 shows how to select these bus modes via the two pins MODE0 and MODE1.

Table 2.2: *Microprocessor access types*

Bus mode	MODE1	MODE0
Serial processor interface (SPI)	0	0
Parallel processor interface		
Modes 2 and 2m: Motorola	0	1
Modes 3 and 3m: Intel	1	0
Auto-EEPROM mode	1	1

The Auto-EEPROM mode lets XHFC-2SU operate without an external microprocessor. The complete setup procedure can be stored in an external EEPROM. This mode is useful for applications that do not need a microprocessor intervention during operation (i.e. only static initialization required), e.g. U_p repeater applications.

The mode selection pins MODE0 and MODE1 must be stable during hardware reset. Sections 2.2 to 2.4 explain how to use XHFC-2SU in the different bus modes.

2.2 Parallel processor interface

Table 2.3: Overview of the parallel processor interface pins

Number	Name	Description
14	/INT	Interrupt request pin, active low / high programmable
15	/RES	Reset input pin, active low
25	MODE0	Interface mode pin 0
26	MODE1	Interface mode pin 1
61	/WAIT	Wait signal for external processor, active low
62	ALE	Address latch enable (only in multiplexed modes)
63	/IOR	Read enable
64	/IOW	Write enable
1	/CS	Chip select, active low
2	A0	Address
11..4	D0..D7	

2.2.1 Overview

XHFC-2SU has four different parallel microprocessor interface modes. According to the name conventions of other Cologne Chip products (HFC series) the non-multiplexed processor interface modes are numbered 2 and 3 like shown in Table 2.4. The corresponding multiplexed modes are named 2m and 3m.¹

The interface mode is determined with power-on. For the non-multiplexed modes 2 and 3, the ALE pin must be stable after reset and should be fixed to ground.



Multiplexed modes are selected after reset with the first rising edge of ALE. XHFC-2SU then switches permanently from mode 2 into mode 2m or from mode 3 into mode 3m respectively. XHFC-2SU cannot switch to multiplexed modes before end of reset time. Rising and falling edges of ALE are ignored during reset time.

2.2.2 Interface signals

The processor interface signals have different names for Motorola and Intel microprocessors. Table 2.5 shows the mapping with the pin names of XHFC-2SU.

¹Mode 3m is formerly known as mode 4 from previous chips of the HFC series.

Table 2.4: Parallel processor interface mode selection

Pin	Mode 2	Mode 3	Mode 2m	Mode 3m
	Motorola	Intel	Motorola	Intel
	Non-multiplexed		Multiplexed	
MODE0	1	0	1	0
MODE1	0	1	0	1
ALE	0 *1	0 *1	 *2	 *2

*1: This pin should be fixed to ground

*2: 1-pulse latches register address

Table 2.5: Pins and signal names of the parallel processor interface modes

XHFC-2SU pins		Signal names			
Number	Name	Mode 2	Mode 3	Mode 2m	Mode 3m
		Motorola	Intel	Motorola	Intel
		Non-multiplexed	Non-multiplexed	Multiplexed	Multiplexed
1	/CS	/CS	/CS	/CS	/CS
63	/IOR	/DS	/RD	/DS	/RD
64	/IOW	R/W	/WR	R/W	/WR
62	ALE	'0'	'0'	ALE	ALE
2	A0	A0	A0	'0'	'0'
4..11	D7..D0	D7..D0	D7..D0	AD7 .. AD0	AD7 .. AD0

2.2.3 Register access

2.2.3.1 Non-multiplexed / multiplexed access

Non-multiplexed modes: With this indirect addressing method, modes 2 and 3, A0 is the address input line. A0 = '0' is used for data write and read, while A0 = '1' is used for address write and read-back accesses.

Both, register address and data, are transferred through the pins D7..D0 (D0 is LSB). The address must first be written on D7..D0 with A0 = '1'. Then data read or write can be performed with A0 = '0' on the same bus D7..D0. Several data accesses can be executed to the same register address without writing the address again. Access details are shown in the timing diagrams in Figures 2.1 and 2.2.

Multiplexed modes: Direct addressing is supported with the multiplexed modes 2m and 3m. These do not use A0 and require A0 = '0' all the time.

All registers can directly be accessed in multiplexed mode. ALE latches the register address. The multiplexed address and data bus is D7..D0 (D0 is LSB). Timing diagrams are shown in Figures 2.3 and 2.4.

2.2.3.2 Read* access

Some registers must be read with an indirect method, the so-called Read* access (written as 'r*' characteristic in register tables). This refers to all readable registers in the address range 0xC0 . . 0xFF, here called *target register*.

The Read* access performs two consecutive read accesses to XHFC-2SU:

1. First, a read access to the target register must be executed. The returned value is not the value of the register and must be ignored.
2. Then, the target register value can be read from register R_INT_DATA.

The Read* access is practical for the target registers R_RAM_DATA, A_SL_CFG, A_CH_MSK, A_CON_HDLC, A_SUBCH_CFG, A_CHANNEL, A_FIFO_SEQ and A_FIFO_CTRL. All other registers have a direct read access.

2.2.3.3 Register address read-back capability

When the non-multiplexed modes 2 and 3 are used, the address read access can be executed to read-back the address of the currently selected register.

2.2.3.4 Problems with interrupts between address write and data read / write accesses

The register address read-back capability is useful for interrupt procedures, e.g., to save and restore the previous state:

```
interrupt procedure: - execute address read access and store the register address
                   - ... (execute the interrupt service routine)
                   - address write access to restore the previous register address
```

This procedure is important to avoid data read or write to an unexpected register address after a register read or write access has been split by an interrupt service routine which executes any access to XHFC-2SU.

2.2.4 Signal and timing characteristics

Table 2.6 shows the interface signals for the different microprocessor interface modes. Timing characteristics are shown in Figures 2.1 and 2.2 for non-multiplexed modes 2 and 3. Figures 2.3 and 2.4 show multiplexed modes 2m and 3m timing characteristics. Please see Table 2.7 for a quick timing and symbol list finding.

2.2.4.1 Bus interface in mode 2 and mode 3 (non-multiplexed)

Read access

8 bit processors read data like shown in Figure 2.1. Timing values are listed in Table 2.8.

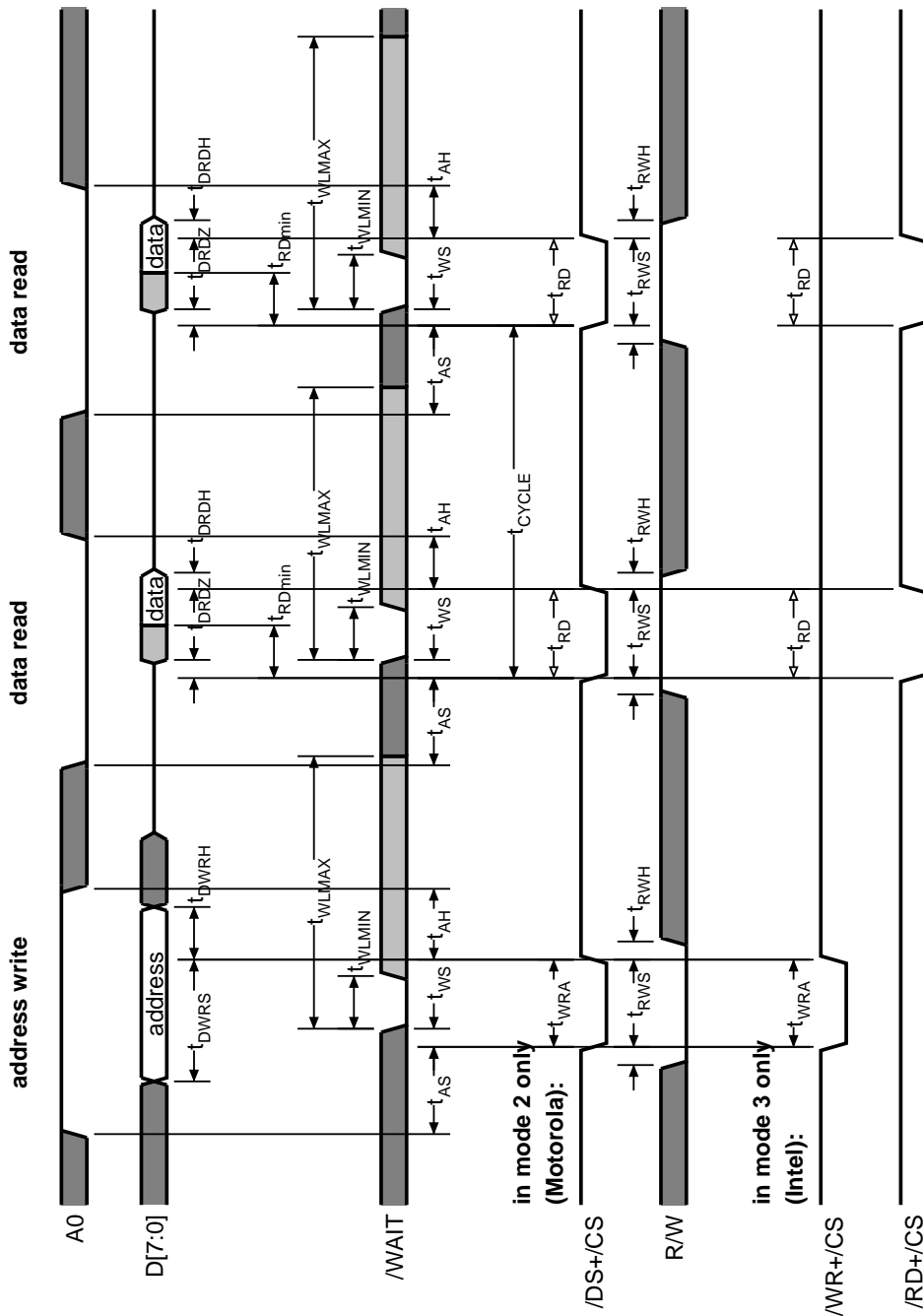






Figure 2.1: Bus interface read access in mode 2 (Motorola) and mode 3 (Intel)

Notes:

- (1) It is not necessary to use $/WAIT$ if the processor is able to ensure the t_{CYCLE} timing constraint.
- (2) $/WAIT$ signal is only active if the interval of two consecutive data phases is less than t_{CYCLE} .
- (3) For faster access, it is recommended to set up a processor timing which does not require the $/WAIT$ signal.

Table 2.6: Overview of accesses in parallel microprocessor interface mode (*X* = don't care)

/CS	/IOR (/DS, /RD)	/IOW (R/W, /WR)	ALE	A0	Operation	Processor interface mode
1	X	X	X	X	no access	all
0	1	1	0	0	no access	all
0	0	1	0	1	read address	mode 2
0	0	0	0	1	write address	mode 2
0	0	1	0	0	read data	mode 2
0	0	0	0	0	write data	mode 2
0	0	1	0	1	read address	mode 3
0	1	0	0	1	write address	mode 3
0	0	1	0	0	read data	mode 3
0	1	0	0	0	write data	mode 3
0	0	1	 *	0	read data	mode 2m
0	0	0	 *	0	write data	mode 2m
0	0	1	 *	0	read data	mode 3m
0	1	0	 *	0	write data	mode 3m

*: 1-pulse latches register address

Table 2.7: Timing diagrams of the parallel microprocessor interface

Mode	Access type	Timing		Timing values	
		Figure	on page	table	on page
2 & 3	read	2.1	46	2.8	48
2 & 3	write	2.2	49	2.9	50
2m & 3m	read	2.3	52	2.10	51
2m & 3m	write	2.4	54	2.11	53

Data can be read with²

$$(/DS + /CS) = '0' \quad \text{and} \quad R/W = '1' .$$

in mode 2 (Motorola) or with

$$(/RD + /CS) = '0' \quad \text{and} \quad /WR = '1'$$

in mode 3 (Intel). The data bus is stable after $t_{RD \min}$ and returns into tristate after t_{DRDH} .

The address line A0 requires a setup time t_{AS} . The hold time of this line is t_{AH} .

²/DS + /CS means logical OR function of the two signals.

Table 2.8: Symbols of read accesses in Figure 2.1

Symbol	min / ns	max / ns	Characteristic
t_{AS}	20		A0 valid to /DS+/CS (/WR+/CS) \downarrow setup time
t_{AH}	0		Address hold time after /DS+/CS (/WR+/CS) \uparrow
t_{WRA}	20		Write time for address write
t_{DWRS}	25		Write data setup time to /DS+/CS (/WR+/CS) \uparrow
t_{DWRH}	0		Write data hold time from /DS+/CS (/WR+/CS) \uparrow
t_{RD}	25		Read time
t_{CYCLE}			/DS+/CS (/RD+/CS) \downarrow to next /DS+/CS (/RD+/CS) \downarrow
	30		Register address range 0x00 . . 0x7F
	$3.5 \cdot t_{SYS}$		Register address range 0x80 . . 0xFF
t_{DRDZ}	3		/DS+/CS (/RD+/CS) \downarrow to data buffer turn on time
t_{DRDH}	2	15	/DS+/CS (/RD+/CS) \uparrow to data buffer turn off time
t_{RWS}	2		R/W setup time to /DS+/CS \downarrow (in mode 2 only)
t_{RWH}	2		R/W hold time after /DS+/CS \uparrow (in mode 2 only)
t_{WS}		10	/WAIT turn on time to /DS+/CS or /WR+/CS \downarrow
t_{WLMIN}	0		Minimum /WAIT low time
t_{WLMAX}		t_{CYCLE}	Maximum /WAIT low time

The cycle time specifies the time between two consecutive data accesses. $t_{SYS} = 1/f_{SYS}$ with the system clock f_{SYS} .

Write access

8 bit processors write data like shown in Figure 2.2. Timing values are listed in Table 2.9.

Data is written with \uparrow of (/DS + /CS) in mode 2 (Motorola) or with \uparrow of (/WR + /CS) in mode 3 (Intel) respectively. XHFC-2SU requires a data setup time t_{DWRS} and a data hold time t_{DWRH} .

The address line A0 requires a setup time t_{AS} which starts when the address signal is valid. The hold time is $t_{AH} \geq 0$ ns after address write access and it is $t_{AHD} \geq 0.5t_{SYS} + 9$ ns after data write access.³

³Please ask Cologne Chip's support team if the intended processor has problems with t_{AHD} (support@CologneChip.com).

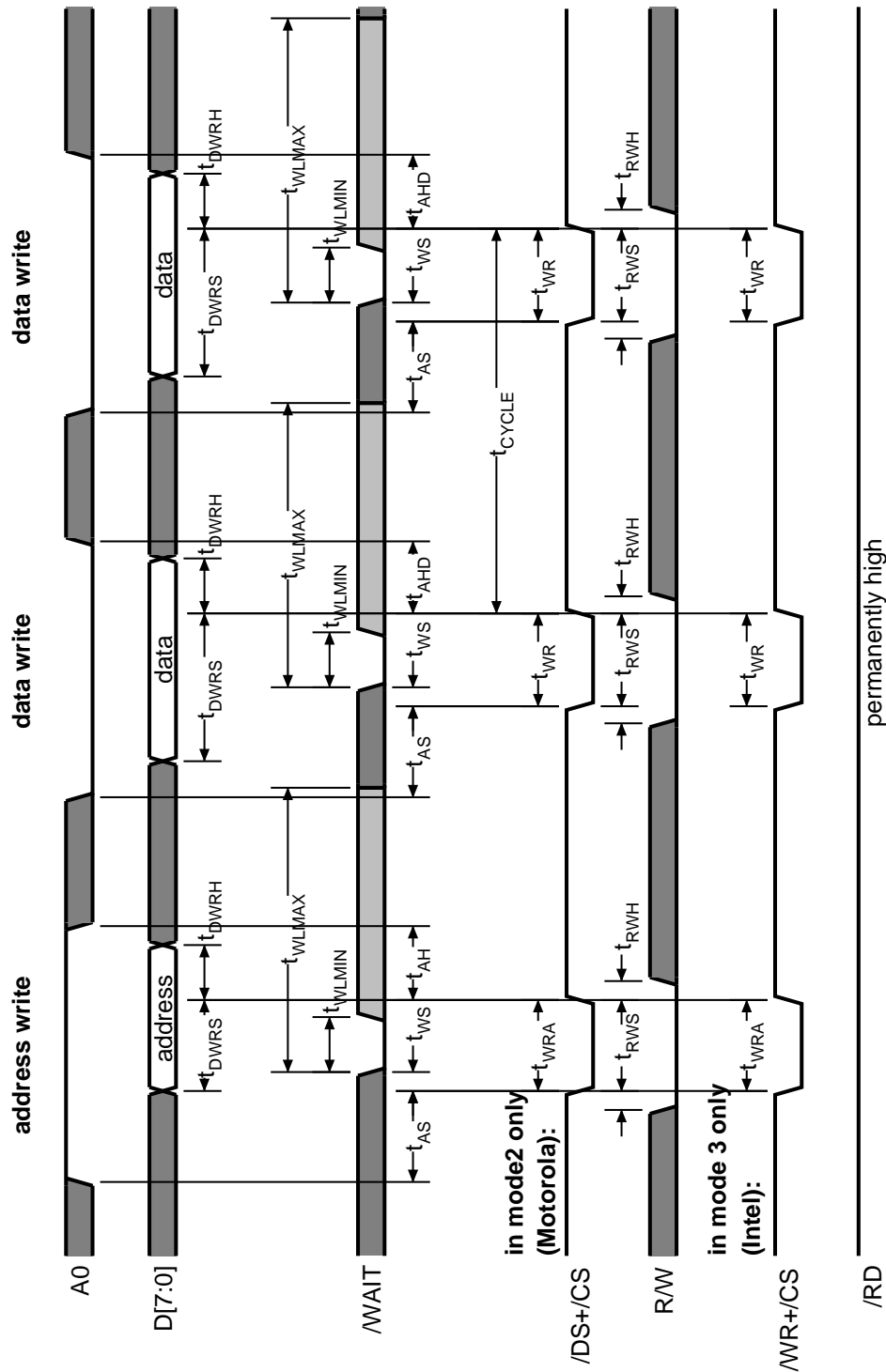


Figure 2.2: Bus interface write access in mode 2 (Motorola) and mode 3 (Intel)

Notes:

- (1) It is not necessary to use /WAIT if the processor is able to ensure the t_{CYCLE} timing constraint.
- (2) /WAIT signal is only active if the interval of two consecutive data phases is less than t_{CYCLE} .
- (3) For faster access, it is recommended to set up a processor timing which does not require the /WAIT signal.

Table 2.9: Symbols of write accesses in Figure 2.2

Symbol	min / ns	max / ns	Characteristic
t_{AS}	20		A0 valid to /DS+/CS (/WR+/CS) \downarrow setup time
t_{AH}	0		Address hold time of address write access after /DS+/CS (/WR+/CS) \downarrow
t_{AHD}	$0.5 \cdot t_{SYS} + 9$		Address hold time of data write access after /DS+/CS (/WR+/CS) \downarrow
t_{WRA}	20		Write time for address write
t_{DWRS}	20		Write data setup time to /DS+/CS (/WR+/CS) \downarrow
t_{DWRH}	0		Write data hold time from /DS+/CS (/WR+/CS) \downarrow
t_{WR}	20		Write time
t_{CYCLE}			/DS+/CS (/RD+/CS) \downarrow to next /DS+/CS (/RD+/CS) \downarrow
	$1.5 \cdot t_{SYS}$		Register address range 0x00 . . 0x7F
	$3.5 \cdot t_{SYS}$		Register address range 0x80 . . 0xFF
t_{RWS}	2		R/W setup time to /DS+/CS \downarrow (in mode 2 only)
t_{RWH}	2		R/W hold time after /DS+/CS \downarrow (in mode 2 only)
t_{WS}		10	/WAIT turn on time to /DS+/CS or /WR+/CS \downarrow
t_{WLMIN}	0		Minimum /WAIT low time
t_{WLMAX}		t_{CYCLE}	Maximum /WAIT low time

Table 2.10: Symbols of read accesses in Figure 2.3

Symbol	min / ns	max / ns	Characteristic
t_{AS}	20		Address valid to ALE \downarrow setup time
t_{AH}	0		Address hold time after ALE \downarrow
t_{ALE}	10		Address latch time
t_{ALEL}	0		ALE \downarrow to /RD+/CS \downarrow
t_{ALEH}	0		/RD+/CS \uparrow to ALE \uparrow
t_{RD}	25		Read time
t_{CYCLE}			/DS+/CS (/RD+/CS) \downarrow to next /DS+/CS (/RD+/CS) \downarrow
	30		Register address range 0x00 . . 0x7F
	$3.5 \cdot t_{SYS}$		Register address range 0x80 . . 0xFF
t_{DRDZ}	3		/RD+/CS \downarrow to data buffer turn on time
t_{DRDH}	2	15	/RD+/CS \uparrow to data buffer turn off time
t_{RWS}	2		R/W setup time to /DS+/CS \downarrow (in mode 2 only)
t_{RWH}	2		R/W hold time after /DS+/CS \uparrow (in mode 2 only)
t_{WS}		10	/WAIT turn on time to /DS+/CS or /WR+/CS \downarrow
t_{WLMIN}	0		Minimum /WAIT low time
t_{WLMAX}		t_{CYCLE}	Maximum /WAIT low time

2.2.4.2 Bus interface in mode 2m and mode 3m (multiplexed)

Read access

8 bit processors read data like shown in Figure 2.3. Timing values are listed in Table 2.10.

Data can be read with⁴

$$(/DS + /CS) = '0' \quad \text{and} \quad R/W = '1' .$$

in mode 2m (Motorola) or with

$$/WR = '1'$$

in mode 3m (Intel). The data bus is stable after $t_{RD \min}$ and returns into tristate after t_{DRDH} .

The address line **A0** requires a setup time t_{AS} in relation to \downarrow of ALE. The hold time of these lines is t_{AH} . If consecutive read accesses are on the same register address, multiple address write accesses are not required.

The cycle time specifies the time between two consecutive data accesses. $t_{SYS} = 1/f_{SYS}$ with the system clock f_{SYS} .

⁴/DS + /CS means logical OR function of the two signals.

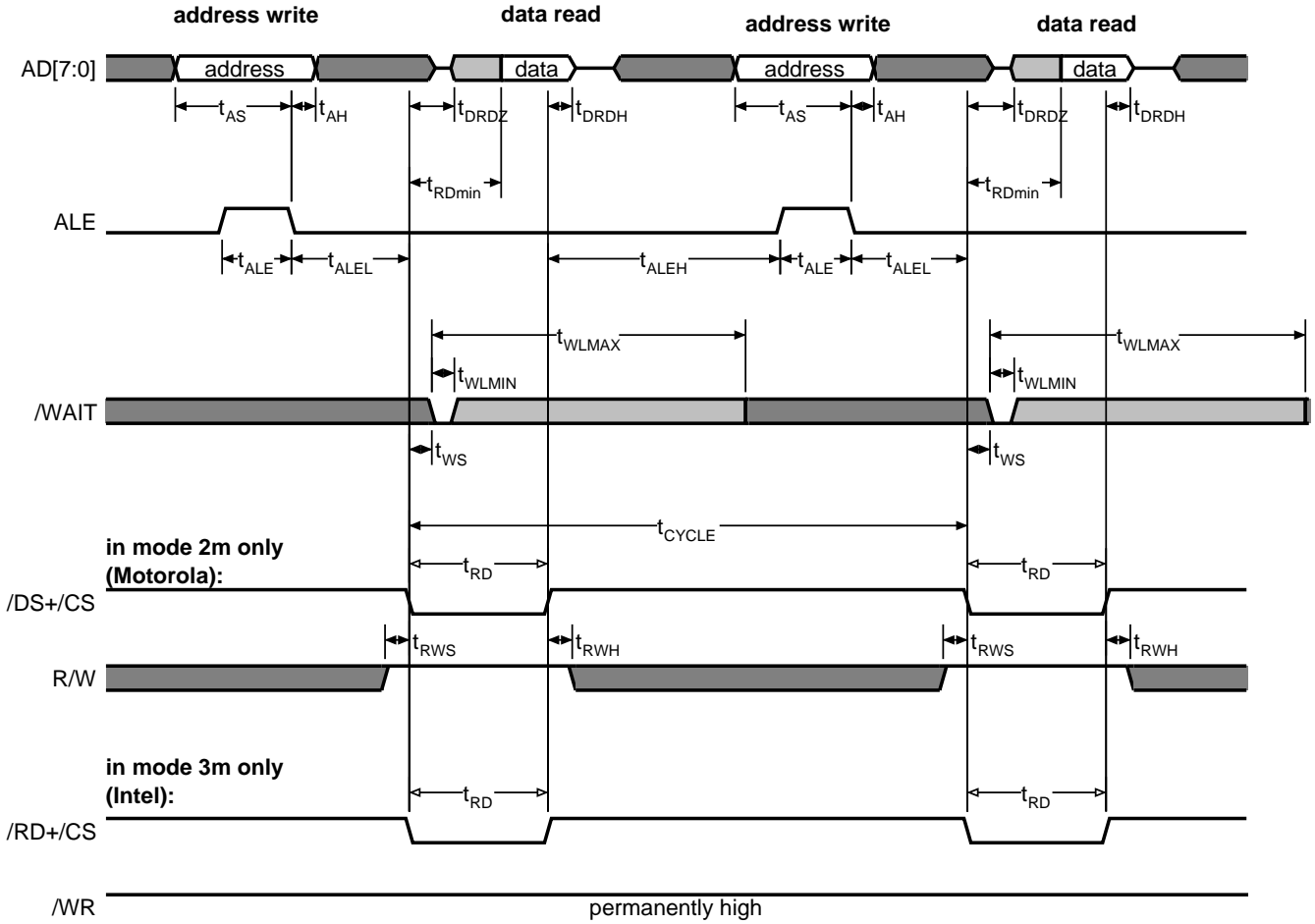


Figure 2.3: Bus interface read access in modes 2m and 3m (multiplexed)

Table 2.11: Symbols of write accesses in Figures 2.4

Symbol	min / ns	max / ns	Characteristic
t_{AS}	20		Address valid to ALE \downarrow setup time
t_{AH}	0		Address hold time after $\overline{WR+}/CS \uparrow$
t_{ALE}	10		Address latch time
t_{ALEL}	0		ALE \downarrow to $\overline{WR+}/CS \downarrow$
t_{ALEH}	0		$\overline{WR+}/CS \uparrow$ to ALE \uparrow
t_{DWRS}	20		Write data setup time to $\overline{WR+}/CS \uparrow$
t_{DWRH}	0		Write data hold time from $\overline{WR+}/CS \downarrow$
t_{WR}	20		Write time
t_{CYCLE}			$\overline{DS+}/CS (\overline{WR+}/CS) \uparrow$ to next $\overline{DS+}/CS (\overline{WR+}/CS) \uparrow$
	$1.5 \cdot t_{SYS}$		Register address range 0x00 . . 0x7F
	$3.5 \cdot t_{SYS}$		Register address range 0x80 . . 0xFF
t_{RWS}	2		R/W setup time to $\overline{DS+}/CS \downarrow$ (in mode 2 only)
t_{RWH}	2		R/W hold time after $\overline{DS+}/CS \uparrow$ (in mode 2 only)
t_{WS}		10	\overline{WAIT} turn on time to $\overline{DS+}/CS$ or $\overline{WR+}/CS \downarrow$
t_{WLMIN}	0		Minimum \overline{WAIT} low time
t_{WLMAX}		t_{CYCLE}	Maximum \overline{WAIT} low time

Write access

8 bit processors write data like shown in Figure 2.4. Timing values are listed in Table 2.11.

Data is written with \uparrow of $(\overline{DS} + \overline{CS})$ in mode 2m (Motorola) or with \uparrow of $(\overline{WR} + \overline{CS})$ in mode 3m (Intel) respectively. XHFC-2SU requires a data setup time t_{DWRS} and a data hold time t_{DWRH} .

The address line A0 requires a setup time t_{AS} in relation to \downarrow of ALE. The hold time of these lines is t_{AH} . If consecutive write accesses are on the same register address, multiple address write accesses are not required.

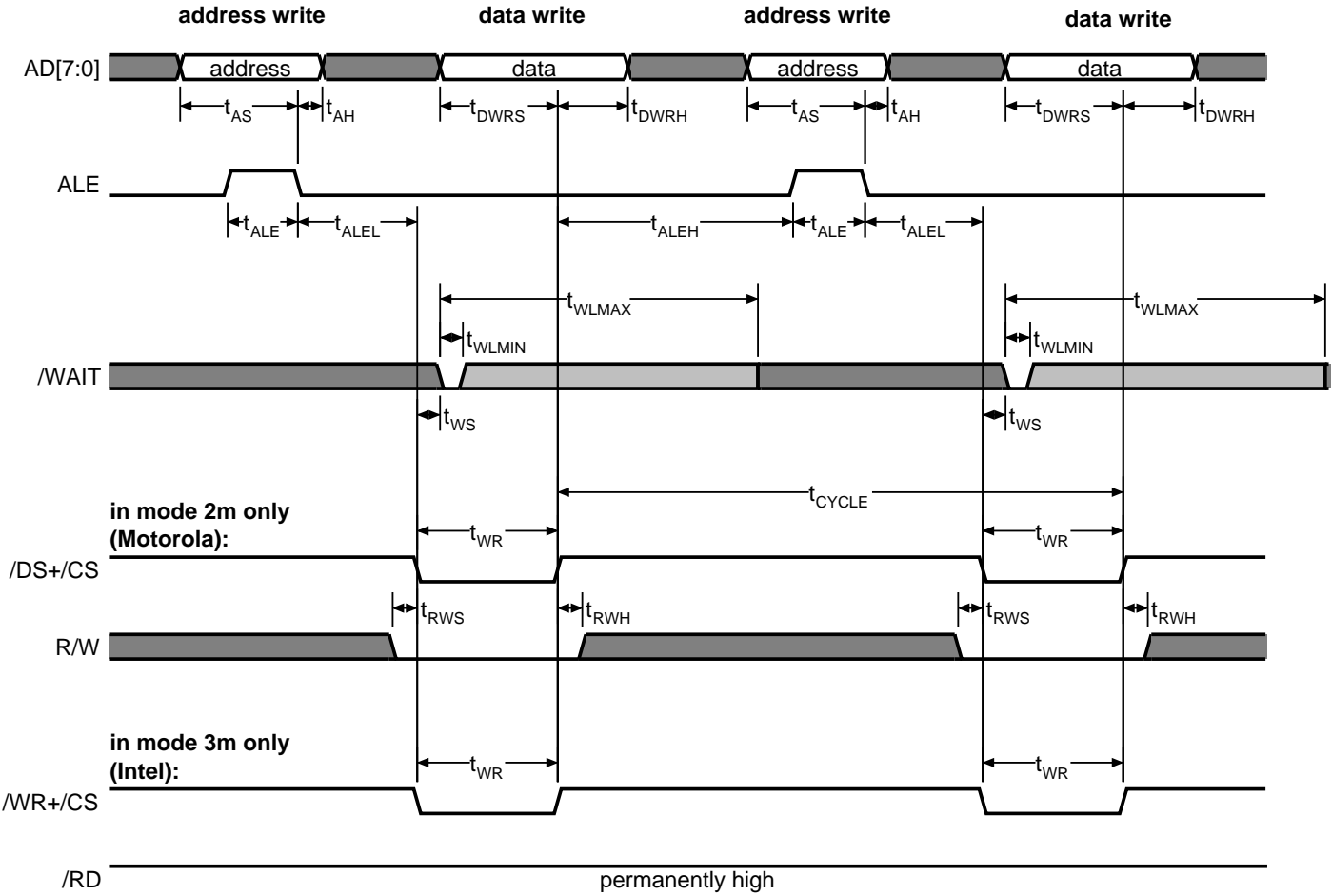


Figure 2.4: Bus interface write access in mode 2m and 3m (multiplexed)

2.2.5 Microprocessor connection circuitries

Figures 2.5 to 2.8 show examples how to connect XHFC-2SU to different parallel processor interfaces.

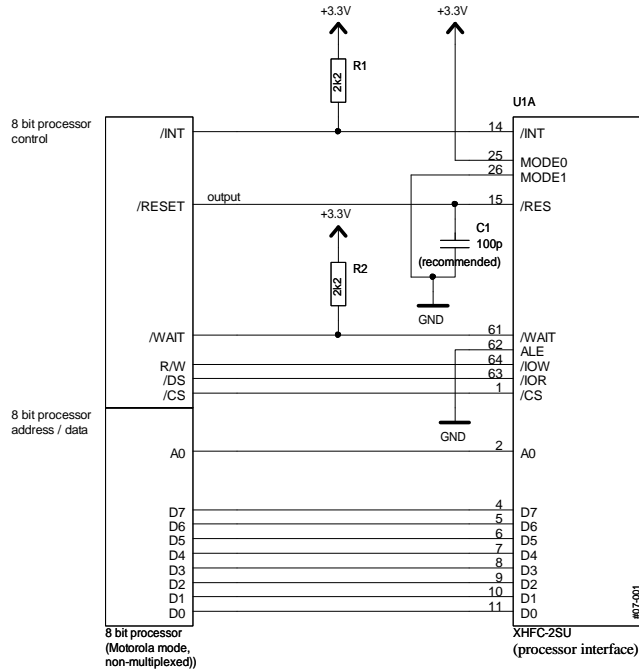


Figure 2.5: 8 bit Motorola processor circuitry example (mode 2)

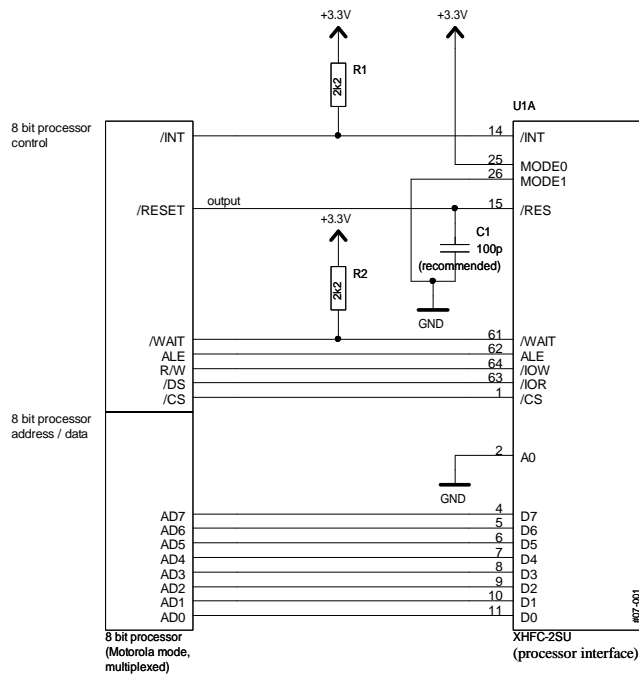


Figure 2.6: 8 bit Motorola processor circuitry example (mode 2m)

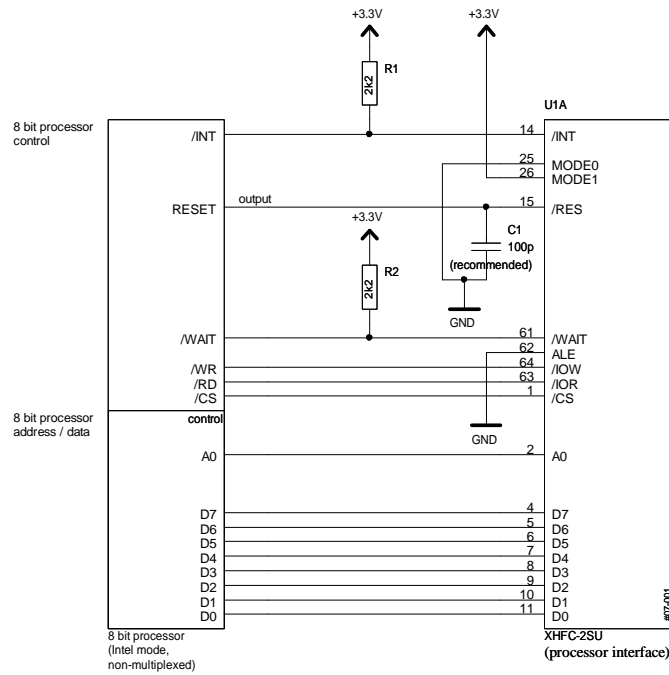


Figure 2.7: 8 bit Intel processor circuitry example (mode 3)

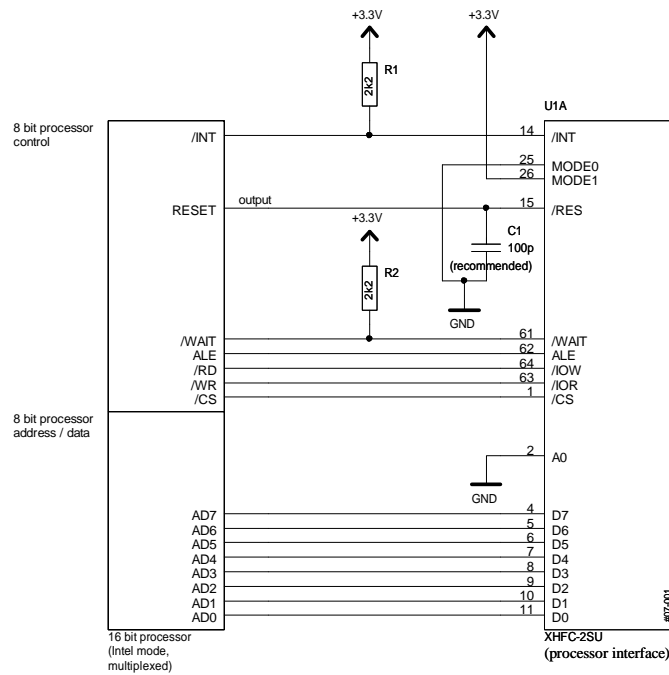


Figure 2.8: 8 bit Intel processor circuitry example (mode 3m)

2.3 Serial processor interface (SPI)

Table 2.12: Overview of the SPI interface pins

Number	Name	Description
14	/INT	Interrupt request pin, active low / high programmable
15	/RES	Reset input pin, active low
25	MODE0	Interface mode pin 0
26	MODE1	Interface mode pin 1
4	SPI_TX	SPI transmit data output
5	SPI_RX	SPI receive data input
63	SPI_CLK	SPI clock input
1	/SPISEL	SPI device select, active low
62	SPI_INV	Invert SPI clock
11	DN0	SPI device number, bit 0
10	DN1	SPI device number, bit 1
9	DN2	SPI device number, bit 2
8	DN3	SPI device number, bit 3

XHFC-2SU has a serial processor interface (SPI) which is compatible with Motorola's SPI. CPUs and MCUs with SPI interface are also available from a variety of semiconductor vendors. The SPI interface has four signal pins as shown in the upper part of Table 2.12. Additional pins are used for SPI clock inversion and SPI device number selection.

SPI interface mode is selected by $\text{MODE0} = '0'$ and $\text{MODE1} = '0'$ (pins 25 and 26). XHFC-2SU supports only SPI slave mode.

Any register access consists of two transactions – an address write transaction first and a data read or write transaction afterwards. SPI transactions of XHFC-2SU have either a length of 16 bits for single byte accesses or 40 bits for high performance accesses. The first byte is a control byte in both cases, whereas the following bits are either one data byte or four data bytes. Control and data bytes are transmitted with MSB first.

2.3.1 SPI control byte

Tables 2.13 and 2.14 show an overview of the control byte construction. R and A are used to specify read/write and address/data transaction types. The meaning of bit position 5 depends on the value of A; broadcasting can be enabled with an address transaction and single or multiple data bytes is selected with a data transaction.

Up to 16 microchips of the XHFC series can be connected to the SPI interface and can operate with the same /SPISEL signal. The desired microchip is selected with the device address C3 .. C0 within an address transaction. Every XHFC microchip must specify its address with DN3 .. DN0 pins (device number) connected to ground or power supply. A microchip is selected with $\text{C3}.. \text{C0} = \text{DN3}.. \text{DN0}$ where all numbers in the range 0 .. 15 are allowed.

In addition to the chip selection, an SPI write access writes its data into *all* connected XHFC microchips if broadcast is used. SPI read accesses with enabled broadcast execute the register read access in all connected XHFC microchips, but only the specified chip delivers its data to the SPI master. The broadcast write access is useful for initialization procedures, e.g., for those registers, which must be initialized in all connected XHFC microchips.

Table 2.15 summarizes the SPI control commands which are coded in the control byte.

2.3.2 SPI transactions

The waveforms of an address or data write transaction with 16 bits length are shown in Figure 2.9. XHFC-2SU receives the control byte and the data byte with R = '0' on the SPI_RX line. The SPI_TX

Table 2.13: SPI control byte with A = '0'

Bit	Name	Description
7	R	'0' = write '1' = read
6	A	'0' = register data
5	M	'0' = single data byte '1' = multi data bytes (4 bytes)
4	'0'	This bit must be zero in all SPI transactions
3..0	'0000'	Must be zero in data transactions

Table 2.14: SPI control byte with A = '1'

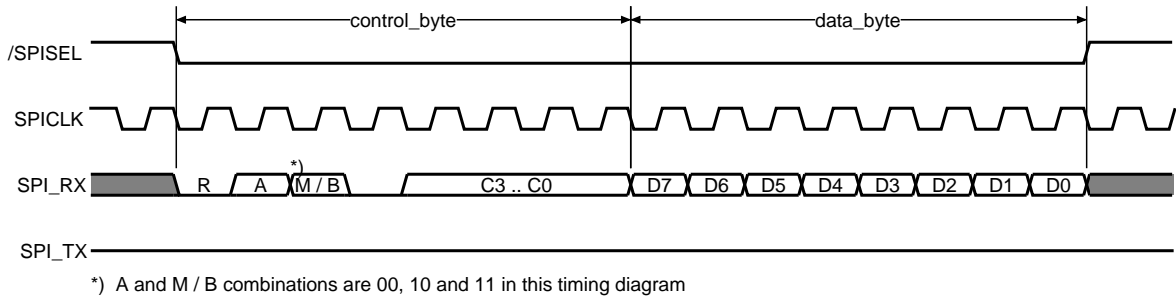
Bit	Name	Description
7	R	'0' = write '1' = read
6	A	'1' = register address
5	B	'0' = no broadcast '1' = broadcast
4	'0'	This bit must be zero in all SPI transactions
3..0	C3 .. C0	Device address (used for chip select generation)

Table 2.15: SPI control commands for register read and write operations

R	Control byte				R	A	M/B	Timing diagram
	A	M/B	0	C3 .. C0				
0	0	0	0	0000	write	data	single data byte	Fig. 2.9
1	0	0	0	0000	read	data	single data byte	Fig. 2.10
0	1	0	0	CCCC	write	address	no broadcast	Fig. 2.9
1	1	0	0	CCCC	read	address	no broadcast	Fig. 2.10
0	0	1	0	0000	write	data	multiple data bytes	Fig. 2.11
1	0	1	0	0000	read	data	multiple data bytes	Fig. 2.12, 2.13
0	1	1	0	CCCC	write	address	broadcast	Fig. 2.9
1	1	1	0	CCCC	read	address	broadcast	Fig. 2.10
X	X	X	1	XXXX	not allowed			–

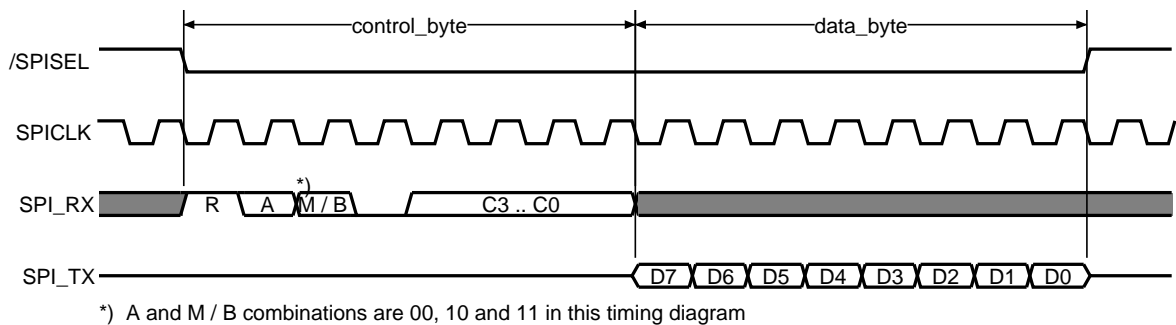
pin is not used for write transactions and has tri-state level all the time.

A read transaction is shown in Figure 2.10. XHFC-2SU receives the control byte on the SPI_RX line and transmits the requested data byte on the SPI_TX line afterwards.



*) A and M / B combinations are 00, 10 and 11 in this timing diagram

Figure 2.9: 16 bit SPI write transaction ($R = '0'$, one data byte)



*) A and M / B combinations are 00, 10 and 11 in this timing diagram

Figure 2.10: 16 bit SPI read transaction ($R = '1'$, one data byte)

$A = '0'$ and $M = '1'$ specify a 40 bit transaction. This is shown in Figures 2.11 and 2.12 for SPI write and read transactions.

Any SPI transactions can be split by the SPI master into the control byte and the data bytes with $/\text{SPISEL} = '1'$. In this case the transmission pauses and will be continued after $/\text{SPISEL}$ returns to low level. An example for a split read transaction with multiple data bytes is shown in Figure 2.13. Write transactions can be split in the same way.

The SPI host is not allowed to break an SPI transaction by an interrupt service routine which executes any access to XHFC-2SU. Otherwise master and slave could have different views whether a specific byte is a control or a data byte. If the SPI master cannot ensure this characteristic, interrupts must be disabled between control byte and data byte.

2.3.3 Transaction duration

16 bit SPI transaction have a minimal length of 16 clock cycles. With $f_{\text{SPICLK}} = 4\text{MHz}$, e.g., a 16 bit transaction takes $4\mu\text{s}$. XHFC-2SU can operate with a maximum SPI clock of 25 MHz which leads to a minimal 16 bit transaction time of

$$T_{\text{trans16,min}} = \frac{16}{25\text{MHz}} = 640\text{ns} .$$

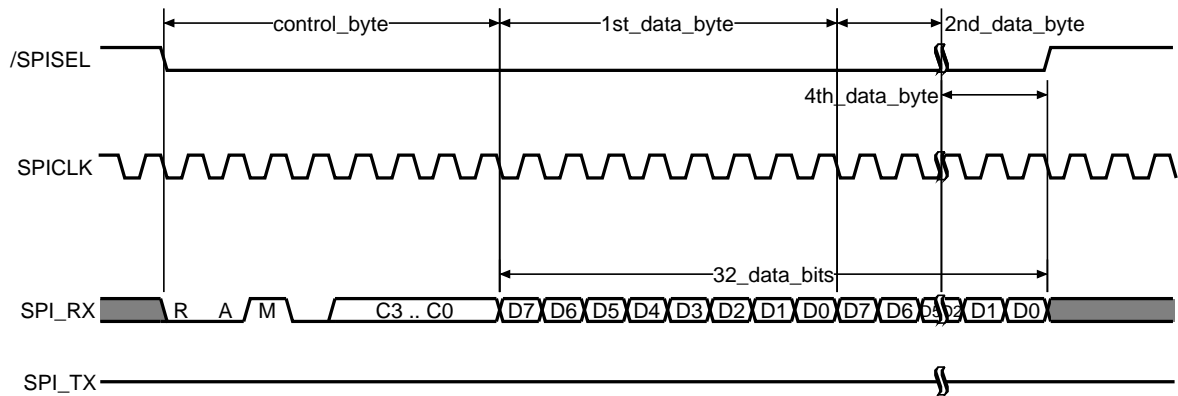


Figure 2.11: 40 bit SPI write transaction ($R = '0'$, four data bytes)

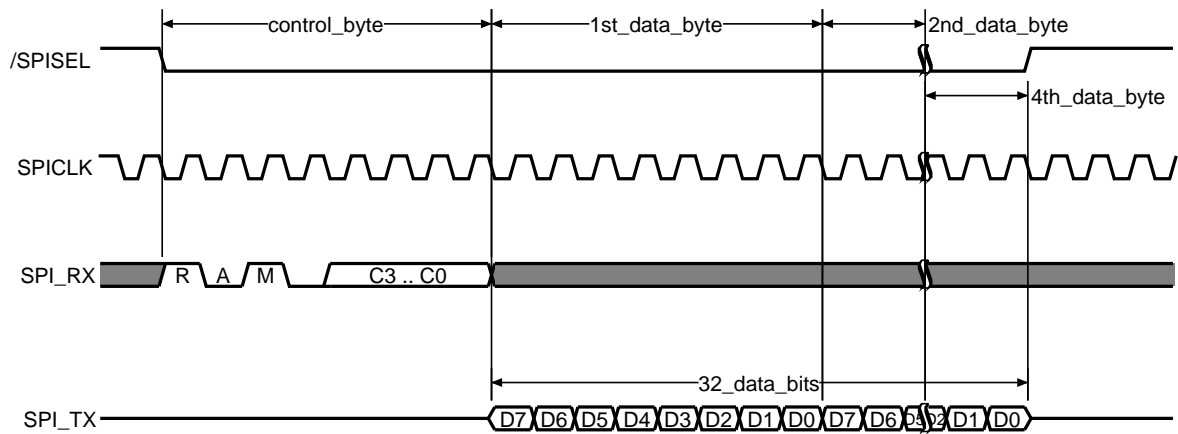


Figure 2.12: 40 bit SPI read transaction ($R = '1'$, four data bytes)

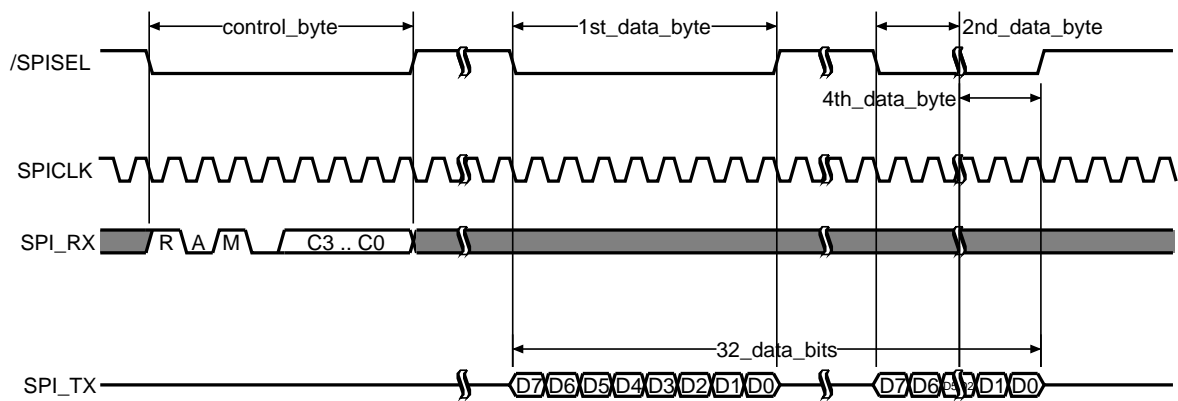


Figure 2.13: Split 40 bit SPI read transaction ($R = '1'$)

40 bit SPI transaction have a minimal length of 40 clock cycles. With $f_{\text{SPICLK}} = 4\text{MHz}$, e.g., a 40 bit transaction takes $10\ \mu\text{s}$. With the maximum SPI clock of 25 MHz, the minimal 40 bit transaction time is

$$T_{\text{trans40,min}} = \frac{40}{25\text{MHz}} = 1.6\ \mu\text{s}$$

which leads to to data rate of 400 ns/byte.

The XHFC-2SU SPI protocol defines the following rules for transactions:

1. The SPI master is allowed to stop the SPI clock at any time. When the SPI clock is restarted afterwards, the transaction is continued as if it has not been stopped.
2. When a write or read transaction is split with $/\text{SPISEL} = '1'$ within the control byte, it is ignored and the next received byte is expected to be a control byte.
3. When a write transaction is split with $/\text{SPISEL} = '1'$ within the data byte, it is ignored and the next received byte is expected to be the data byte again.
4. When a read transaction is split with $/\text{SPISEL} = '1'$ within the data byte, the transaction quits immediately (SPI_TX is always tri-state when $/\text{SPISEL} = '1'$). The next received byte is expected to be the data byte again.

2.3.4 Register write access

Register write accesses consist always of a transaction sequence with an address write transaction first and one or several data write transactions afterwards. XHFC-2SU offers four ways of executing register write accesses:

1. A register write access is a sequence of two SPI write transactions as shown in Figure 2.14. With the first transaction the SPI master specifies the register address (control byte is '01X0CCCC'). Afterwards, the new register value is transferred from the SPI master to XHFC-2SU (control byte is '0000 0000').
 'X' = '0' disables broadcast so that 'CCCC' must specify the desired microchip. Alternatively, broadcast is enabled with 'X' = '1' and 'CCCC' is ignored in this case.
2. It is allowed to execute multiple data write transactions to the same register address without address write transactions in between. This is shown in Figure 2.15 and is typically used for transmit FIFO data.
3. Another way of writing multiple bytes into the same register is available with the 40 bit write transaction (Figure 2.16). With the first transaction (16 bits) the SPI master specifies the register address (control byte is '01X0CCCC'). Afterwards, four new register values are transferred from the SPI master to XHFC-2SU (control byte is '0010 0000').
 Broadcasting is handled in the same way as described in (1).
4. Finally, a combination of (2) and (3) can be used to write a multiple of four bytes. First, the control byte '01X0CCCC' executes the address write transaction, and afterwards four data bytes can be written several times with the control byte '0010 0000' in each 40 bit write transaction.

Figures 2.14 to 2.16 show $/\text{SPISEL} = '1'$ between the transactions. This splits the sequence into an address write transaction and one or several data write transactions. The split time has an arbitrary duration. It can also decrease to zero, which means that $/\text{SPISEL}$ remains '0' for several transactions.

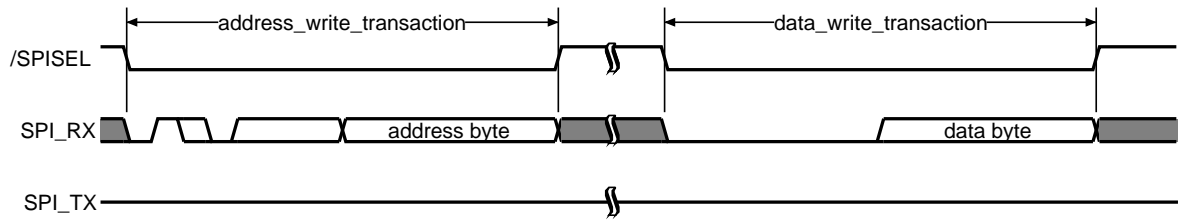


Figure 2.14: Register write access (transaction sequence)

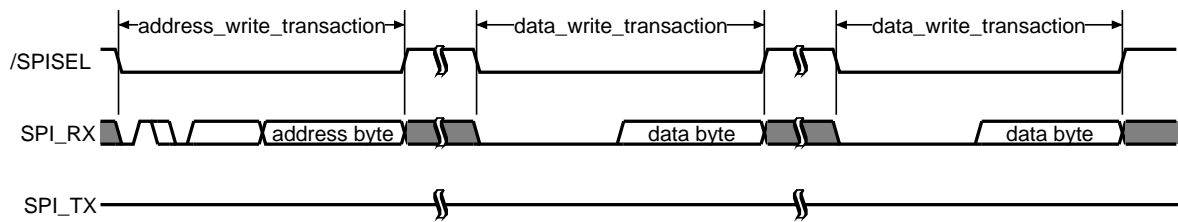


Figure 2.15: Multiple write accesses to the same register

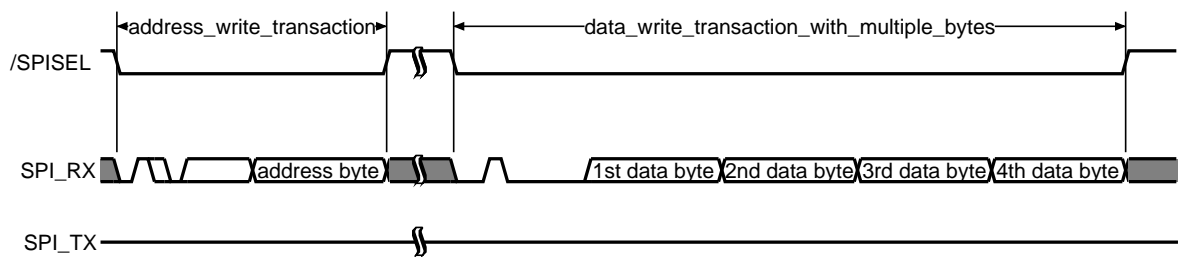


Figure 2.16: Register write access with a 40 bit transaction

The XHFC-2SU SPI protocol defines the following rules for transaction sequences:

1. Every data write transaction stores the received data byte into the register which has been selected with the last address write transaction.
2. When several consecutive address write transactions occur, the last address write transaction specifies the address for the next data access. All previous address write accesses are ignored, i.e. they have no effect to the XHFC-2SU status.

2.3.5 Register read access

Register read accesses consist always of a transaction sequence with an address write transaction first and one or several data read transactions afterwards. XHFC-2SU offers four ways of executing register read accesses:

1. A register read access is a sequence of one SPI write address transactions and one SPI read data transaction as shown in Figure 2.17. With the first transaction the SPI master specifies the register address (control byte is '01X0 CCCC'). Afterwards, XHFC-2SU transfers the register value to the SPI master (control byte is '1000 0000').

'X' = '0' disables broadcast so that 'CCCC' must specify the desired microchip. Alternatively, broadcast is enabled with 'X' = '1' and 'CCCC' is ignored in this case.

2. It is allowed to execute multiple data read transactions to the same register address without address write transactions in between. This is shown in Figure 2.18 and is typically used for receiving FIFO data.
3. Another way of reading multiple bytes from the same register is available with the 40 bit read transaction (Figure 2.19). With the first transaction (16 bits) the SPI master specifies the register address (control byte is '01X0 CCCC'). Afterwards, four register values are transferred from XHFC-2SU to the SPI master (control byte is '1010 0000').

Broadcasting is handled in the same way as described in (1).

4. Finally, a combination of (2) and (3) can be used to read a multiple of four bytes. First, the control byte '01X0 CCCC' executes the address write transaction, and afterwards four data bytes can be read several times with the control byte '1010 0000' in each 40 bit read transaction.

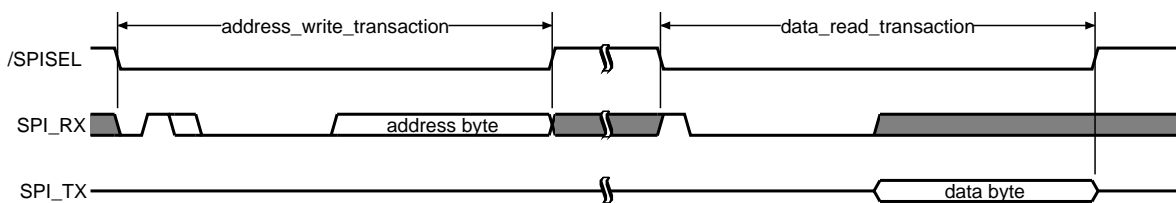


Figure 2.17: Register read access (transaction sequence)

Figures 2.17 to 2.19 show $\text{/SPISEL} = '1'$ between the transactions. This splits the sequence into an address write transaction and one or several data read transactions. The split time has an arbitrary duration. It can also decrease to zero, which means that /SPISEL remains '0' for several transactions.

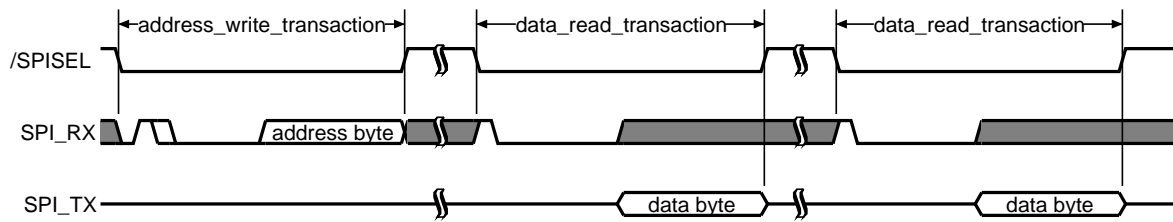


Figure 2.18: Multiple read accesses to the same register

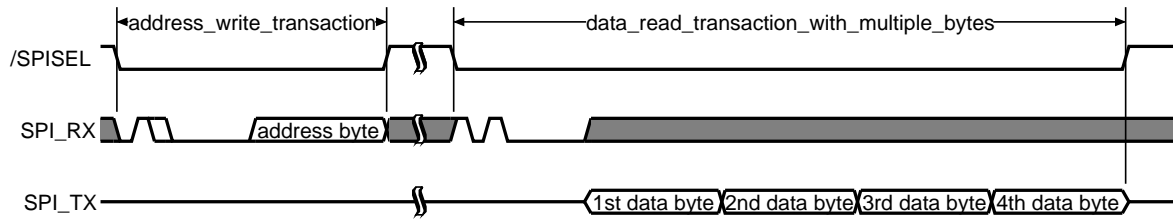


Figure 2.19: Register read access with a 40 bit transaction

2.3.6 Register access duration

There are several SPI data rates due to the different register access types. Table 2.16 gives some examples for $f_{\text{SPICLK}} = 25\text{MHz}$ and either one or 16 accesses to the same register.

Sequence duration and data rates are always calculated without split times neither within nor between transactions.

Table 2.16: SPI data rates

SPI data transaction control byte	Single or multiple data bytes	Number of accesses to the same register	Minimum number of clock cycles	Sequence duration	Data rate for $f_{\text{SPICLK}} = 25\text{MHz}$ and $N = 16$, e.g.
'X000 0000'	single	1	32	$T = \frac{32}{f_{\text{SPICLK}}}$	$DR = \frac{1}{T} = 781\text{ kByte/s}$
'X000 0000'	single	N	$(N + 1) \cdot 16$	$T = \frac{(N+1) \cdot 16}{f_{\text{SPICLK}}}$	$DR = \frac{N}{T} = 1471\text{ kByte/s}$
'X010 0000'	multiple	4	56	$T = \frac{56}{f_{\text{SPICLK}}}$	$DR = \frac{4}{T} = 1786\text{ kByte/s}$
'X010 0000'	multiple	$4N$	$16 + N \cdot 40$	$T = \frac{16+N \cdot 40}{f_{\text{SPICLK}}}$	$DR = \frac{4N}{T} = 2439\text{ kByte/s}$

2.3.7 Register address read-back capability

The address read transaction with the control byte '11X0 CCCC' can be executed to read the address of the currently selected register.

This address read transaction does not perform a register read access. Thus it can be used even on those registers that change their contents on a read access, i.e. register contents is changed not until a data read transaction has been executed.

2.3.8 Problems with interrupts during transaction sequences

The address read-back capability is useful for interrupt procedures, e.g., to save and restore the previous state:

```
interrupt procedure: - execute address read transaction and store the register address
                   - ... (execute the interrupt service routine)
                   - address write transaction to restore the previous register address
```

This procedure is important to avoid data read or write to an unexpected register address after the transaction sequence has been split between transactions by an interrupt service routine which executes any access to XHFC-2SU. Please note, that transactions are not allowed to be split from the SPI master (see Section 2.3.2).

2.3.9 SPI timing diagrams

Figure 2.20 shows the timing diagram for data write transactions (data from master to slave). Four different variations of the SPI clock are shown. SPI clock can be inverted with $SPI_INV = '1'$. Further, idle intervals are allowed during non-selected phases where $/SPISEL = '1'$. During idle intervals, SPI clock must be low for $SPI_INV = '0'$ and it must be high otherwise.

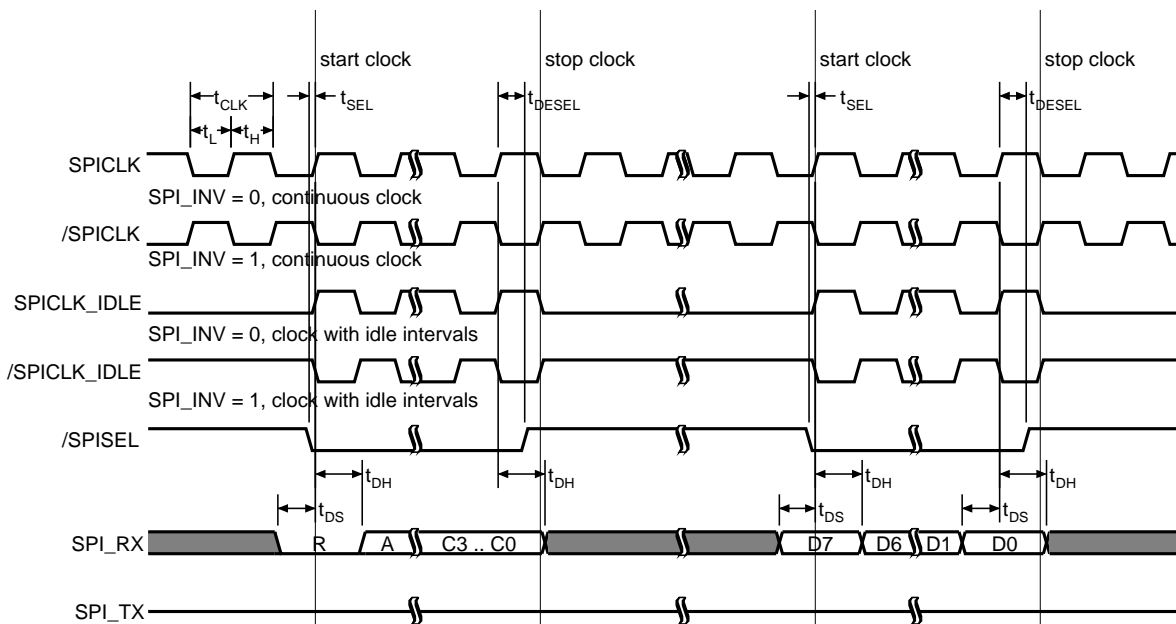


Figure 2.20: SPI timing diagram for data write transactions

Address and data is sampled on the rising edge of SPI_CLK when $SPI_INV = '0'$. The SPI selection signal $/SPISEL$ must be '0' with this edge at the latest.

When the SPI access is splitted by the SPI master, the rising edge of $/SPISEL$ is required to be $t_{DESEL} \geq 2\text{ns}$ after the sampling time at the earliest. When clock idle intervals are used, it is necessary to have at least one clock edge after the last sample edge (marked with 'stop clock' in Figure 2.20). This is important to terminate the access internally.

The timing diagram for data read transactions (data from slave to master) is shown in Figure 2.21. The same different variations of the SPI clock which are shown in Figure 2.20 are valid for data read

Table 2.17: Symbols of write access in Figure 2.20

Symbol	min / ns	max / ns	Characteristic
t_{CLK}	40		SPI clock cycle time
t_L	15		Clock low time
t_H	15		Clock high time
t_{SEL}	0		SPI selection to sample edge setup time
t_{DESEL}	2		SPI deselection delay
t_{DS}	7		Data setup time
t_{DH}	7		Data hold time

transactions but only continuous clock with $SPI_INV = '0'$ is shown here.

Address is sampled on the rising edge of SPI_CLK when $SPI_INV = '0'$. The SPI selection signal $/SPISEL$ must be '0' with this edge at the latest. Data is put out $t_{DEN} \leq 10ns$ after the falling edge of SPI_CLK at the latest.

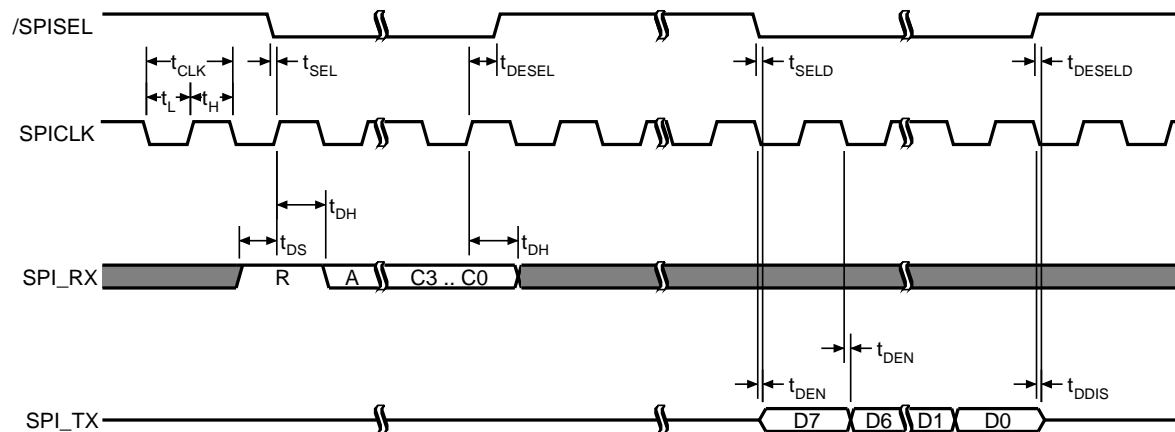


Figure 2.21: SPI timing diagram for data read transactions (see Figure 2.20 for additional clock signals $/SPICLK$, $SPICLK_IDLE$ and $/SPICLK_IDLE$)

Table 2.18: Symbols of read access in Figure 2.21

Symbol	min / ns	max / ns	Characteristic
t_{CLK}	40		SPI clock cycle time
t_{L}	15		Clock low time
t_{H}	15		Clock high time
t_{SEL}	0		SPI selection to sample edge setup time
t_{DESEL}	2		SPI deselection delay
t_{DS}	7		Data setup time
t_{DH}	7		Data hold time
t_{SELD}	0	10	SPI selection to data enable delay
$t_{\text{DESEL D}}$	0		SPI deselection to data disable delay
t_{DEN}	0	10	Clock to data enable setup time
t_{DDIS}	0	10	Clock to data disable setup time

2.3.10 SPI connection circuitry

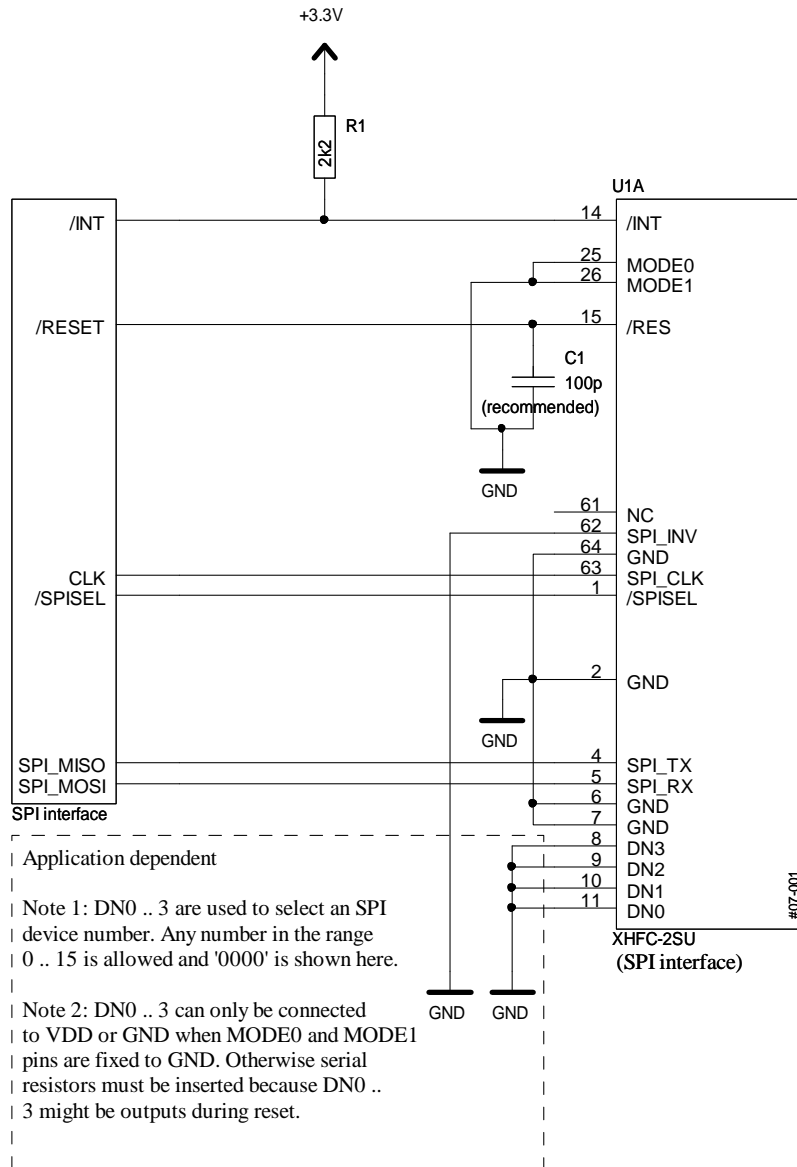


Figure 2.22: SPI connection circuitry

2.4 Auto-EEPROM mode

Please contact Cologne Chip for details if you are interested in using the Auto-EEPROM mode.

2.5 Register description

2.5.1 Write only registers

R_CTRL	(w)	(Reset group: H)	0x01
Common control register			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	V_FIFO_LPRIO	FIFO access priority for host accesses '0' = normal priority '1' = low priority
2	0	(reserved)	Must be '0'.
3	0	V_NT_SYNC	Synchronization source for NT mode The transmit data path of the Universal ISDN Ports can be synchronized to different signals. '0' = F0I is used as NT synchronization source '1' = F1_7 is used as NT synchronization source (see register R_SL_SEL7 for time slot selection) Note: This bit selects the synchronization source for both Universal ISDN Ports in NT/LT mode together. It is ignored in TE mode.
4	0	(reserved)	Must be '0'.
5	0	V_OSC_OFF	Disable oscillator '0' = normal operation '1' = clock oscillator is switched off This bit is reset at every write access to XHFC-2SU or with a wake-up signal on pin WAKEUP. Any chip access is valid not before the oscillator frequency is stable again.
7..6	0	V_SU_CLK	Line interface clock selection The line interface clock f_{SU} is derived from the system clock f_{SYS} '00' = $f_{SYS} / 2$ '01' = $f_{SYS} / 4$ '10' = f_{SYS} (normally unused) '11' = $f_{SYS} / 8$ (normally unused) f_{SU} must be 12.288 MHz.

R_RAM_ADDR	(w)	(Reset group: H, 0)	0x08
Address pointer, lower part			
Lower address byte for SRAM access.			
Bits	Reset value	Name	Description
7..0	0x00	V_RAM_ADDR0	Address bits 7..0

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_RAM_CTRL	(w)	(Reset group: H, 0)	0x09
SRAM access control register			
High address bits and control bits for SRAM access.			
Bits	Reset value	Name	Description
3..0	0	V_RAM_ADDR1	Address bits 11..8
5..4	0	(reserved)	Must be '00'.
6	0	V_ADDR_RES	Address reset '0' = normal operation '1' = address bits 0..11 are set to zero This bit is automatically cleared.
7	0	V_ADDR_INC	Address increment '0' = no address increment '1' = automatically increment of the address after every write or read on register R_RAM_DATA

2.5.2 Read only registers

R_RAM_USE	(r)	(Reset group: –)	0x15
SRAM duty factor			
Usage of SRAM access bandwidth by the internal data processor.			
Bits	Reset value	Name	Description
7..0		V_SRAM_USE	Relative duty factor 0x00 = 0% bandwidth used 0x7C = 100% bandwidth used

R_CHIP_ID	(r)	(Reset group: H)	0x16
Chip identification register			
Bits	Reset value	Name	Description
7..0	0x61	V_CHIP_ID	Chip identification code '0110 0001' (0x61) means XHFC-2SU.

R_CHIP_RV	(r)	(Reset group: H)	0x1F
XHFC-2SU revision			
Bits	Reset value	Name	Description
3..0	0	V_CHIP_RV	Chip revision 0
7..4		(reserved)	

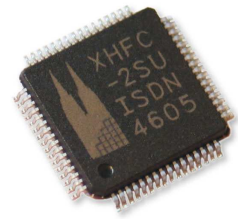
R_INT_DATA	(r)	(Reset group: –)	0x88
Internal data register			
This register can be read to access data with Read* method.			
Bits	Reset value	Name	Description
7..0		V_INT_DATA	Internal data buffer

(See Section 2.2.3.2 on page 45 for details on Read* access.)

2.5.3 Read/write register

R_RAM_DATA	(r*/w)	(Reset group: -)	0xC0
SRAM data access			
Direct access to the internal SRAM			
Bits	Reset value	Name	Description
7..0		V_RAM_DATA	SRAM data access The address must be written into registers R_RAM_ADDR and R_RAM_CTRL in advance.

(See Section 2.2.3.2 on page 45 for details on Read* access.)



Chapter 3

XHFC-2SU data flow

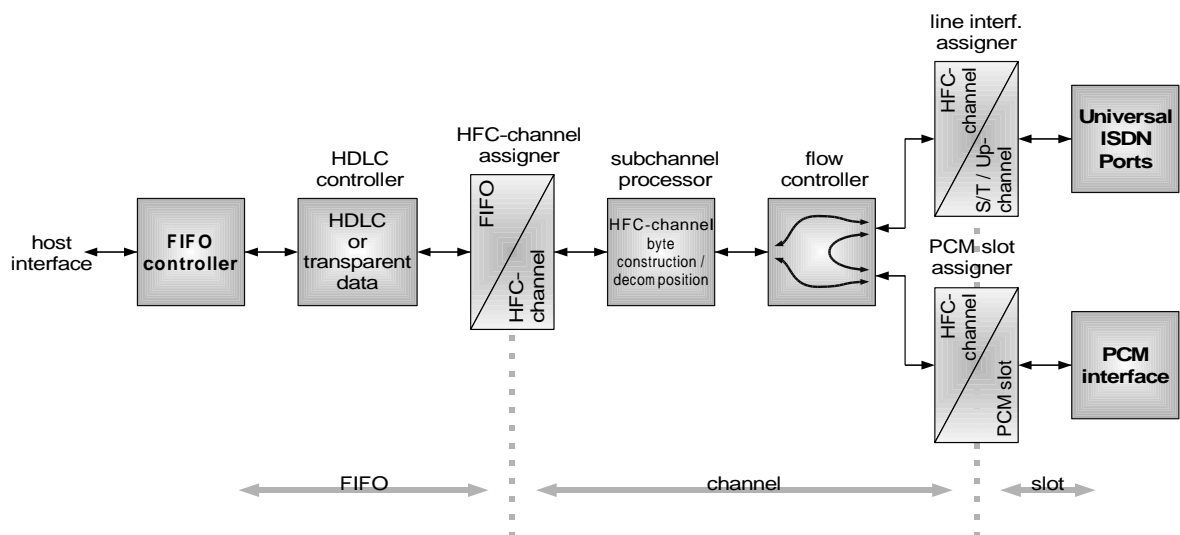


Figure 3.1: Data flow block diagram

3.1 Data flow concept

3.1.1 Overview

XHFC-2SU has a programmable data flow unit, in which the FIFOs are connected to the PCM and the ST/ U_p interfaces. Moreover the data flow unit can directly connect PCM and ST/ U_p interfaces or two PCM time slots with each other¹.

The fundamental features of the XHFC-2SU data flow are as follows:

- programmable interconnection capability between FIFOs, PCM time slots and channels of the Universal ISDN Ports
- 2 Universal ISDN Ports (combined ST/ U_p interfaces)
- in transmit and receive direction there are
 - up to 16 FIFOs each
 - 12, 32, 64 or 128 PCM time slots each in PCM master mode
 - 1 .. 128 PCM time slots each in PCM slave mode
 - 16 HFC-channels each to connect the above-mentioned data interfaces
- 3 data flow modes to satisfy different application tasks
- subchannel processing for bitwise data handling

The complete XHFC-2SU data flow block diagram is shown in Figure 3.1. Basically, data routing requires an allocation number at each block. So there are three areas where numbering is based on FIFOs, HFC-channels and PCM time slots.

FIFO handling and HDLC controller, PCM and ST/ U_p interfaces are described in Chapters 4 to 6. So this chapter deals with the data flow unit which is located between and including the HFC-channel assigner, the PCM slot assigner and the line interface assigner.

3.1.2 Term definitions

Figure 3.2 clarifies the relationship and the differences between the numbering of FIFOs, HFC-channels and PCM time slots. The inner circle symbolizes the HFC-channel oriented part of the data flow, while the outer circle shows the connection of three data sources and data drains respectively. The ST/ U_p interfaces have a fixed mapping between HFC-channels and ST/ U_p -channels so that there is no need of a separate ST/ U_p -channel numbering.

FIFO: The FIFOs are buffers between the microprocessor bus interface and the PCM and ST/ U_p interfaces. The HDLC controllers are located on the non host bus side of the FIFOs. The number of FIFOs depends on the FIFO size configuration (see Section 4.3) and starts with number 0. The maximum FIFO number is 15. Furthermore data directions transmit and receive are associated with every FIFO number.

HFC-channel: HFC-channels are used to define data paths between FIFOs on the one side and PCM and ST/ U_p interfaces on the other side. The HFC-channels are numbered 0 .. 15. Furthermore data directions transmit and receive are associated with every HFC-channel number.

¹In this data sheet the shorter expression “slot” instead of “time slot” is also used with the same meaning.

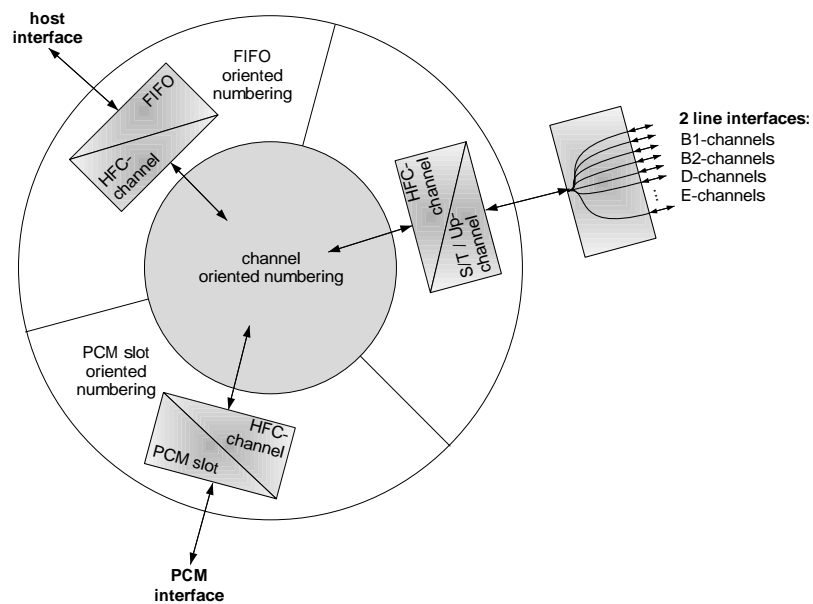


Figure 3.2: Areas of FIFO oriented, HFC-channel oriented and PCM time slot oriented numbering

It is important not to mix up the HFC-channels of the here discussed data flow (inner circle of Figure 3.2) with the ST/ U_p -channels of the multiple ST/ U_p interfaces.

PCM time slot: The PCM data stream is organized in time slots. The number of PCM time slots depends on the data rate, i.e. there are 32 time slots with 2 MBit/s (numbered 0..31), 64 time slots with 4 MBit/s (numbered 0..63) or 128 time slots with 8 MBit/s (numbered 0..127). Every PCM time slot exists both in transmit and receive data directions.

Every FIFO, HFC-channel and PCM time slot number exist for transmit and receive direction. The data rate is always 8 kByte/s for every ST/ U_p -channel and every PCM time slot. FIFOs, HFC-channels, ST/ U_p -channels and PCM time slots have always a width of 8 bit.

3.2 Flow controller

3.2.1 Overview

The various connections between FIFOs, ST/ U_p -channels and PCM time slots are set up by programming the flow controller, the HFC-channel assigner and the PCM slot assigner.

The flow controller sets up connections between FIFOs and the ST/ U_p interfaces, FIFOs and the PCM interface and between the ST/ U_p interfaces and the PCM interface. Bitmap `V_DATA_FLOW` in register `A_CON_HDLC` (which exists for each FIFO) configures these connections. The numbering of transmit and corresponding receive FIFOs, HFC-channels and PCM time slots is independent from each other. But in practice the connection table is more clear if the same number is chosen for corresponding transmit and receive direction.

A direct connection between two PCM time slots can be set up inside the PCM slot assigner and will be described in Section 3.3.

The flow controller operates on HFC-channel data. Nevertheless it is programmed with a bitmap of

a FIFO-indexed array register. With this concept it is possible to change the FIFO-to-HFC-channel assignment of a ready-configured FIFO without re-programming its parameters again.

The internal structure of the flow controller contains

- 4 switching buffers, i.e. one for the ST/ U_p and PCM interfaces in transmit and receive direction each and
- 3 switches to control the data paths.

3.2.2 Switching buffers

The switching buffers decouple the data inside the flow controller from the data that is transmitted to or received from the ST/ U_p and PCM interfaces. With every 125 μ s cycle the switching buffers change their pointers.

If a byte is read from the FIFO and written into a switching buffer, it is transmitted by the connected interface during the *next* 125 μ s cycle. In the reverse case, a received byte which is stored in a switching buffer is copied to the FIFO during the next 125 μ s cycle.

A direct PCM-to-ST/ U_p connection delays each data byte two cycles. That means the received byte is stored in the switching buffer during the first 125 μ s cycle, then copied into the transmit buffer during the second 125 μ s cycle and finally transmitted from the interface during the third 125 μ s cycle.

3.2.3 Timed sequence

The data transmission algorithm of the flow controller is FIFO-oriented and handles all FIFOs, and of course all connected HFC-channels, every 125 μ s in the following sequence:

FIFO[0,TX]
FIFO[0,RX]
FIFO[1,TX]
FIFO[1,RX]
⋮
FIFO[n ,TX]
FIFO[n ,RX]

The number of existing FIFOs, and consequently the value of n , depends on the FIFO configuration (see Table 4.2 on page 128). In any case n cannot exceed 15. There are other configurations with $n = 7$ and $n = 3$.

Due to the FIFO oriented operation, the number of existing FIFOs also defines the number of processed HFC-channels. HFC-channel numbers can be used in the full range 0..15 but with every processed FIFO just one HFC-channel is processed within the same cycle.

If a faulty configuration writes data from several sources into the same switching buffer, the last write access overwrites the previous ones. Only in this case it is necessary to know the process sequence of the flow controller.

XHFC-2SU has three data flow modes. One of them (*FIFO sequence mode*) is used to configure a programmable FIFO sequence which can be used instead of the ascending FIFO numbering. This is explained in Section 3.4.

3.2.4 Transmit operation (FIFO in transmit data direction)

In transmit operation one HDLC or transparent byte is read from a FIFO and can be transmitted to the ST/ U_p and the PCM interface as shown in Figure 3.3. Furthermore, data can be transmitted from the ST/ U_p interface to the PCM interface. From the flow controller point of view, the switches select the source for outgoing data. They are controlled by bitmap $V_DATA_FLOW[2..1]$ in register $A_CON_HDLC[n, TX]$ where n is a FIFO number. Transmit operation is configured with $V_FIFO_DIR = '0'$ in the multi-register R_FIFO .

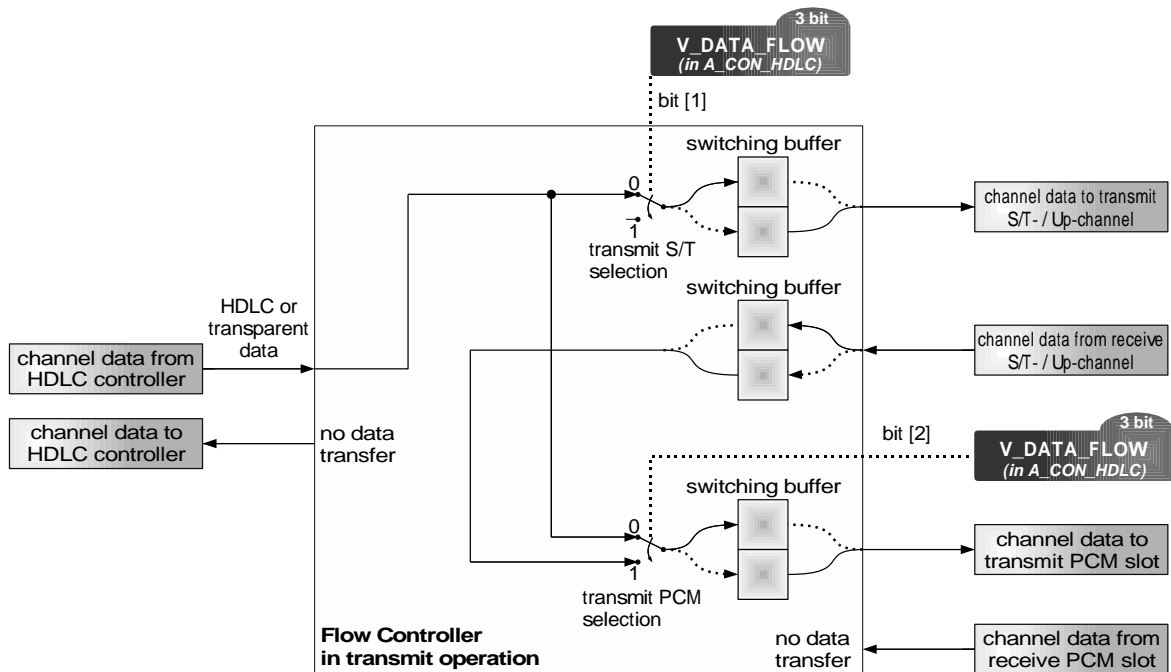


Figure 3.3: The flow controller in transmit operation

- FIFO data is only transmitted to the line interface if $V_DATA_FLOW[1] = '0'$.
- The PCM interface can transmit a data byte which comes either from the FIFO or from the line interface. Bit $V_DATA_FLOW[2]$ selects the source for the PCM transmit slot (see Figure 3.3). The receiving ST/ U_p -channel has always the same number as the transmitting S/T/ U_p -channel.
- Bit $V_DATA_FLOW[0]$ is ignored in transmit operation.

3.2.5 Receive operation (FIFO in receive data direction)

Figure 3.4 shows the flow controller structure in receive operation. The two switches are controlled by bitmap $V_DATA_FLOW[1..0]$ in register $A_CON_HDLC[n, RX]$ where n is a FIFO number. Receive operation is configured with $V_FIFO_DIR = '1'$ in the multi-register R_FIFO . FIFO data can either

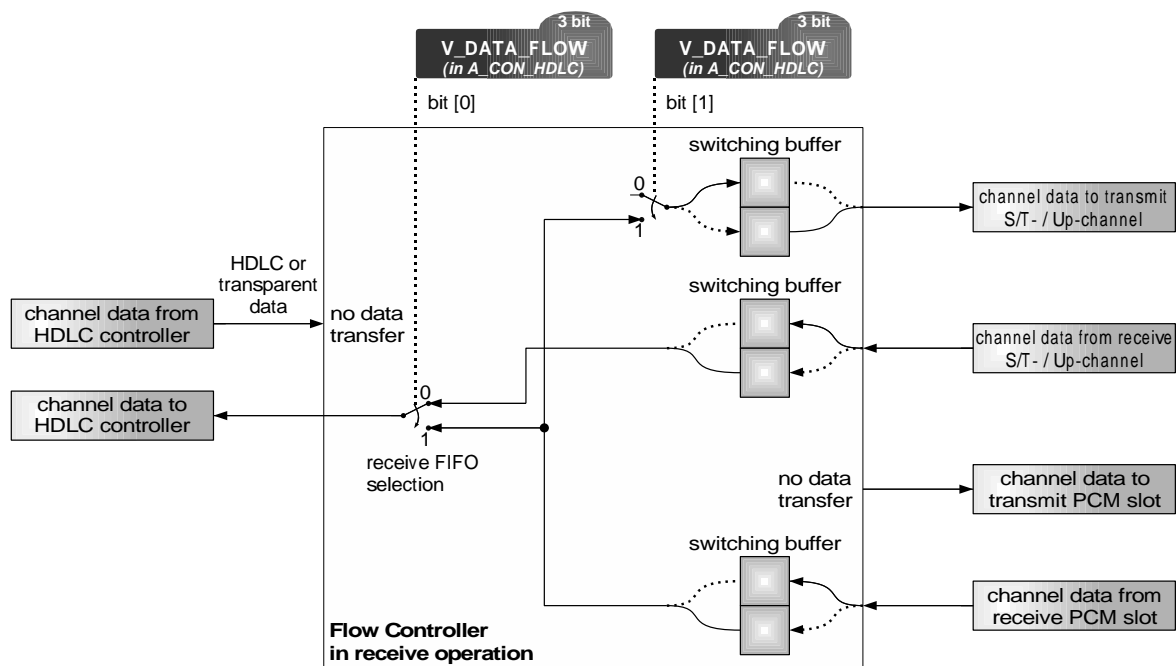


Figure 3.4: The flow controller in receive FIFO operation

be received from the ST/ U_p or from the PCM interface. Furthermore, data can be transmitted from the PCM interface to the ST/ U_p interface.

- Bit V_DATA_FLOW[0] selects the source for the receive FIFO which can either be the PCM or the line interface.
- Furthermore, the received PCM byte can be transferred to the line interface. This requires bit V_DATA_FLOW[1] = '1'.
- Bit V_DATA_FLOW[2] is ignored in receive FIFO operation.

3.2.6 Connection summary

Table 3.1 shows the flow controller connections as a whole. Bidirectional connections² are pointed out with a gray box because they are typically used to establish the data transmissions. All rows have an additional connection to a second destination.

The most important connections are bidirectional data transmissions. For these connections it is possible to manage the configuration programming of V_DATA_FLOW with only three different values for transmit and receive FIFO operations. Table 3.2 shows the suitable programming values which can be used to simplify the programming algorithm.

²In fact, all connections are unidirectional. However, in typical applications there is always a pair of transmit and receive data channels which belong together. Instead of "transmit and corresponding receive data connection" the shorter expression "bidirectional connection" is used in this data sheet.

Table 3.1: Flow controller connectivity

V_DATA_FLOW	Transmit (V_FIFO_DIR = '0')		Receive (V_FIFO_DIR = '1')	
	'000'	FIFO → PCM	FIFO → ST/U _p	FIFO ← ST/U _p
'001'	FIFO → ST/U _p	FIFO → PCM	FIFO ← PCM	
'010'	FIFO → PCM		FIFO ← ST/U _p	ST/U _p ← PCM
'011'		FIFO → PCM	FIFO ← PCM	ST/U _p ← PCM
'100'	ST/U _p → PCM	FIFO → ST/U _p	FIFO ← ST/U _p	
'101'	FIFO → ST/U _p	ST/U _p → PCM	FIFO ← PCM	
'110'		ST/U _p → PCM	ST/U _p ← PCM	FIFO ← ST/U _p
'111'		ST/U _p → PCM	ST/U _p ← PCM	FIFO ← PCM

Table 3.2: V_DATA_FLOW programming values for bidirectional connections

Connection	V_FIFO_DIR	Required	Recommended
		V_DATA_FLOW	V_DATA_FLOW
FIFO → ST/U _p	'0' (TX)	'x0x'	'000'
FIFO ← ST/U _p	'1' (RX)	'xx0'	
FIFO → PCM	'0' (TX)	'0xx'	'001'
FIFO ← PCM	'1' (RX)	'xx1'	
ST/U _p → PCM	'0' (TX)	'1xx'	'110'
ST/U _p ← PCM	'1' (RX)	'x1x'	

3.3 Assigners

The data flow block diagram in Figure 3.1 contains three assigners. These functional blocks are used to connect FIFOs, ST/ U_p -channels and PCM time slots to the HFC-channels.

3.3.1 HFC-channel assigner

The HFC-channel assigner interconnects FIFOs and HFC-channels. Its functionality depends on the data flow mode described in Section 3.4.

3.3.2 PCM slot assigner

The PCM slot assigner can connect each PCM time slot to an arbitrary HFC-channel. Therefore, for a selected time slot³ the connected HFC-channel number and data direction must be written into register A_SL_CFG[SLOT] as follows:

Register setup:

A_SL_CFG[SLOT]: V_CH_SDIR = <HFC-channel data direction>
 : V_CH_SNUM = <HFC-channel number>

Typically, the data direction of a HFC-channel and its connected PCM time slot is the same.

If two PCM time slots are connected to each other, incoming data on a slot is transferred to the PCM slot assigner and stored in the PCM receive switching buffer of the connected HFC-channel. From there it is read (i.e. same HFC-channel) and transmitted to a transmit PCM time slot.

3.3.3 Line interface assigner

Table 3.3 shows the assignment between HFC-channels and the ST/ U_p -channels. There is no possibility to change this allocation, so there is no register for programming the line interface assigner.

If ST/ U_p -channels are coded as

B1-channel = 0
 B2-channel = 1
 D-channel = 2
 E-channel = 3

it is possible to calculate

$$\text{HFC-channel number} = \text{interface number} \cdot 4 + \text{S/T-channel code} .$$

For a given HFC-channel number the belonging ST/ U_p -channel is calculated with⁴

$$\begin{aligned} \text{interface number} &= \text{HFC-channel number} \text{ div } 4 \\ \text{ST}/U_p\text{-channel code} &= \text{HFC-channel number} \text{ mod } 4 . \end{aligned}$$

³A time slot is specified by writing its number and data direction into register R_SLOT. Then all accesses to the slot array registers belong to this time slot. Please see Chapter 6 for details.

⁴div is the integer division. mod is the division remainder. $i \text{ mod } j = (i/j - i \text{ div } j) \cdot j$.

Table 3.3: Line interface assigner

HFC-channel		ST/U _p -channel			HFC-channel		ST/U _p -channel		
number	direction	interface	channel	direction	number	direction	interface	channel	direction
[0,TX]		#0	B1	TX	[4,TX]		#1	B1	TX
[0,RX]		#0	B1	RX	[4,RX]		#1	B1	RX
[1,TX]		#0	B2	TX	[5,TX]		#1	B2	TX
[1,RX]		#0	B2	RX	[5,RX]		#1	B2	RX
[2,TX]		#0	D	TX	[6,TX]		#1	D	TX
[2,RX]		#0	D	RX	[6,RX]		#1	D	RX
[3,TX]		#0	BAC/S	TX	[7,TX]		#1	BAC/S	TX
[3,RX]		#0	E	RX	[7,RX]		#1	E	RX

In both cases the equivalence

$$\text{HFC-channel direction} = \text{ST/U}_p\text{-channel direction}$$

is valid.

3.3.4 Assigner summary

The three different assigner types of XHFC-2SU are shown in Figure 3.5. Assigner programming is always handled with array registers. This can be a FIFO array register or a PCM slot array register.

- The line interface assigner is not programmable. Every HFC-channel is connected to a specific ST/U_p-channel like shown in Table 3.3.
- The PCM slot assigner is programmed by register A_SL_CFG[SLOT]. The PCM time slot must be selected before by writing the desired slot number and direction into register R_SLOT.
- The HFC-channel assigner programming depends on the data flow mode which is described in Section 3.4. This section explains in what cases the assigner is programmable and how this can be done. Figure 3.5 gives a hint, that the programming procedure is handled with array register A_CHANNEL[FIFO]. Please see section 3.4 for details and restrictions.

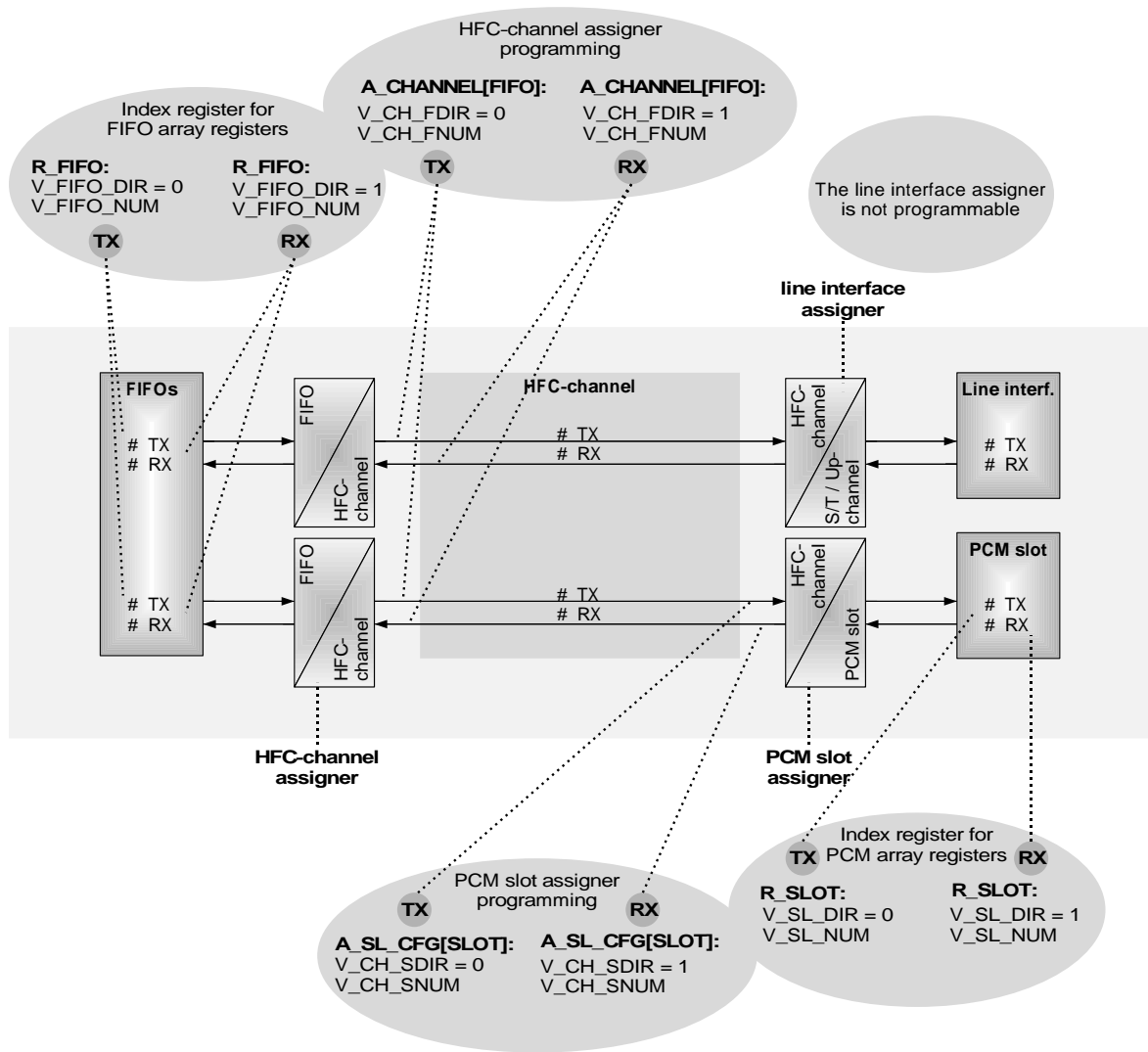


Figure 3.5: Overview of the assigner programming

3.4 Data flow modes

The internal operation of the HFC-channel assigner and the subchannel processor as well depends on the selected data flow mode. Three modes are available and will be described in this section:

- *Simple Mode* (SM),
- *Channel Select Mode* (CSM) and
- *FIFO Sequence Mode* (FSM)

Various array registers are available to configure the data flow. Unused FIFOs and PCM time slots should remain in their reset state.

FIFO array registers are indexed by R_FIFO in most cases. But there are some exceptions depending on the data flow mode and the target array register. Table 3.4 shows all FIFO array registers and their index registers at the different data flow modes.

3.4.1 Simple Mode (SM)

3.4.1.1 Mode description

In *Simple Mode* (SM) only one-to-one connections are possible. That means one FIFO, one ST/U_p-channel or one PCM time slot can be connected to each other. The number of connections is limited by the number of FIFOs. It is possible to establish as many connections as there are FIFOs⁵. The actual number of FIFOs depends on the FIFO setup (see Section 4.3).

Simple Mode is selected with V_DF_MD = '00' in register R_FIFO_MD. All FIFO array registers are indexed by the multi-register R_FIFO (address 0x0F) in this data flow mode.

The FIFO number is always the same as the HFC-channel number. Thus, the HFC-channel assigner cannot be programmed in *Simple Mode*. In contrast to this, the PCM time slot number can be chosen independently from the HFC-channel number.

Due to the fixed correspondence between FIFO number and HFC-channel, a pair of transmit and receive FIFOs is allocated even if a bidirectional data connection between the PCM interface and the line interface is established without using the FIFO. Nevertheless, in this case the FIFO must be enabled to enable the data transmission.

A direct coupling of two PCM time slots uses a PCM switching buffer and no FIFO has to be enabled. This connection requires a HFC-channel number (resp. the same FIFO number). An arbitrary HFC-channel number can be chosen. If there are less than 16 transmit and receive FIFOs each, it is useful to choose a HFC-channel number that is greater than the maximum FIFO number. This saves FIFO resources where no data is stored in a FIFO.

⁵Except PCM-to-PCM connections which do not need a FIFO resource if the involved HFC-channel number is higher than the maximum FIFO number.

Table 3.4: Index registers of the FIFO array registers (sorted by address)

Context	Array register		I/O mode	Index register in			Index meaning in		
	name	address		SM	CSM	FSM	SM	CSM	FSM
FIFO data counters	0x04	A_Z1[FIFO]	r	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
	0x06	A_Z2[FIFO]	r	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
FIFO frame counters	0x0C	A_F1[FIFO]	r	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
	0x0D	A_F2[FIFO]	r	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
FIFO configuration	0x0E	A_INC_RES_FIFO[FIFO]	w	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
FIFO data access	0x80	A_FIFO_DATA[FIFO]	r/w	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
	0x84	A_FIFO_DATA_NOINC[FIFO]	r/w	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
Subchannel processor	0xF4	A_CH_MSK[FIFO]	r*/w	R_FIFO	R_FIFO	R_FIFO	HFC-channel	HFC-channel	HFC-channel
FIFO configuration	0xFA	A_CON_HDLC[FIFO]	r*/w	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO
Subchannel Processor	0xFB	A_SUBCH_CFG[FIFO]	r*/w	R_FIFO	R_FIFO	R_FSM_IDX	FIFO	FIFO	list index
FIFO configuration	0xFC	A_CHANNEL[FIFO]	r*/w	R_FIFO	R_FIFO	R_FSM_IDX	FIFO	FIFO	list index
FIFO configuration	0xFD	A_FIFO_SEQ[FIFO]	r*/w	R_FIFO	R_FIFO	R_FSM_IDX	FIFO	FIFO	list index
FIFO configuration	0xFF	A_FIFO_CTRL[FIFO]	r*/w	R_FIFO	R_FIFO	R_FIFO	FIFO	FIFO	FIFO

**Please note !**

The index of FIFO array registers is always denoted '[FIFO]' even if the meaning is different from this in particular cases (grey marked fields in Table 3.4).

3.4.1.2 Subchannel processing

If the data stream of a FIFO does not require full 8 kByte/s data rate, the subchannel processor might be used. Unused bits can be masked out and replaced by bits of an arbitrary mask byte which can be specified in A_CH_MSK.

For D- and E-channel processing the subchannel functionality must be enabled. Only two bits of a data byte are processed every 125 μ s.

In transparent mode only the non-masked bits of a byte are processed. Masked bits are taken from register A_CH_MSK. So the effective FIFO data rate always remains 8 kByte/s whereas the usable data rate depends on the number of non-masked bits.

In HDLC mode the data rate of the FIFO is reduced according to how many bits are not masked out.

Please see Section 3.5 from page 108 for details concerning the subchannel processor.

3.4.1.3 Example for SM

Figure 3.6 shows an example with four bidirectional connections (❶ FIFO-to-ST/U_p, ❷ FIFO-to-PCM, ❸ PCM-to-ST/U_p and ❹ PCM-to-PCM). The FIFO box on the left side contains the number and direction information of the used FIFOs. The ST/U_p and PCM boxes on the right side contain the ST/U_p-channels and PCM time slot numbers and directions which are used in this example. Black lines illustrate data paths, whereas dotted lines symbolize blocked resources. These are not used for the data transmission, but they are necessary to enable the settings.



Please note !

All settings in Figure 3.6 are configured in bidirectional data paths due to typical applications of XHFC-2SU. However, transmit and receive directions are independent from each other and could occur one at a time as well.

The following settings demonstrate the required register values to establish the connections. All involved FIFOs have to be enabled with V_FIFO_IRQ \neq 0 in register A_CON_HDLC[FIFO].

The subchannel processor is not used in this example. For this reason, registers A_SUBCH_CFG and A_CH_MSK remain in their reset state.

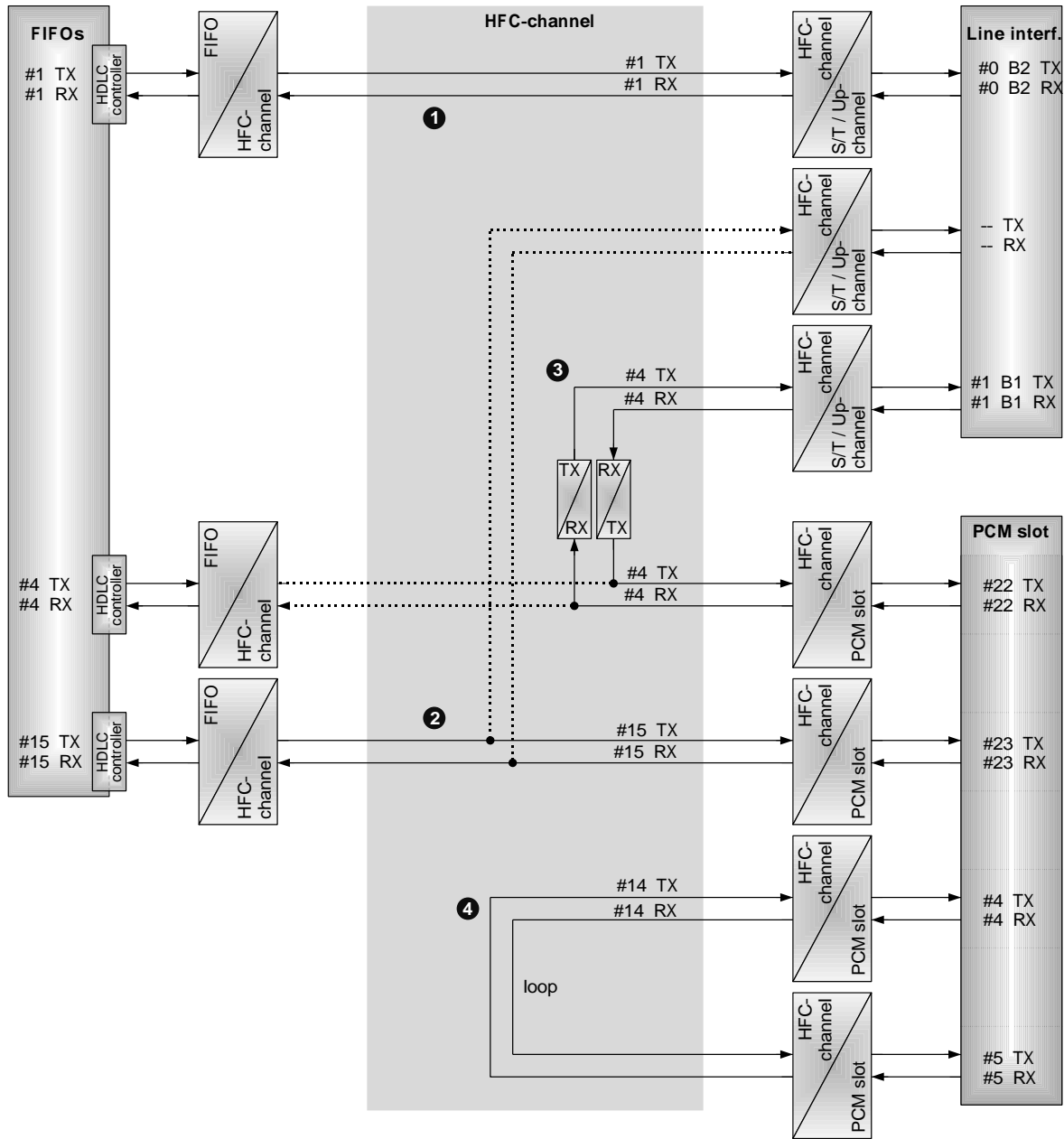


Figure 3.6: SM example

① FIFO-to-ST/U_p

As HFC-channel and FIFO numbers are the same in SM, a selected ST/U_p-channel specifies the corresponding FIFO (and same in inverse, of course). There is no need of programming the HFC-channel assigner.

To set up a FIFO-to-ST/U_p connection, the desired ST/U_p -channel has to be chosen and the linked FIFO (see Table 3.3) has to be programmed. Due to the user's requirements, V_REV can be programmed either to normal or inverted bit order of the FIFO data.

HDLC or transparent mode (V_HDLC_TRP) can freely be chosen as well. In addition to the settings shown here, a periodic interrupt (in transparent mode) or a *end of frame* interrupt (in HDLC mode) can be enabled.

Register setup:		(SM ① TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 1	(FIFO #1)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[1, TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)

Register setup:		(SM ① RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 1	(FIFO #1)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[1, RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)

② FIFO-to-PCM

The FIFO-to-PCM connection can use different numbers for the involved HFC-channels and PCM time slots. The desired numbers are linked together in the PCM slot assigner.

As the line interface assigner links the HFC-channels to the ST/U_p-channels, every used HFC-channel blocks the connected ST/U_p-channel. In this example a FIFO is chosen, which is not assigned to a line interface channel. Thus there are no resources blocked.

Again, V_REV and V_HDLC_TRP can freely be chosen according to the user's requirements. As in the previous setting, a periodic interrupt in transparent mode or a *end of frame* interrupt in HDLC mode can be enabled.

Register setup:		(SM ② TX)	
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)	
	: V_FIFO_NUM = 15	(FIFO #15)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[15,TX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '001'	(FIFO → ST/U _p , FIFO → PCM)	
R_SLOT	: V_SL_DIR = 0	(transmit slot)	
	: V_SL_NUM = 23	(slot #23)	
A_SL_CFG[23,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)	
	: V_CH_SNUM = 15	(HFC-channel #15)	
	: V_ROUT = '10'	(data to pin STIO1)	

Register setup:		(SM ② RX)	
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)	
	: V_FIFO_NUM = 15	(FIFO #15)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[15,RX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '001'	(FIFO ← PCM)	
R_SLOT	: V_SL_DIR = 1	(receive slot)	
	: V_SL_NUM = 23	(slot #23)	
A_SL_CFG[23,RX]	: V_CH_SDIR = 1	(receive HFC-channel)	
	: V_CH_SNUM = 15	(HFC-channel #15)	
	: V_ROUT = '10'	(data from pin STIO2)	

③ PCM-to-ST/U_p

A direct PCM-to-ST/U_p coupling is shown in the third connection set. The array registers of FIFO[4,TX] and FIFO[4,RX] contain the data flow settings, so they must be configured and the FIFOs must be enabled to switch on the data transmission. This is done with V_FIFO_IRQ ≠ 0 in register A_CON_HDLC[FIFO].

In receive direction, data is stored in the connected FIFO. But it is not used and needs not to be read. A FIFO overflow has no effect and can be ignored. Consequently, the V_HDLC_TRP setting has no effect to the transferred data between the PCM and the ST/U_p interface neither in receive nor in transmit direction. A PCM-to-ST/U_p connection operates always in transparent mode.

For a PCM-to-ST/U_p connection, the data direction changes between the two interfaces. In detail, data is received on a RX line and then transmitted on a TX line to the other interface. Therefore, a TX-RX-exchanger is inserted for this connection. The blocked FIFOs are on the PCM side of the TX-RX-exchanger. Like shown in the register setting below, data direction of FIFO, ST/U_p and PCM lines are never mixed up when programming the assigners in *Simple Mode*.

Register setup:		(SM ③ TX)	
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)	
	: V_FIFO_NUM = 4	(FIFO #4)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[4,TX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 1	(transparent mode)	
	: V_FIFO_IRQ = 7	(enable data transmission)	
	: V_DATA_FLOW = '110'	(ST/U _p → PCM)	
R_SLOT	: V_SL_DIR = 0	(transmit slot)	
	: V_SL_NUM = 22	(slot #22)	
A_SL_CFG[22,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)	
	: V_CH_SNUM = 4	(HFC-channel #4)	
	: V_ROUT = '10'	(data to pin STIO1)	

Register setup:		(SM ③ RX)	
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)	
	: V_FIFO_NUM = 4	(FIFO #4)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[4,RX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 1	(transparent mode)	
	: V_FIFO_IRQ = 7	(enable data transmission)	
	: V_DATA_FLOW = '110'	(FIFO ← ST/U _p , ST/U _p ← PCM)	
R_SLOT	: V_SL_DIR = 1	(receive slot)	
	: V_SL_NUM = 22	(slot #22)	
A_SL_CFG[22,RX]	: V_CH_SDIR = 1	(receive HFC-channel)	
	: V_CH_SNUM = 4	(HFC-channel #4)	
	: V_ROUT = '10'	(data from pin STIO2)	

④ PCM-to-PCM

A PCM-to-PCM configuration does not occupy any FIFO resources. An example is shown in the last connection set. HFC-channel[14,RX] is used to connect PCM slot[4,RX] to PCM slot[5,TX]. Data from PCM slot[5,RX] to PCM slot[4,TX] is transferred through HFC-channel[14,TX].

A HFC-channel loop is easily established by linking two PCM time slots to the same HFC-channel. It is a good choice to select a HFC-channel which is not assigned to a ST/U_p -channel to avoid waste of resources.

Register setup:		(SM ④ no. 1)
R_SLOT	: V_SL_DIR = 1	(receice slot)
	: V_SL_NUM = 4	(slot #4)
A_SL_CFG[4,RX]	: V_CH_SDIR = 1	(receive HFC-channel)
	: V_CH_SNUM = 14	(HFC-channel #14)
	: V_ROUT = '11'	(data from pin STIO1)
R_SLOT	: V_SL_DIR = 0	(transmit slot)
	: V_SL_NUM = 5	(slot #5)
A_SL_CFG[5,TX]	: V_CH_SDIR = 1	(receive HFC-channel)
	: V_CH_SNUM = 14	(HFC-channel #14)
	: V_ROUT = '11'	(data to pin STIO2)

Register setup:		(SM ④ no. 2)
R_SLOT	: V_SL_DIR = 1	(receice slot)
	: V_SL_NUM = 5	(slot #5)
A_SL_CFG[5,RX]	: V_CH_SDIR = 0	(transmit HFC-channel)
	: V_CH_SNUM = 14	(HFC-channel #14)
	: V_ROUT = '10'	(data from pin STIO2)
R_SLOT	: V_SL_DIR = 0	(transmit slot)
	: V_SL_NUM = 4	(slot #4)
A_SL_CFG[4,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)
	: V_CH_SNUM = 14	(HFC-channel #14)
	: V_ROUT = '10'	(data to pin STIO1)



Rule

In *Simple Mode* for every used FIFO[*n*] the HFC-channel[*n*] is also used. This is valid in reverse case, too, except for PCM-to-PCM configurations.

3.4.2 Channel Select Mode (CSM)

3.4.2.1 Mode description

The *Channel Select Mode* (CSM) allows an arbitrary assignment between a FIFO and the connected HFC-channel as shown in Figure 3.7 (left side). Beyond this, it is possible to connect several FIFOs to one HFC-channel (Fig. 3.7, right side). This works in transmit and receive direction and can be used to connect one 8 kByte/s ST/U_p-channel or PCM time slot to multiple FIFO data streams, with lower data rate each. In this case the subchannel processor must be used.



Figure 3.7: HFC-channel assigner in CSM

Channel Select Mode is selected with $V_DF_MD = '01'$ in register R_FIFO_MD . All FIFO array registers are indexed by the multi-register R_FIFO (address $0x0F$) in this data flow mode.

3.4.2.2 HFC-channel assigner

The connection between a FIFO and a HFC-channel can be established by register $A_CHANNEL$ which exists for every FIFO. For a selected FIFO, the HFC-channel to be connected must be written into V_CH_FNUM of register $A_CHANNEL$. Typically, the data direction in V_CH_FDIR is the same as the FIFO data direction V_FIFO_DIR in the multi-register R_FIFO . With the following register settings the HFC-channel assigner connects the selected FIFO to HFC-channel n .

Register setup:

$A_CHANNEL[FIFO]: V_CH_FDIR = V_FIFO_DIR$
 $: V_CH_FNUM = n$

A direct connection between a PCM time slot and an ST/U_p-channel allocates one FIFO although this FIFO does not store any data. In *Channel Select Mode* – in contrast to *Simple Mode* – an arbitrary FIFO can be chosen. This FIFO must be enabled to switch on the data transmission.

3.4.2.3 Subchannel Processing

If more than one FIFO is connected to one HFC-channel, this HFC-channel number must be written into the V_CH_FNUM bitmap of all these FIFOs. In this case every FIFO contributes one or more bits to construct one HFC-channel byte. Unused bits of a HFC-channel byte can be set with an arbitrary mask byte in register A_SUBCH_CFG .

In transparent mode the FIFO data rate always remains 8 kByte/s. In HDLC mode the FIFO data rate is determined by the number of bits transmitted to the HFC-channel.

Please see Section 3.5 on page 108 for details concerning the subchannel processor.

3.4.2.4 Example for CSM

The example for a *Channel Select Mode* configuration in Figure 3.8 shows three bidirectional connections (❶ FIFO-to-ST/Up, ❷ FIFO-to-PCM and ❸ PCM-to-ST/Up). The black lines illustrate data paths, whereas the dotted lines symbolize blocked resources. These are not used for data transmission, but they are necessary to enable the settings.

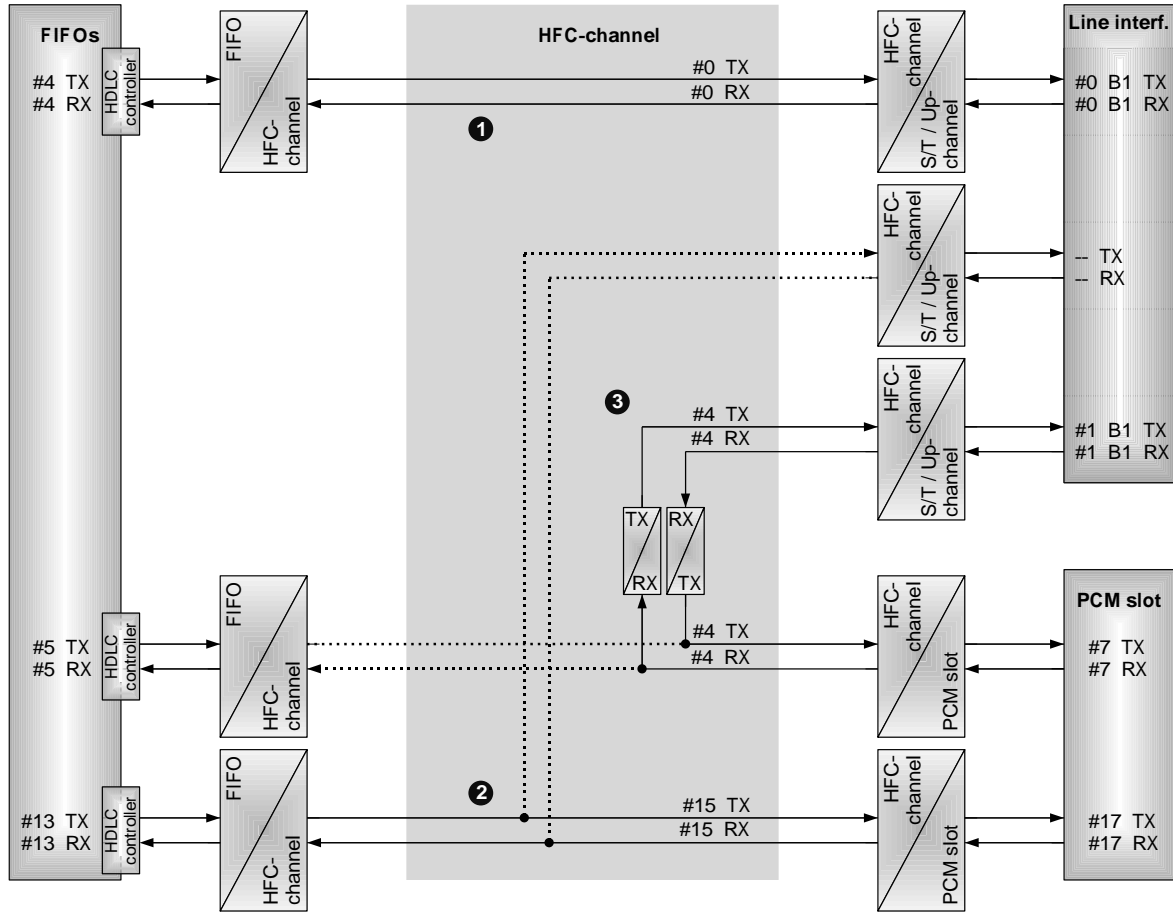


Figure 3.8: CSM example

The following settings demonstrate the required register values to establish the connections. All involved FIFOs have to be enabled with $V_FIFO_IRQ \neq 0$ in register $A_CON_HDLC[FIFO]$.

The subchannel processor is not used in this example. For this reason, registers A_SUBCH_CFG and A_CH_MSK remain in their reset state.

① FIFO-to-ST/U_p

HFC-channel and FIFO numbers can be chosen independently from each other. This is shown in the FIFO-to-ST/U_p connection.

Due to the user's requirements, V_REV can be programmed either to normal or inverted bit order of the FIFO data.

HDLC or transparent mode (V_HDLC_TRP) can freely be chosen as well. In addition to the settings shown here, a periodic interrupt (in transparent mode) or a *end of frame* interrupt (in HDLC mode) can be enabled.

Register setup:		(CSM ① TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 4	(FIFO #4)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[4,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)
A_CHANNEL[4,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 0	(HFC-channel #0)

Register setup:		(CSM ① RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 4	(FIFO #4)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[4,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)
A_CHANNEL[4,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 0	(HFC-channel #0)

② FIFO-to-PCM

The FIFO-to-PCM connection blocks one transmit and one receive ST/ U_p -channel .

In this example, the selected HFC-channel is not assigned to a line interface channel. So there are no resources blocked.

Again, V_REV and V_HDLC_TRP can freely be chosen according to the user's requirements. As in the previous setting, HDLC mode is selected and the FIFOs are enabled with $V_FIFO_IRQ = 1$.

Register setup:		(CSM ② TX)	
R_FIFO	: $V_FIFO_DIR = 0$		(transmit FIFO)
	: $V_FIFO_NUM = 13$		(FIFO #13)
	: $V_REV = 0$		(normal bit order)
A_CON_HDLC[13,TX]	: $V_IFF = 0$		(0x7E as inter frame fill)
	: $V_HDLC_TRP = 0$		(HDLC mode)
	: $V_FIFO_IRQ = 7$		(enable FIFO)
	: $V_DATA_FLOW = '001'$		(FIFO → ST/ U_p , FIFO → PCM)
A_CHANNEL[13,TX]	: $V_CH_FDIR = 0$		(transmit HFC-channel)
	: $V_CH_FNUM = 15$		(HFC-channel #15)
R_SLOT	: $V_SL_DIR = 0$		(transmit slot)
	: $V_SL_NUM = 17$		(slot #17)
A_SL_CFG[17,TX]	: $V_CH_SDIR = 0$		(transmit HFC-channel)
	: $V_CH_SNUM = 15$		(HFC-channel #15)
	: $V_ROUT = '10'$		(data to pin STIO1)

Register setup:		(CSM ② RX)	
R_FIFO	: $V_FIFO_DIR = 1$		(receive FIFO)
	: $V_FIFO_NUM = 13$		(FIFO #13)
	: $V_REV = 0$		(normal bit order)
A_CON_HDLC[13,RX]	: $V_IFF = 0$		(0x7E as inter frame fill)
	: $V_HDLC_TRP = 0$		(HDLC mode)
	: $V_FIFO_IRQ = 7$		(enable FIFO)
	: $V_DATA_FLOW = '001'$		(FIFO ← PCM)
A_CHANNEL[13,RX]	: $V_CH_FDIR = 1$		(receive HFC-channel)
	: $V_CH_FNUM = 15$		(HFC-channel #15)
R_SLOT	: $V_SL_DIR = 1$		(receive slot)
	: $V_SL_NUM = 17$		(slot #17)
A_SL_CFG[17,RX]	: $V_CH_SDIR = 1$		(receive HFC-channel)
	: $V_CH_SNUM = 15$		(HFC-channel #15)
	: $V_ROUT = '10'$		(data from pin STIO2)

③ PCM-to-ST/ U_p

The PCM-to-ST/ U_p connection blocks one transmit and one receive FIFO. Although these FIFOs are not used, they must be enabled to switch on the data transmission between the PCM and the ST/ U_p interface.

In receive direction, data is stored in the connected FIFO. But it is not used and needs not to be read. A FIFO overflow has no effect and can be ignored. Consequently, the V_HDLC_TRP setting has no effect to the transferred data between the PCM and the ST/U_p interface neither in receive nor in transmit direction. A PCM-to-ST/U_p connection operates always in transparent mode.

For a PCM-to-ST/U_p connection, the data direction changes between the two interfaces. In detail, data is received on a RX line and then transmitted on a TX line to the other interface. Therefore, a TX-RX-exchanger is inserted for this connection. The blocked FIFOs are on the PCM side of the TX-RX-exchanger, typically.⁶

Register setup:		(CSM ③ TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 5	(FIFO #5)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[5,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable data transmission)
	: V_DATA_FLOW = '110'	(ST/U _p → PCM)
A_CHANNEL[5,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 4	(HFC-channel #4)
R_SLOT	: V_SL_DIR = 0	(transmit slot)
	: V_SL_NUM = 7	(slot #7)
A_SL_CFG[7,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)
	: V_CH_SNUM = 4	(HFC-channel #4)
	: V_ROUT = '10'	(data to pin STIO1)

Register setup:		(CSM ③ RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 5	(FIFO #5)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[5,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable data transmission)
	: V_DATA_FLOW = '110'	(FIFO ← ST/U _p , ST/U _p ← PCM)
A_CHANNEL[5,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 4	(HFC-channel #4)
R_SLOT	: V_SL_DIR = 1	(receive slot)
	: V_SL_NUM = 7	(slot #7)
A_SL_CFG[7,RX]	: V_CH_SDIR = 1	(receive HFC-channel)
	: V_CH_SNUM = 4	(HFC-channel #4)
	: V_ROUT = '10'	(data from pin STIO2)

⁶It is not forbidden to connect the blocked FIFOs at the ST/U_p side of the TX-RX-exchanger. 'Advanced users' might find configurations where this is useful. But all typical configuration settings do not require this exceptional option.

**Rule**

In *Channel Select Mode*

- Every used HFC-channel requires at least one enabled FIFO (except for the PCM-to-PCM connection) with the same data direction.
- Every used PCM time slot requires one HFC-channel (except for the PCM-to-PCM connection where a full duplex connection with four time slots allocates only two HFC-channels).

3.4.3 FIFO Sequence Mode (FSM)

3.4.3.1 Mode description

In contrast to the PCM and ST/ U_p -channels, the FIFO data rate is not fixed to 8 kByte/s in *FIFO Sequence Mode*. In the previous section the CSM allows the functional capability of a FIFO data rate with less than 8 kByte/s. This section shows how to use FIFOs with a data rate which is higher than 8 kByte/s. In transmit direction one FIFO can cyclically distribute its data to several HFC-channels. In opposite direction, received data from several HFC-channels can be collected cyclically in one FIFO (see Fig. 3.9, right side). A one-to-one connection between FIFO and HFC-channel is also possible in FSM, of course (Fig. 3.9, left side).



Figure 3.9: HFC-channel assigner in FSM

FIFO Sequence Mode is selected with $V_DF_MD = '11'$ in register R_FIFO_MD . This data flow mode selects the multi-register R_FSM_IDX at the address $0x0F$ for some FIFO array registers (see Table 3.4 on page 86).

3.4.3.2 FIFO sequence

To achieve a FIFO data rate higher than 8 kByte/s, a FIFO must be connected to more than one HFC-channel. As there is only one register $A_CHANNEL[FIFO]$ for each FIFO, the FSM programming method must differ from the previous modes. Some array registers which are indexed by R_FIFO must be indexed by R_FSM_IDX in *FIFO Sequence Mode* (see Table 3.4).

In FSM all FIFOs are organized in a list with up to 32 entries. Every list entry is assigned to a FIFO. The FIFO configuration can be set up as usual, i.e. HFC-channel allocation, flow controller programming and subchannel processing can be configured as described in the previous sections. Additionally, each list entry specifies the next FIFO of the sequence. The list is terminated by an 'end of list' entry. This procedure is shown in Figure 3.10 with $j + 1$ list entries. The first FIFO of the sequence must be specified in register R_FIRST_FIFO .

A quite simple FSM configuration with every FIFO and every HFC-channel specified only one time in the list, would have the same data transmission result as the CSM with an equivalent FIFO \longleftrightarrow HFC-channel setup. But if a specific FIFO is selected n times in the list and connected to n different HFC-channels, the FIFO data rate is $n \cdot 8$ kByte/s.

The complete list is processed every $125 \mu s$ with ascending list index beginning with 0. Suppose the transmit FIFO m occurs several times in the list. Then the first FIFO byte is transferred to the first connected HFC-channel, the second byte of FIFO m to the second connected HFC-channel and so on. This is similar in receive data direction. The first byte written into FIFO m comes from the first connected HFC-channel, the second byte from the second connected HFC-channel and so on.

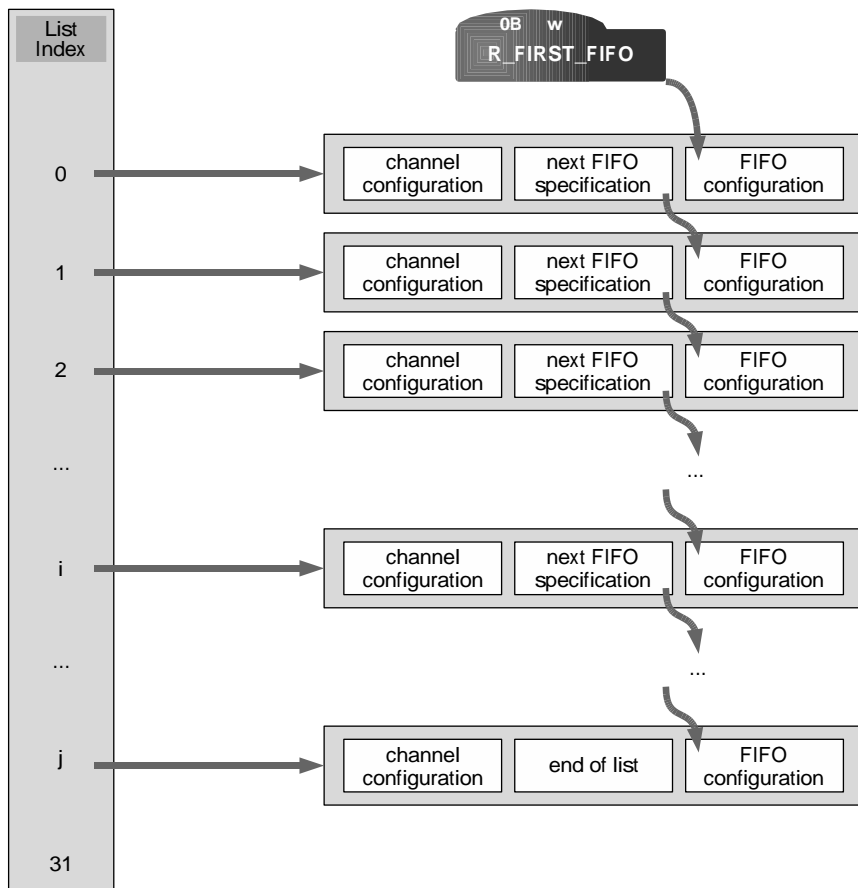


Figure 3.10: FSM list processing

3.4.3.3 FSM programming

Register `R_FSM_IDX` specifies the list index with bitmap `V_IDX` in the range of $0..31$. `R_FSM_IDX` is a multi-register and has the same address as `R_FIFO` because in FSM it replaces `R_FIFO` for the list programming of the HFC-channel based registers. The array registers `A_CHANNEL`, `A_FIFO_SEQ` and `A_SUBCH_CFG` are indexed with the list index `V_IDX` instead of the FIFO number (see Table 3.4 on page 86). All other FIFO array registers remain indexed by `R_FIFO`.

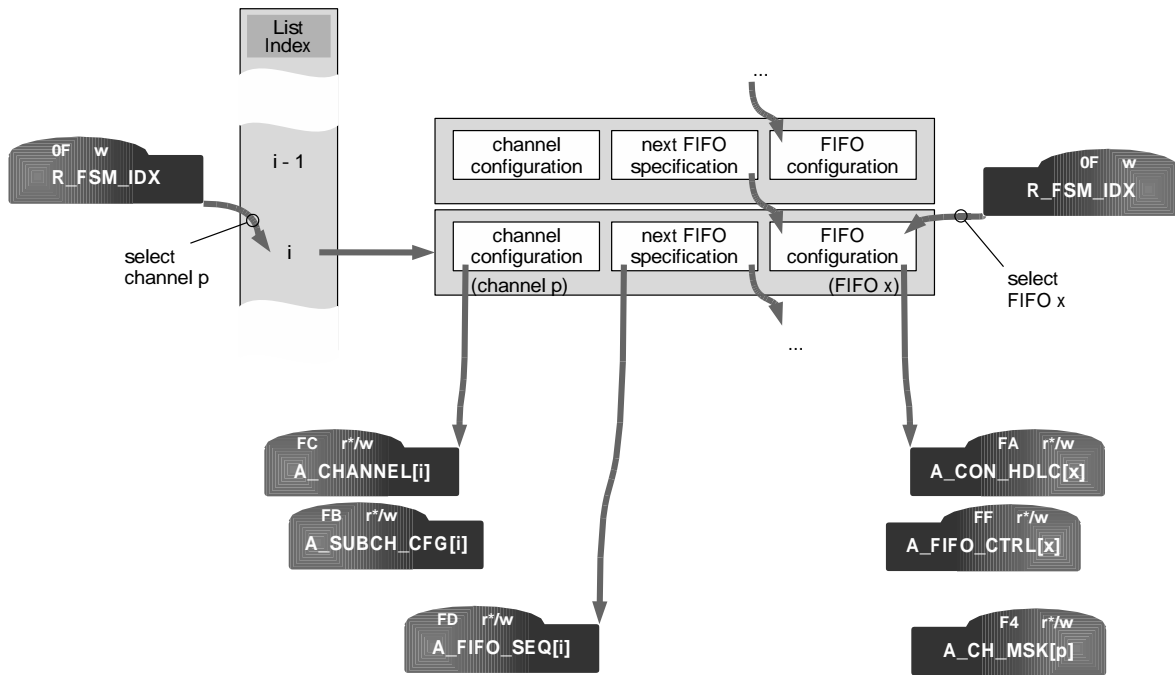


Figure 3.11: FSM list programming

The first processed FIFO has to be specified in register `R_FIRST_FIFO` with the direction bit `V_FIRST_FIFO_DIR` and the FIFO number `V_FIRST_FIFO_NUM`. The next FIFO has to be specified in register `A_FIFO_SEQ[V_IDX = 0]`.

A FIFO handles more than one HFC-channel if a FIFO is specified several times in the 'next FIFO' entries.

The FIFO sequence list terminates with `V_SEQ_END = '1'` in register `A_FIFO_SEQ`. The other list entries must specify `V_SEQ_END = '0'` to continue the sequence processing with the next entry.

Programming of the HFC-channel and FIFO registers is shown in Figure 3.11. The connected HFC-channel array registers are indexed by the list index which is written into register `R_FSM_IDX`. On the other hand, FIFO array registers are indexed by register `R_FIFO` as usual.

- After writing the list index i into register `R_FSM_IDX`, the registers `A_CHANNEL[i]` and `A_SUBCH_CFG[i]` can be programmed to assign and configure an HFC-channel.
- The next FIFO in the sequence must be specified in register `A_FIFO_SEQ[i]`.
- Supposed, that the previous list entry $i - 1$ has specified `A_FIFO_SEQ[i - 1] = FIFO x`, then the corresponding FIFO array registers have to be programmed by first setting `R_FIFO = x`. Afterwards, registers `A_CON_HDLC[x]`, `A_FIFO_CTRL[x]` and `A_CH_MSK` can be programmed

in the usual way. Please note, that register A_CH_MSK requires the addressed HFC-channel to be specified in register R_FIFO (see remark on page 109).

3.4.3.4 Example for FSM

Figure 3.12 shows an example with three bidirectional connections (❶ 8 kByte/s-FIFO-to-ST / U_p , ❷ 8 kByte/s-FIFO-to-PCM and ❸ 16 kByte/s-FIFO-to-ST / U_p). The black lines illustrate data paths, whereas the dotted lines symbolize blocked HFC-channels. These are not used for data transmission, but they are necessary to enable the settings.

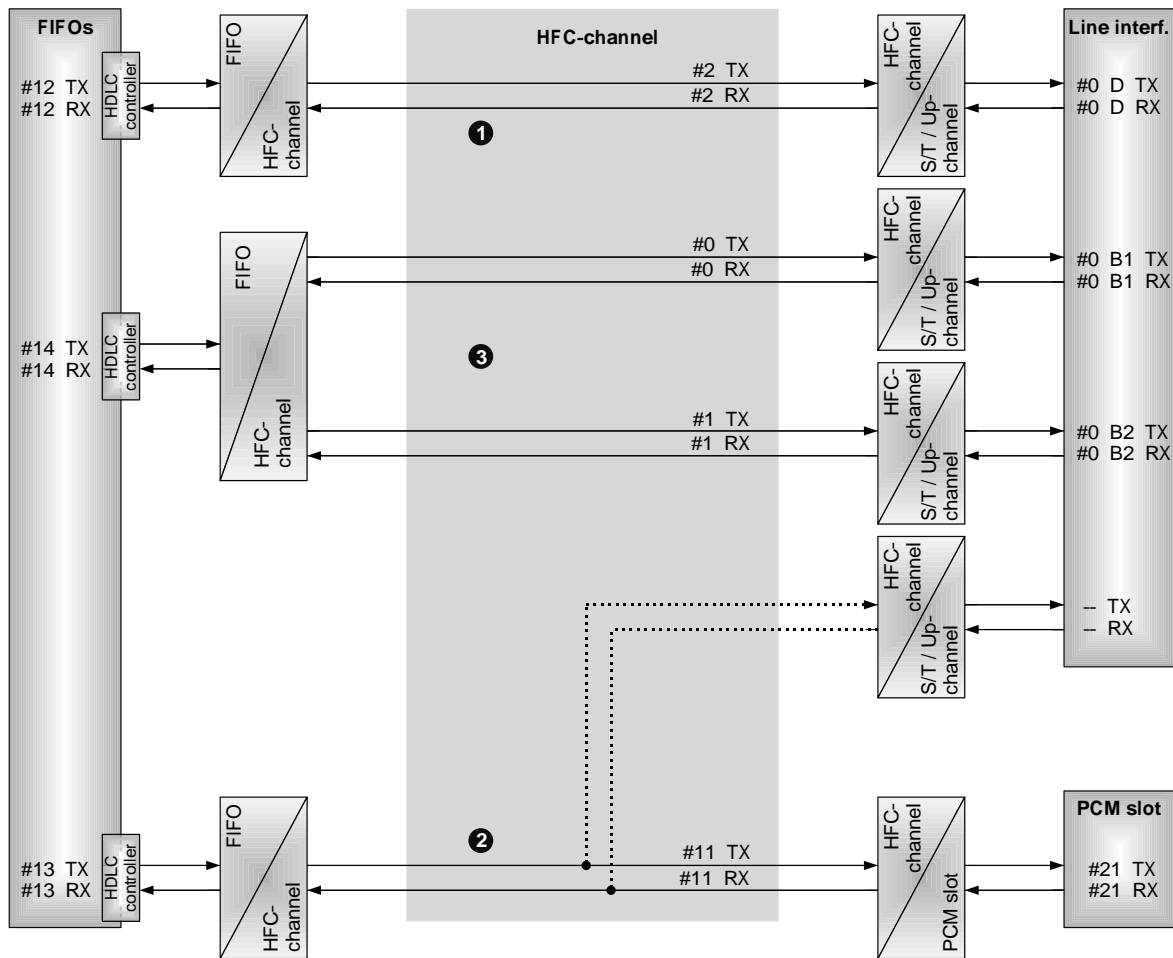


Figure 3.12: FSM example

The following settings demonstrate the required register values to establish the connections. All involved FIFOs have to be enabled with V_FIFO_IRQ ≠ 0 in register A_CON_HDLC[FIFO].

The subchannel processor is not used in this example. For this reason, registers A_SUBCH_CFG and A_CH_MSK remain in their reset state.

All FIFOs can be arranged in arbitrary order. In the example the list specification of Table 3.5 is chosen. To select FIFO[12,TX] being the first FIFO, R_FIRST_FIFO is set as follows:

Register setup:

R_FIRST_FIFO : V_FIRST_FIFO_DIR = '0' (transmit FIFO)
 : V_FIRST_FIFO_NUM = 12 (FIFO #12)

Table 3.5: List specification of the example in Figure 3.12

Example number	List index	Connection	
①	0	FIFO[12,TX]	→ ST/U _p interf. #0, D TX
①	1	FIFO[12,RX]	← ST/U _p interf. #0, D RX
②	2	FIFO[13,RX]	← PCM time slot[21,RX]
②	3	FIFO[13,TX]	→ PCM time slot[21,TX]
③	4	FIFO[14,TX]	→ ST/U _p interf. #0, B1 TX
③	5	FIFO[14,RX]	← ST/U _p interf. #0, B1 RX
③	6	FIFO[14,TX]	→ ST/U _p interf. #0, B2 TX
③	7	FIFO[14,RX]	← ST/U _p interf. #0, B2 RX

❶ FIFO-to-ST/U_p

The bidirectional FIFO-to-ST/U_p connection use the list indices 0 and 1. Registers A_CHANNEL and A_FIFO_SEQ are indexed by the list index.

Register setup:		(FSM ❶ list indices 0 and 1)	
R_FSM_IDX	: V_IDX	= 0	(List index #0, used for FIFO[12,TX])
A_CHANNEL[#0]	: V_CH_FDIR	= 0	(transmit HFC-channel)
	: V_CH_FNUM	= 2	(HFC-channel #2)
A_FIFO_SEQ[#0]	: V_NEXT_FIFO_DIR	= 1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	= 12	(next: FIFO #12)
	: V_SEQ_END	= 0	(continue)
R_FSM_IDX	: V_IDX	= 1	(List index #1, used for FIFO[12,RX])
A_CHANNEL[#1]	: V_CH_FDIR	= 1	(receive HFC-channel)
	: V_CH_FNUM	= 2	(HFC-channel #2)
A_FIFO_SEQ[#1]	: V_NEXT_FIFO_DIR	= 1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	= 13	(next: FIFO #13)
	: V_SEQ_END	= 0	(continue)

The FIFO programming sequence is indexed by the FIFO number and direction. V_REV, V_HDLC_TRP and V_FIFO_IRQ can be programmed due to the user's requirements. FIFO[12,TX] and FIFO[12,RX] must both be enabled.

Register setup:		(FSM ❶ FIFO programming for list indices 0 and 1)	
R_FIFO	: V_FIFO_DIR	= 0	(transmit FIFO)
	: V_FIFO_NUM	= 12	(FIFO #12)
	: V_REV	= 0	(normal bit order)
A_CON_HDLC[12,TX]	: V_IFF	= 0	(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0	(HDLC mode)
	: V_FIFO_IRQ	= 7	(enable FIFO)
	: V_DATA_FLOW	= '000'	(FIFO → ST/U _p , FIFO → PCM)
R_FIFO	: V_FIFO_DIR	= 1	(receive FIFO)
	: V_FIFO_NUM	= 12	(FIFO #12)
	: V_REV	= 0	(normal bit order)
A_CON_HDLC[12,RX]	: V_IFF	= 0	(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0	(HDLC mode)
	: V_FIFO_IRQ	= 7	(enable FIFO)
	: V_DATA_FLOW	= '000'	(FIFO ← ST/U _p)

❷ FIFO-to-PCM

The following two list entries (indices 2 and 3) define the bidirectional FIFO-to-PCM connection.

Using two HFC-channels which are not assigned to line interface channels does not block any resources.

Register setup:		(FSM ② list indices 2 and 3)	
R_FSM_IDX	: V_IDX = 2	(List index #2, used for FIFO[13,RX])	
A_CHANNEL[#2]	: V_CH_FDIR = 1	(receive HFC-channel)	
	: V_CH_FNUM = 11	(HFC-channel #11)	
A_FIFO_SEQ[#2]	: V_NEXT_FIFO_DIR = 0	(next: transmit FIFO)	
	: V_NEXT_FIFO_NUM = 13	(next: FIFO #13)	
	: V_SEQ_END = 0	(continue)	
R_FSM_IDX	: V_IDX = 3	(List index #3, used for FIFO[13,TX])	
A_CHANNEL[#3]	: V_CH_FDIR = 0	(transmit HFC-channel)	
	: V_CH_FNUM = 11	(HFC-channel #11)	
A_FIFO_SEQ[#3]	: V_NEXT_FIFO_DIR = 0	(next: transmit FIFO)	
	: V_NEXT_FIFO_NUM = 14	(next: FIFO #14)	
	: V_SEQ_END = 0	(continue)	

Register setup:		(FSM ② RX FIFO programming for list indices 2 and 3)	
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)	
	: V_FIFO_NUM = 13	(FIFO #13)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[13,RX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '001'	(FIFO ← PCM)	
R_SLOT	: V_SL_DIR = 1	(receive slot)	
	: V_SL_NUM = 21	(slot #21)	
A_SL_CFG[21,RX]	: V_CH_SDIR = 1	(receive HFC-channel)	
	: V_CH_SNUM = 11	(HFC-channel #11)	
	: V_ROUT = '10'	(data from pin STIO2)	

Register setup:		(FSM ② TX FIFO programming for list indices 2 and 3)	
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)	
	: V_FIFO_NUM = 13	(FIFO #13)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[13,TX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '001'	(FIFO → ST/U _p , FIFO → PCM)	
R_SLOT	: V_SL_DIR = 0	(transmit slot)	
	: V_SL_NUM = 21	(slot #21)	
A_SL_CFG[21,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)	
	: V_CH_SNUM = 11	(HFC-channel #11)	
	: V_ROUT = '10'	(data to pin STIO1)	

③ FIFO to multiple ST/U_p-channels

The last setting shows a channel bundling configuration of one FIFO to two B-channels of the ST/U_p interface for both transmit and receive directions. The FIFOs have a data rate of 16 kByte/s each.

Register setup:		(FSM ③ list indices 4 and 5)	
R_FSM_IDX	: V_IDX = 4	(List index #4, used for FIFO[14,TX])	
A_CHANNEL[#4]	: V_CH_FDIR = 0	(transmit HFC-channel)	
	: V_CH_FNUM = 0	(HFC-channel #0)	
A_FIFO_SEQ[#4]	: V_NEXT_FIFO_DIR = 1	(next: receive FIFO)	
	: V_NEXT_FIFO_NUM = 14	(next: FIFO #14)	
	: V_SEQ_END = 0	(continue)	
R_FSM_IDX	: V_IDX = 5	(List index #5, used for FIFO[14,RX])	
A_CHANNEL[#5]	: V_CH_FDIR = 1	(receive HFC-channel)	
	: V_CH_FNUM = 0	(HFC-channel #0)	
A_FIFO_SEQ[#5]	: V_NEXT_FIFO_DIR = 0	(next: transmit FIFO)	
	: V_NEXT_FIFO_NUM = 14	(next: FIFO #14)	
	: V_SEQ_END = 0	(continue)	

Register setup:		(FSM ③ FIFO programming for list indices 4 and 5)	
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)	
	: V_FIFO_NUM = 14	(FIFO #14)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[14,TX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)	
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)	
	: V_FIFO_NUM = 14	(FIFO #14)	
	: V_REV = 0	(normal bit order)	
A_CON_HDLC[14,RX]	: V_IFF = 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP = 0	(HDLC mode)	
	: V_FIFO_IRQ = 7	(enable FIFO)	
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)	

When the FIFO[14,TX] and FIFO[14,RX] are used for the second time, they need not to be programmed again. So just the HFC-channels have to be programmed for the list indices #6 and #7.

Register setup:

(FSM ③ list indices 6 and 7)

R_FSM_IDX	: V_IDX	= 6	(List index #6, used for FIFO[14,TX])
A_CHANNEL[#6]	: V_CH_FDIR	= 0	(transmit HFC-channel)
	: V_CH_FNUM	= 1	(HFC-channel #1)
A_FIFO_SEQ[#6]	: V_NEXT_FIFO_DIR	= 1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	= 14	(next: FIFO #14)
	: V_SEQ_END	= 0	(continue)
R_FSM_IDX	: V_IDX	= 7	(List index #7, used for FIFO[14,RX])
A_CHANNEL[#7]	: V_CH_FDIR	= 1	(receive HFC-channel)
	: V_CH_FNUM	= 1	(HFC-channel #1)
A_FIFO_SEQ[#7]	: V_NEXT_FIFO_DIR	= 0	
	: V_NEXT_FIFO_NUM	= 0	
	: V_SEQ_END	= 1	(end of list)

3.5 Subchannel Processing

3.5.1 Overview

Data transmission between a FIFO and the connected HFC-channel can be controlled by the sub-channel processor. The behavior of this functional unit depends on the selected data flow mode (SM, CSM or FSM) and the operation mode of the HDLC controller (transparent or HDLC mode). The subchannel controller allows to process less than 8 bits of the transferred FIFO data bytes.

The subchannel processor cannot be used for direct PCM-to-ST/Up or PCM-to-PCM connections, because a FIFO must participate in the data flow.

A general overview of the subchannel processor in transmit direction is shown as a simplified example in Figure 3.13. Three transmit FIFOs are connected to one HFC-channel. Details of subchannel processing are described in the following sections, partitioned into the different modes of the data flow and the HDLC controller.

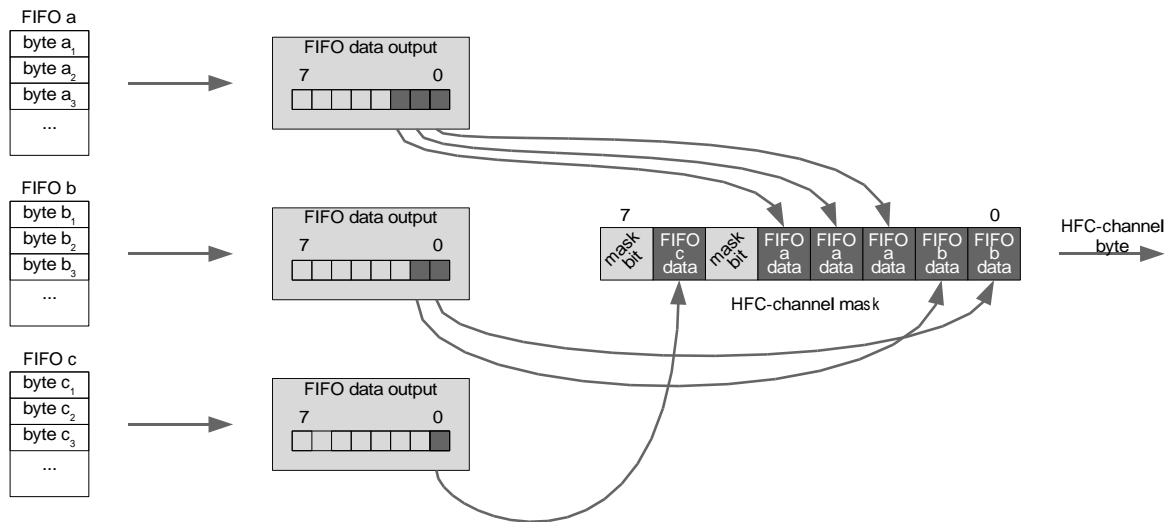


Figure 3.13: General structure of the subchannel processor shown with an example of three connected FIFOs

The essence of the subchannel processor is a bit extraction /insertion unit for every FIFO and a byte mask for every HFC-channel. Therefore, the subchannel processor is divided into two parts A and B like shown in Figure 3.14. The behaviour of the FIFO oriented part A depends on the HDLC or transparent mode selection. The HFC-channel oriented part B has a different behaviour due to the selected data flow SM or CSM/FSM.

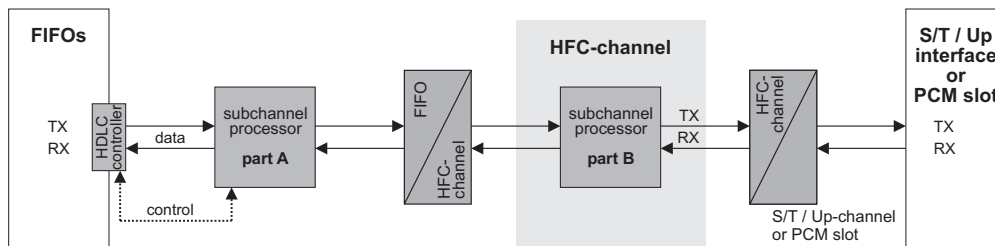


Figure 3.14: Location of the subchannel parts A and B in the data flow diagram

3.5.2 Subchannel registers

The FIFO bit extraction /insertion requires two register settings. `V_BIT_CNT` defines the number of bits to be extracted /inserted. These bits are always aligned to position 0 in the FIFO data. This bit field can freely be placed in the HFC-channel byte. For this, the start bit can be selected with `V_START_BIT` in the range of 0..7. Both values are located in register `A_SUBCH_CFG[FIFO]`.

The HFC-channel mask can be stored in register `A_CH_MSK[FIFO]`. This mask is only used for transmit data. The processed FIFO bits are stored in this register, so it must be re-initialized after changing the settings in `A_SUBCH_CFG[FIFO]`. Each HFC-channel has its own mask byte. To write this byte for HFC-channel $[n, TX]$, the HFC-channel must be written into the multi-register `R_FIFO` first. The desired mask byte m can be written with `A_CH_MSK = m` after this index selection.



Important !

Typically, the multi-register `R_FIFO` contains always a FIFO index. There is one exception where the `R_FIFO` value has a different meaning: The HFC-channel mask byte `A_CH_MSK` is programmed by writing the HFC-channel into the `R_FIFO` register.

The default subchannel configuration in register `A_SUBCH_CFG` leads to a transparent behavior. That means, only complete data bytes are transmitted in receive and transmit direction.

3.5.3 Details of the FIFO oriented part of the subchannel processor (part A)

The subchannel processor part A lies between the HDLC controller and the HFC-channel assigner. Figure 3.15 shows the block diagram for both receive and transmit data directions.

At the HDLC controller side, there are a data path and two control lines. These communicate the number of bits to be processed and the HDLC /transparent mode selection between the two modules. In transparent mode always one byte is transferred between the HDLC controller and the subchannel controller part A every 125 μ s cycle. In HDLC mode the number of bits is specified by the subchannel bitmap `V_BIT_CNT` in register `A_SUBCH_CFG[FIFO]`.

On the other side, the data path between subchannel processor part A and the HFC-channel assigner transfers always one byte in transmit and receive direction during every 125 μ s cycle.

3.5.3.1 FIFO transmit operation in transparent mode

In transparent mode every FIFO has a data rate of 8 kByte/s. Every 125 μ s one byte of a FIFO is processed. The number of bits specified in `V_BIT_CNT` is placed at position `V_START_BIT + V_BIT_CNT - 1..V_START_BIT` while the other bits are not used and will be overwritten from the HFC-channel mask in part B of the subchannel processor.

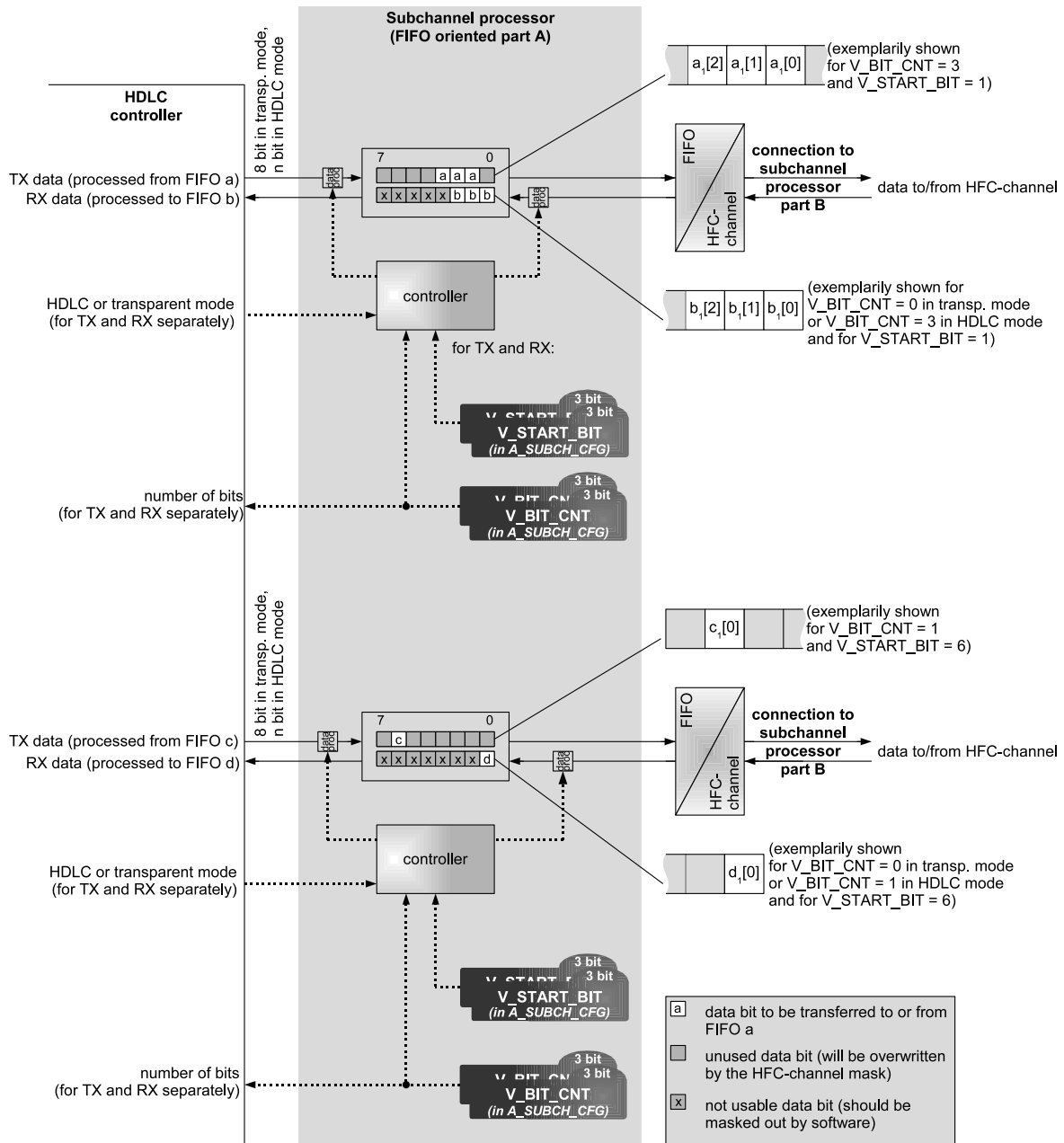


Figure 3.15: Part A of the subchannel processor

3.5.3.2 FIFO transmit operation in HDLC mode

The HDLC mode allows to reduce the data rate of a FIFO. With every 125 μ s cycle the subchannel processor requests V_BIT_CNT bits from the HDLC controller. The FIFO data rate is

$$DR_{FIFO} = \begin{cases} V_BIT_CNT \text{ kBit/s} & : V_BIT_CNT > 0 \\ 8 \text{ kBit/s} & : V_BIT_CNT = 0 \end{cases}$$

or might be a little lower due to the bit stuffing (zero insertion).

3.5.3.3 FIFO receive operation in transparent mode

The subchannel processor part A receives one byte every 125 μ s cycle. Typically, only some bits – depending on the usage mode of this receive channel – contain valid data. V_START_BIT defines the position of the valid bit field in the received HFC-channel byte. The subchannel processor part A shifts the valid bit field to position 0 before a whole byte is transferred to the HDLC controller. The invalid bits must be masked out by software. The FIFO data rate is always 8 kByte/s in this configuration.

If transparent mode is selected, V_BIT_CNT must always be '000' in receive direction. The number of valid bits must be handled by the software.

3.5.3.4 FIFO receive operation in HDLC mode

From every received HFC-channel data byte only V_BIT_CNT bits beginning at position V_START_BIT contain valid data. Only these bits are transferred to the HDLC controller. So the FIFO data rate is

$$DR_{FIFO} = \begin{cases} V_BIT_CNT \text{ kBit/s} & : V_BIT_CNT > 0 \\ 8 \text{ kBit/s} & : V_BIT_CNT = 0 \end{cases}$$

or might be a little lower due to the bit stuffing (zero deletion).

3.5.4 Details of the HFC-channel oriented part of the subchannel processor (part B)

Part B of the subchannel processor is located inside the HFC-channel area. With every 125 μ s cycle it transmits and receives always one data byte to/from the connected interface (either PCM or ST/U_p interface). On the other side, to/from every connected HFC-channel assigner one byte is transferred in both transmit and receive directions. Figure 3.16 shows the block diagram of this module.

3.5.4.1 FIFO transmit operation in SM

As the FIFO and HFC-channel numbers are the same in *Simple Mode*, only one FIFO can be connected to a HFC-channel. Subchannel processing can do nothing more than masking out some bits of every transmitted data byte.

The specified bit field is put into the HFC-channel mask byte before the data byte is transmitted to the connected interface.

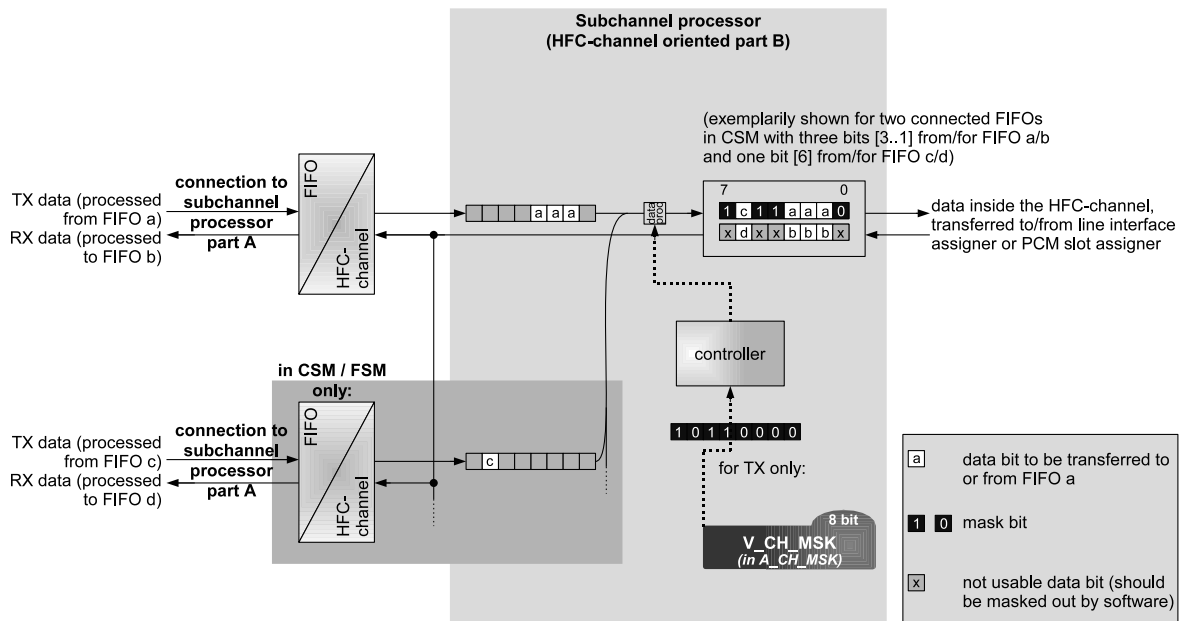


Figure 3.16: Part B of the subchannel processor

3.5.4.2 FIFO transmit operation in CSM and FSM

In *Channel Select Mode* and *FIFO Sequence Mode*, several FIFOs can contribute data to one HFC-channel data byte. From every connected HFC-channel assigner, one or more bits are extracted and are joined to a single HFC-channel data byte.

Here, the subchannel processor works in the same way as in *Simple Mode*, except that multiple bit insertion is performed. All FIFOs which contribute data bits to the HFC-channel byte should specify different bit locations to avoid overwriting data.

3.5.4.3 FIFO receive operation in SM

The received data byte is transferred to the HFC-channel assigner without modification. Part B of the subchannel processor has no effect to the receive data. Typically, only some bits contain valid data which will be extracted by the part A of the subchannel processor.

3.5.4.4 FIFO receive operation in CSM and FSM

If there are several FIFOs connected to one receive HFC-channel in *Channel Select Mode* or *FIFO Sequence Mode*, every received data byte is transferred to all connected HFC-channel assigners without modification. Part B of the subchannel processor has no effect to the receive data. Typically, the HFC-channel data byte contains bit fields for several FIFOs which will be extracted by their part A of the subchannel processor.

3.5.5 Subchannel example for SM

The subchannel processing example in Figure 3.17 shows two bidirectional configurations (❶ FIFO-to-ST/U_p and ❷ FIFO-to-PCM) in *Simple Mode*.

Please note !

All subchannel examples in this document have always the same number of bits and the same start bit for corresponding transmit and receive FIFOs. Actually, transmit and receive configuration settings are independently from each other. The settings are chosen for clearness and can simply be reproduced with looped data paths.

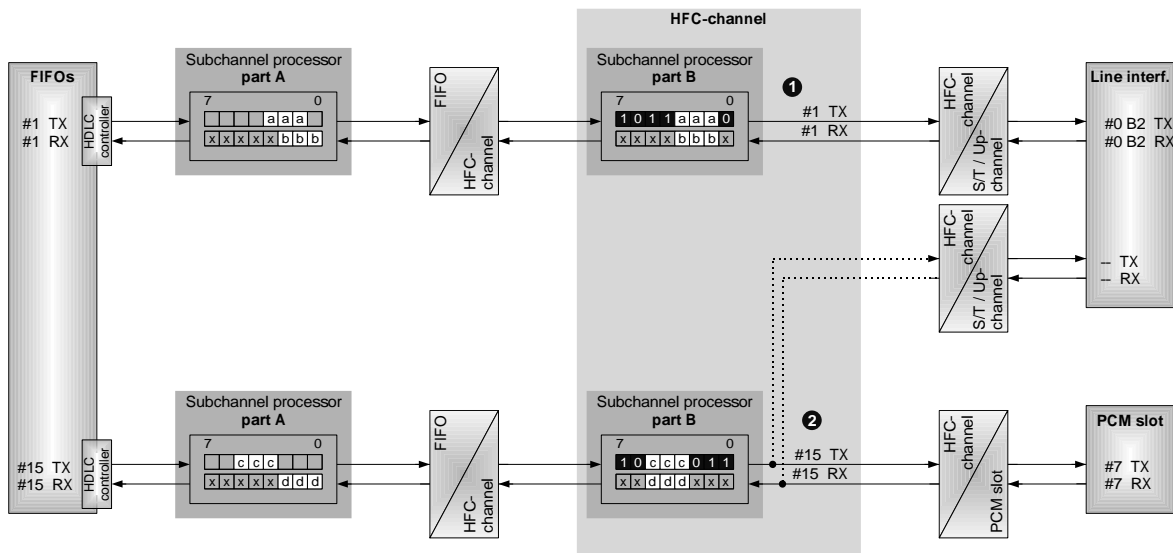


Figure 3.17: SM example with subchannel processor

❶ FIFO-to-ST/U_p (TX)

The first setting shows a FIFO-to-ST/U_p data transmission in transparent mode.

Register A_SUBCH_CFG[FIFO] defines three bits [2..0] to be transmitted from each FIFO byte. These bits have the position [3..1] in the HFC-channel data byte.

All other data bits in the HFC-channel byte are defined by the HFC-channel mask V_CH_MSK = '1011 0000' in register A_CH_MSK. This array register must be selected by writing the HFC-channel number and direction into register R_FIFO. The mask bits [3..1] are *don't care* because they are overwritten from the FIFO data.

A detailed overview of the transmitted data is shown in Table 3.6. The first data byte in FIFO[1,TX] is a_1 , the second byte is a_2 , and so on. In transparent mode only ($a_1[2..0], a_2[2..0], \dots$) are placed in the HFC-channel bytes at the location [3..1] and ($a_1[7..3], a_2[7..3], \dots$) are ignored and replaced by the HCF-channel mask.

Register setup:		(SM 1 TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 1	(FIFO #1)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[1, TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)
A_SUBCH_CFG[1, TX]	: V_BIT_CNT = 3	(process 3 bits)
	: V_START_BIT = 1	(start with bit 1)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit HFC-channel)
	: V_FIFO_NUM = 1	(HFC-channel #1)
	: V_REV = 0	(normal bit order)
A_CH_MSK[1, TX]	: V_CH_MSK = '1011 0000'	(mask byte)

1 FIFO-to-ST/U_p (RX)

Only three bits [3..1] from the received HFC-channel byte are assumed to be valid data. Nevertheless, the number of received bits must be set to the value $V_BIT_CNT = 0$ which means 'one byte'. The start position is specified with $V_START_BIT = 1$ in register A_SUBCH_CFG . As the received bit field is aligned to position 0, these bits represent FIFO data $b[2..0]$.

A detailed overview of the received data is shown in Table 3.7. The first data byte in $FIFO[1, RX]$ is b_1 , the second byte is b_2 , and so on. Only $(b_1[2..0], b_2[2..0], \dots)$ contain valid data and $(b_1[7..3], b_2[7..3], \dots)$ must be masked out by software.

Register setup:		(SM 1 RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 1	(FIFO #1)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[1, RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)
A_SUBCH_CFG[1, RX]	: V_BIT_CNT = 0	(process 8 bits)
	: V_START_BIT = 1	(start with bit 1)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)

Table 3.6: Subchannel processing according to Figure 3.17 (SM ① TX, transparent mode)

	7							0
HFC-channel mask:	1	0	1	1	0	0	0	0
HFC-channel transmit byte 1:	1	0	1	1	$a_1[2]$	$a_1[1]$	$a_1[0]$	0
HFC-channel transmit byte 2:	1	0	1	1	$a_2[2]$	$a_2[1]$	$a_2[0]$	0
HFC-channel transmit byte 3:	1	0	1	1	$a_3[2]$	$a_3[1]$	$a_3[0]$	0
...								

Table 3.7: Subchannel processing according to Figure 3.17 (SM ① RX, transparent mode)

	7							0
HFC-channel receive byte 1:	x	x	x	x	$b_1[2]$	$b_1[1]$	$b_1[0]$	x
HFC-channel receive byte 2:	x	x	x	x	$b_2[2]$	$b_2[1]$	$b_2[0]$	x
HFC-channel receive byte 3:	x	x	x	x	$b_3[2]$	$b_3[1]$	$b_3[0]$	x
...								

② FIFO-to-PCM (TX)

The second *Simple Mode* configuration connects a FIFO in HDLC mode with the PCM interface⁷. Bitmap `V_BIT_CNT` in register `A_SUBCH_CFG[FIFO]` defines three FIFO data bits to be transmitted during every 125 μ s cycle. The bit field location in the HFC-channel data byte is specified by bitmap `V_START_BIT` in the same register.

All other data bits in the HFC-channel are defined by the HFC-channel mask in register `A_CH_MSK`. This array register must be selected by writing the HFC-channel number and direction into register `R_FIFO`. The mask bits [5..3] are *don't care* because they are overwritten from the FIFO data.

A detailed overview of the transmitted data is shown in Table 3.8. The first data byte in `FIFO[15,TX]` is c_1 , the second byte is c_2 , and so on. In HDLC mode, FIFO bytes are dispersed among several HFC-channel bytes.

⁷HDLC bit stuffing is not shown in this example.

Register setup:		(SM ② TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 15	(FIFO #15)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[15,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO → ST/U _p , FIFO → PCM)
A_SUBCH_CFG[15,TX]	: V_BIT_CNT = 3	(process 3 bits)
	: V_START_BIT = 3	(start with bit 3)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit HFC-channel)
	: V_FIFO_NUM = 15	(HFC-channel #15)
	: V_REV = 0	(normal bit order)
A_CH_MSK[15,TX]	: V_CH_MSK = '1011 0011'	(mask byte)
R_SLOT	: V_SL_DIR = 0	(transmit slot)
	: V_SL_NUM = 7	(slot #7)
A_SL_CFG[7,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)
	: V_CH_SNUM = 15	(HFC-channel #15)
	: V_ROUT = '10'	(data to pin STIO1)

② FIFO-to-PCM (RX)

Only three bits [5..3] from the received HFC-channel byte are assumed to be valid data. This is done with bitmaps $V_BIT_CNT = 3$ and $V_START_BIT = 3$ in register A_SUBCH_CFG . The bit field is aligned to position 0 and transferred to the HDLC controller. There, FIFO data bytes are constructed from several received bit fields.

A detailed overview of the received data is shown in Table 3.9. The first data byte in $FIFO[15,RX]$ is d_1 , the second byte is d_2 , and so on. In HDLC mode, FIFO bytes are constructed from several HFC-channel bytes.

Register setup:		(SM 2 RX)
R_FIFO	: V_FIFO_DIR	= 1 (receive FIFO)
	: V_FIFO_NUM	= 15 (FIFO #15)
	: V_REV	= 0 (normal bit order)
A_CON_HDLC[15,RX]	: V_IFF	= 0 (0x7E as inter frame fill)
	: V_HDLC_TRP	= 0 (HDLC mode)
	: V_FIFO_IRQ	= 7 (enable FIFO)
	: V_DATA_FLOW	= '001' (FIFO ← PCM)
A_SUBCH_CFG[15,RX]	: V_BIT_CNT	= 3 (process 3 bits)
	: V_START_BIT	= 3 (start with bit 3)
	: V_LOOP_FIFO	= 0 (normal operation)
	: V_INV_DATA	= 0 (normal data transmission)
R_SLOT	: V_SL_DIR	= 1 (receive slot)
	: V_SL_NUM	= 7 (slot #7)
A_SL_CFG[7,RX]	: V_CH_SDIR	= 1 (receive HFC-channel)
	: V_CH_SNUM	= 15 (HFC-channel #15)
	: V_ROUT	= '10' (data from pin STIO2)

Table 3.8: Subchannel processing according to Figure 3.17 (SM 2 TX, HDLC mode)

	7	6	5	4	3	2	1	0
HFC-channel mask:	1	0	0	0	0	0	1	1
HFC-channel transmit byte 1:	1	0	c ₁ [2]	c ₁ [1]	c ₁ [0]	0	1	1
HFC-channel transmit byte 2:	1	0	c ₁ [5]	c ₁ [4]	c ₁ [3]	0	1	1
HFC-channel transmit byte 3:	1	0	c ₂ [0]	c ₁ [7]	c ₁ [6]	0	1	1
HFC-channel transmit byte 4:	1	0	c ₂ [3]	c ₂ [2]	c ₂ [1]	0	1	1
...								

Table 3.9: Subchannel processing according to Figure 3.17 (SM 2 RX, HDLC mode)

	7	6	5	4	3	2	1	0
HFC-channel receive byte 1:	x	x	d ₁ [2]	d ₁ [1]	d ₁ [0]	x	x	x
HFC-channel receive byte 2:	x	x	d ₁ [5]	d ₁ [4]	d ₁ [3]	x	x	x
HFC-channel receive byte 3:	x	x	d ₂ [0]	d ₁ [7]	d ₁ [6]	x	x	x
HFC-channel receive byte 4:	x	x	d ₂ [3]	d ₂ [2]	d ₂ [1]	x	x	x
...								

3.5.6 Subchannel example for CSM

In *Channel Select Mode* up to 8 FIFOs can be assigned to one HFC-channel if only 1 bit is processed by every FIFO. The example in Figure 3.18 shows two bidirectional configurations (❶ FIFO-to-ST/U_p and ❷ FIFO-to-PCM) with two FIFOs per direction each.

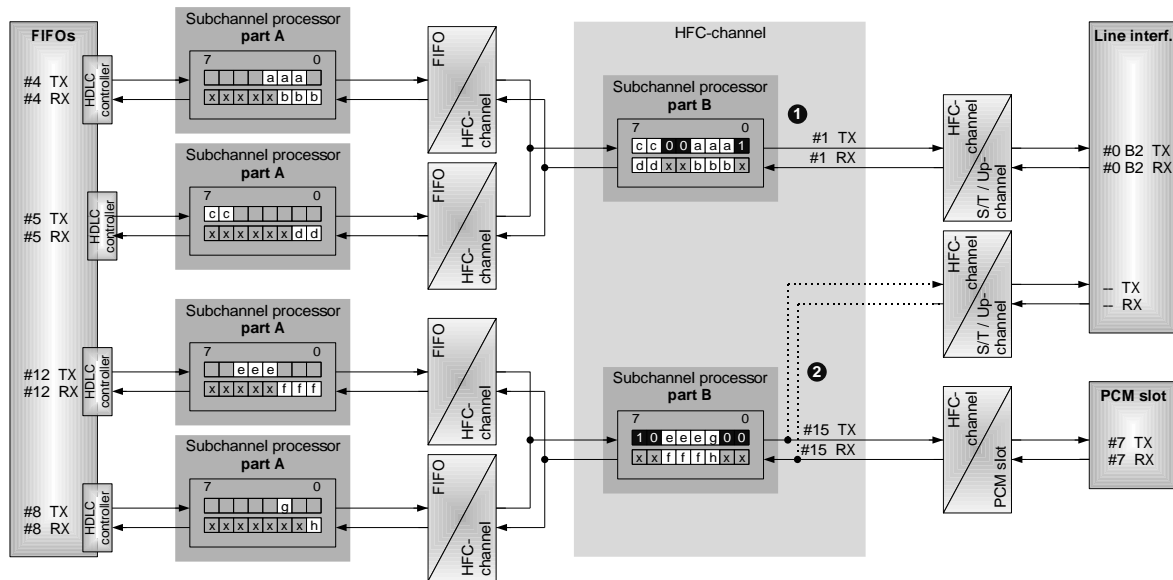


Figure 3.18: CSM example with subchannel processor

❶ FIFO-to-ST/U_p (TX)

In the first setting two transmit FIFOs are connected to one HFC-channel. Transparent mode is selected in this example.

Registers A_SUBCH_CFG[FIFO] of FIFO[4,TX] and FIFO[5,TX] define both, the number of bits to be extracted from the FIFO data bytes and their position in the HFC-channel byte.

The HFC-channel mask in register A_CH_MSK defines the bit values that are not used for FIFO data. The array register must be selected by writing the HFC-channel number and direction into register R_FIFO. The mask bits [7..6, 3..1] are *don't care* because they are overwritten from the FIFO data.

A detailed overview of the transmitted data is shown in Table 3.10. The first data byte in FIFO[4,TX] is a_1 , the second byte is a_2 , and so on. FIFO[5,TX] is represented by the data bytes c_1 , c_2 , and so on.

Register setup:		(CSM ① TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 4	(FIFO #4)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[4,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)
A_CHANNEL[4,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 1	(HFC-channel #1)
A_SUBCH_CFG[4,TX]	: V_BIT_CNT = 3	(process 3 bits)
	: V_START_BIT = 1	(start with bit 1)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 5	(FIFO #5)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[5,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO → ST/U _p , FIFO → PCM)
A_CHANNEL[5,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 1	(HFC-channel #1)
A_SUBCH_CFG[5,TX]	: V_BIT_CNT = 2	(process 2 bits)
	: V_START_BIT = 6	(start with bit 6)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit HFC-channel)
	: V_FIFO_NUM = 1	(HFC-channel #1)
	: V_REV = 0	(normal bit order)
A_CH_MSK[0,TX]	: V_CH_MSK = '0000 0001'	(mask byte)

① FIFO-to-ST/U_p (RX)

The received HFC-channel byte is distributed to two FIFOs. The bit fields [7..6] and [3..1] from the received HFC-channel byte are assumed to be valid data. Nevertheless, the number of received bits must be set to the value $V_BIT_CNT = 0$ which means 'one byte'. The start position is specified with V_START_BIT in register A_SUBCH_CFG . As the received bit fields are aligned to position 0, these bits represent FIFO data $b[2..0]$ and $d[1..0]$.

A detailed overview of the received data is shown in Table 3.11. The first data byte in FIFO[4,RX] is b_1 , the second byte is b_2 , and so on. FIFO[5,RX] data bytes are d_1 , d_2 , and so on.

Register setup:		(CSM 1 RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 4	(FIFO #4)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[4,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)
A_CHANNEL[4,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 1	(HFC-channel #1)
A_SUBCH_CFG[4,RX]	: V_BIT_CNT = 0	(process 8 bits)
	: V_START_BIT = 1	(start with bit 1)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 5	(FIFO #5)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[5,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 1	(transparent mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '000'	(FIFO ← ST/U _p)
A_CHANNEL[5,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 1	(HFC-channel #1)
A_SUBCH_CFG[5,RX]	: V_BIT_CNT = 0	(process 8 bits)
	: V_START_BIT = 6	(start with bit 6)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)

Table 3.10: Subchannel processing according to Figure 3.18 (CSM 1 TX, transparent mode)

	7	0
HFC-channel mask:	0	1
HFC-channel transmit byte 1:	c ₁ [1] c ₁ [0] 0 0 a ₁ [2] a ₁ [1] a ₁ [0] 1	
HFC-channel transmit byte 2:	c ₂ [1] c ₂ [0] 0 0 a ₂ [2] a ₂ [1] a ₂ [0] 1	
HFC-channel transmit byte 3:	c ₃ [1] c ₃ [0] 0 0 a ₃ [2] a ₃ [1] a ₃ [0] 1	
...		...

Table 3.11: Subchannel processing according to Figure 3.18 (CSM ① RX, transparent mode)

	7							0
HFC-channel transmit byte 1:	$d_1[1]$	$d_1[0]$	x	x	$b_1[2]$	$b_1[1]$	$b_1[0]$	x
HFC-channel transmit byte 2:	$d_2[1]$	$d_2[0]$	x	x	$b_2[2]$	$b_2[1]$	$b_2[0]$	x
HFC-channel transmit byte 3:	$d_3[1]$	$d_3[0]$	x	x	$b_3[2]$	$b_3[1]$	$b_3[0]$	x
...	...							

② FIFO-to-PCM (TX)

A FIFO-to-PCM configuration in HDLC mode with two FIFOs in transmit and receive direction each is shown in the second example setting⁸.

Registers A_SUBCH_CFG[FIFO] of FIFO[12,TX] and FIFO[8,TX] define both, the numbers of FIFO data bits to be transmitted during every 125 μ s cycle and their position in the HFC-channel byte.

All other data bits in the HFC-channel are defined by the HFC-channel mask in register A_CH_MSK. This array register must be selected by writing the HFC-channel number and direction into register R_FIFO. The mask bits [5..2] are *don't care* because they are overwritten from the FIFO data.

A detailed overview of the transmitted data is shown in Table 3.12. The first data byte in FIFO[12,TX] is e_1 , the second byte is e_2 , and so on. FIFO[8,TX] transmits bytes g_1 , g_2 , and so on. In HDLC mode, FIFO bytes are dispersed among several HFC-channel bytes.

⁸HDLC bit stuffing is not shown in this example.

Register setup:		(CSM ② TX)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 12	(FIFO #12)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[12,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO → ST/U _p , FIFO → PCM)
A_CHANNEL[12,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 15	(HFC-channel #15)
A_SUBCH_CFG[12,TX]	: V_BIT_CNT = 3	(process 3 bits)
	: V_START_BIT = 3	(start with bit 3)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit FIFO)
	: V_FIFO_NUM = 8	(FIFO #8)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[8,TX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO → ST/U _p , FIFO → PCM)
A_CHANNEL[8,TX]	: V_CH_FDIR = 0	(transmit HFC-channel)
	: V_CH_FNUM = 15	(HFC-channel #15)
A_SUBCH_CFG[8,TX]	: V_BIT_CNT = 1	(process 1 bit)
	: V_START_BIT = 2	(start with bit 2)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 0	(transmit HFC-channel)
	: V_FIFO_NUM = 15	(HFC-channel #15)
	: V_REV = 0	(normal bit order)
A_CH_MSK[15,TX]	: V_CH_MSK = '1000 1100'	(mask byte)
R_SLOT	: V_SL_DIR = 0	(transmit slot)
	: V_SL_NUM = 7	(slot #7)
A_SL_CFG[7,TX]	: V_CH_SDIR = 0	(transmit HFC-channel)
	: V_CH_SNUM = 15	(HFC-channel #15)
	: V_ROUT = '10'	(data to pin STIO1)

② FIFO-to-PCM (RX)

HFC-channel[15,RX] receives data bits that are to be distributed to FIFO[12,RX] and FIFO[8,RX].

Registers A_SUBCH_CFG[FIFO] of FIFO[12,RX] and FIFO[8,RX] define the numbers of valid data bits and their positions in the HFC-channel byte. These bits are dispersed to FIFO[12,RX] and FIFO[8,RX] where they are aligned to bit 0.

A detailed overview of the received data is shown in Table 3.13. The first data byte in FIFO[12,RX] is f_1 , the second byte is f_2 , and so on. FIFO[8,RX] receives bytes h_1 , h_2 , and so on. In HDLC mode, FIFO bytes are collected from several HFC-channel bytes.

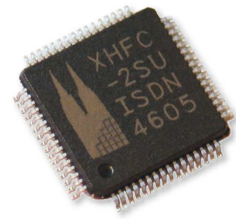
Register setup:		(CSM 2 RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 12	(FIFO #12)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[12,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO ← PCM)
A_CHANNEL[12,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 15	(HFC-channel #15)
A_SUBCH_CFG[12,TX]	: V_BIT_CNT = 3	(process 3 bits)
	: V_START_BIT = 3	(start with bit 3)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 8	(FIFO #8)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[8,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO ← PCM)
A_CHANNEL[8,RX]	: V_CH_FDIR = 1	(receive HFC-channel)
	: V_CH_FNUM = 15	(HFC-channel #15)
A_SUBCH_CFG[8,TX]	: V_BIT_CNT = 1	(process 1 bit)
	: V_START_BIT = 2	(start with bit 2)
	: V_LOOP_FIFO = 0	(normal operation)
	: V_INV_DATA = 0	(normal data transmission)
R_SLOT	: V_SL_DIR = 1	(receive slot)
	: V_SL_NUM = 7	(slot #7)
A_SL_CFG[7,RX]	: V_CH_SDIR = 1	(receive HFC-channel)
	: V_CH_SNUM = 15	(HFC-channel #15)
	: V_ROUT = '10'	(data from pin STIO2)

Table 3.12: Subchannel processing according to Figure 3.18 (CSM 2 TX, HDLC mode)

	7							0
HFC-channel mask:	1	0	0	0	1	1	0	0
HFC-channel transmit byte 1:	1	0	$e_1[2]$	$e_1[1]$	$e_1[0]$	$g_1[0]$	0	0
HFC-channel transmit byte 2:	1	0	$e_1[5]$	$e_1[4]$	$e_1[3]$	$g_1[1]$	0	0
HFC-channel transmit byte 3:	1	0	$e_2[0]$	$e_1[7]$	$e_1[6]$	$g_1[2]$	0	0
HFC-channel transmit byte 4:	1	0	$e_2[3]$	$e_2[2]$	$e_2[1]$	$g_1[3]$	0	0
...								

Table 3.13: Subchannel processing according to Figure 3.18 (CSM 2 RX, HDLC mode)

	7							0
HFC-channel transmit byte 1:	x	x	$f_1[2]$	$f_1[1]$	$f_1[0]$	$h_1[0]$	x	x
HFC-channel transmit byte 2:	x	x	$f_1[5]$	$f_1[4]$	$f_1[3]$	$h_1[1]$	x	x
HFC-channel transmit byte 3:	x	x	$f_2[0]$	$f_1[7]$	$f_1[6]$	$h_1[2]$	x	x
HFC-channel transmit byte 4:	x	x	$f_2[3]$	$f_2[2]$	$f_2[1]$	$h_1[3]$	x	x
...								



Chapter 4

FIFO handling and HDLC controller

Table 4.1: Overview of the XHFC-2SU FIFO registers

Write only registers:			Read / write registers:		
Address	Name	Page	Address	Name	Page
0x0B	R_FIRST_FIFO	133	0x80	A_FIFO_DATA	145
0x0C	R_FIFO_THRES	134	0x84	A_FIFO_DATA_NOINC	145
0x0D	R_FIFO_MD	135	0xF4	A_CH_MSK	146
0x0E	A_INC_RES_FIFO	136	0xFA	A_CON_HDLC	147
0x0F	R_FIFO	137	0xFB	A_SUBCH_CFG	149
0x0F	R_FSM_IDX	137	0xFC	A_CHANNEL	150
Read only register:			0xFD	A_FIFO_SEQ	151
Address	Name	Page	0xFF	A_FIFO_CTRL	152
0x04	A_Z1	138			
0x06	A_Z2	138			
0x0C	A_F1	138			
0x0D	A_F2	139			
0x0E	A_FIFO_STA	140			
0x14	A_USAGE	141			
0x24	R_FILL_BL0	141			
0x25	R_FILL_BL1	142			
0x26	R_FILL_BL2	143			
0x27	R_FILL_BL3	144			

4.1 Overview

There are up to 16 receive FIFOs and up to 16 transmit FIFOs with 32 HDLC controllers in whole. The HDLC controllers are located on the ST/U_p interface side of the FIFOs. Thus plain data is always stored in the FIFOs. Automatic zero insertion is done in HDLC mode when HDLC data goes from the FIFOs to the ST/U_p interface or to the PCM bus (transmit FIFO operation). Automatic zero deletion is done in HDLC mode when the HDLC data comes from the ST/U_p interface or PCM bus (receive FIFO operation).

There are more transmit and receive FIFOs than B-, D- and E-channels. These can be used for PCM-to-FIFO and PCM-to-ST/U_p configurations.

The FIFO control registers are used to select and control the FIFOs of XHFC-2SU. The FIFO register set exists for every FIFO number and receive/transmit direction. A FIFO is selected by the FIFO select register R_FIFO.

All FIFOs are disabled after reset (hardware reset, global software reset or HFC reset). With register A_CON_HDLC the selected FIFO is enabled by setting V_FIFO_IRQ to a value different from zero.

4.2 FIFO counters

The FIFOs are realized as ring buffers in the internal SRAM. They are controlled by counters. The counter sizes depend on the setting of the FIFO sizes. *Z1* is the FIFO input counter and *Z2* is the FIFO output counter.

Each counter points to a byte position in the SRAM. On a FIFO input operation *Z1* is incremented. On an output operation *Z2* is incremented. If $Z1 = Z2$ and $F1 = F2$ the FIFO is empty.

After every pulse on the F0IO signal, HDLC bytes are written into the ST/U_p interface (from a transmit FIFO) and HDLC bytes are read from the ST/U_p interface (to a receive FIFO). A connection to the PCM interface is also possible.

The D-channel data is processed in exactly the same way as the B-channel data, except that the D-FIFO data rate is reduced in HDLC mode.

Additionally there are two 4 bit counters *F1* and *F2* for every FIFO. They count the HDLC frames and form a ring buffer as *Z1* and *Z2* do, too.

F1 is incremented when a complete frame has been received and stored in the FIFO. *F2* is incremented when a complete frame has been read from the FIFO. If $F1 = F2$ there is no complete frame in the FIFO.

The reset state of the *Z*- and *F*-counters are

- $Z1 = Z2 = Z_{MAX}^1$ and
- $F1 = F2 = F_{MAX} = 0x07$.

This initialization can be carried out with a global software reset or a HDLC reset. For this, bits V_SRES or V_HFC_RES in register R_CIRM have to be set. Individual FIFOs can be reset with bit V_RES_FIFO in register A_INC_RES_FIFO.

¹See Z_{max} value in Table 4.2.

In addition, a hardware reset initializes the counters.

 **Please note !**

Abort D-channel transmission

A FIFO reset should never be initiated while an HDLC ending flag is just in transmission. If this cannot be ensured – or to be on the safe side – any FIFO reset can be wrapped in a D-channel reset as shown below.

Additionally, V_RES_FIFO_ERR should always be set together with a FIFO reset even if no FIFO error is pending. This can be done with a single register write access.

Register setup:

R_FIFO	: V_FIFO_DIR	= <i>dir</i>	Select FIFO data direction *
	: V_FIFO_NUM	= <i>n</i>	Select FIFO number
repeat until			
R_STATUS	: V_BUSY	= 0	Wait until not busy
R_SU_SEL	: V_SU_SEL	= <i>m</i>	Select interface number
A_SU_CTRL1[ST/Up]	: V_D_RES	= 1	D-channel reset
A_INC_RES_FIFO[FIFO]	: V_RES_FIFO_ERR	= 1	Reset FIFO error *
	: V_RES_FIFO	= 1	Reset selected FIFO
repeat until			
R_STATUS	: V_BUSY	= 0	Wait until not busy
A_SU_CTRL1[ST/Up]	: V_D_RES	= 0	Normal D-channel operation

* Single register write access together with the following command.

 **Important !**

Busy status after FIFO change, FIFO reset and F1/F2 incrementation

Changing a FIFO, resetting a FIFO or incrementing the *F*-counters causes a short BUSY period of XHFC-2SU. This means an access to FIFO control registers is not allowed until BUSY status is cleared (bit V_BUSY in register R_STATUS). The maximum duration takes 25 clock cycles ($\sim 1 \mu\text{s}$). Status, interrupt and control registers can be read and written at any time.

 **Please note !**

The counter state Z_{MIN} (resp. F_{MIN}) of the Z-counters (resp. *F*-counters) follows counter state Z_{MAX} (resp. F_{MAX}) in the FIFOs.

Please note that Z_{MIN} and Z_{MAX} depend on the FIFO number and FIFO size (s. Section 4.3 and Table 4.2).

4.3 FIFO size setup

Table 4.2 shows how the FIFO size can be varied. Additionally, the initial Z_{\max} and Z_{\min} values are given in Table 4.2. A global software reset should be initiated after changing the FIFO size.

Normal operation with transmit and receive FIFOs is configured with $V_UNIDIR_MD = '0'$ in register R_FIFO_MD . Some applications may use only transmit or only receive FIFOs. In this case $V_UNIDIR_MD = '1'$ provides two settings as shown in Table 4.2. The data direction can be chosen either with $V_UNIDIR_RX = '0'$ (only transmit FIFOs available) or with $V_UNIDIR_RX = '1'$ (only receive FIFOs available). V_UNIDIR_RX is ignored if $V_UNIDIR_MD = '0'$.

Table 4.2: FIFO size setup *

V_FIFO_MD	V_UNIDIR_MD	FIFO numbers	Z_{MIN}	Z_{MAX}	FIFO size (byte)
'00'	'0'	0 .. 15	0x00	0x3F	64
'01'	'0'	0 .. 7	0x00	0x7F	128
'10'	'0'	0 .. 3	0x00	0xFF	256
'00'	'1'	0 .. 15	0x00	0x7F	128
'01'	'1'	0 .. 7	0x00	0xFF	256

*: Please note, that any configuration that uses *receive FIFOs beyond the maximum FIFO number* can store data in the RAM. This may destroy data from other FIFOs.

4.4 FIFO operation



Important !

The HDLC controller needs F0IO and either C4IO or C2IO clocks for operation. These clocks are generated from XHFC-2SU if PCM master mode is selected ($V_PCM_MD = '1'$).

In PCM slave mode ($V_PCM_MD = '0'$), F0IO and either C4IO or C2IO clocks must be feed into these pins (see Figure 6.5 on page 229, signal C4I is required).

4.4.1 HDLC transmit FIFOs

Data can be transmitted from the host bus interface to the FIFO with write access to registers A_FIFO_DATA and $A_FIFO_DATA_NOINC$. XHFC-2SU converts the data into HDLC code and transfers it from the FIFO to the ST/U_p or the PCM bus interface.

XHFC-2SU checks $Z1$ and $Z2$. If $Z1 = Z2$ (FIFO empty), XHFC-2SU generates a HDLC flag ('0111 1110') or continuous '1's (depending on bit V_IFF in register A_CON_HDLC) and transmits it to the ST/U_p interface. In this case $Z2$ is not incremented. If also $F1 = F2$ only HDLC flags

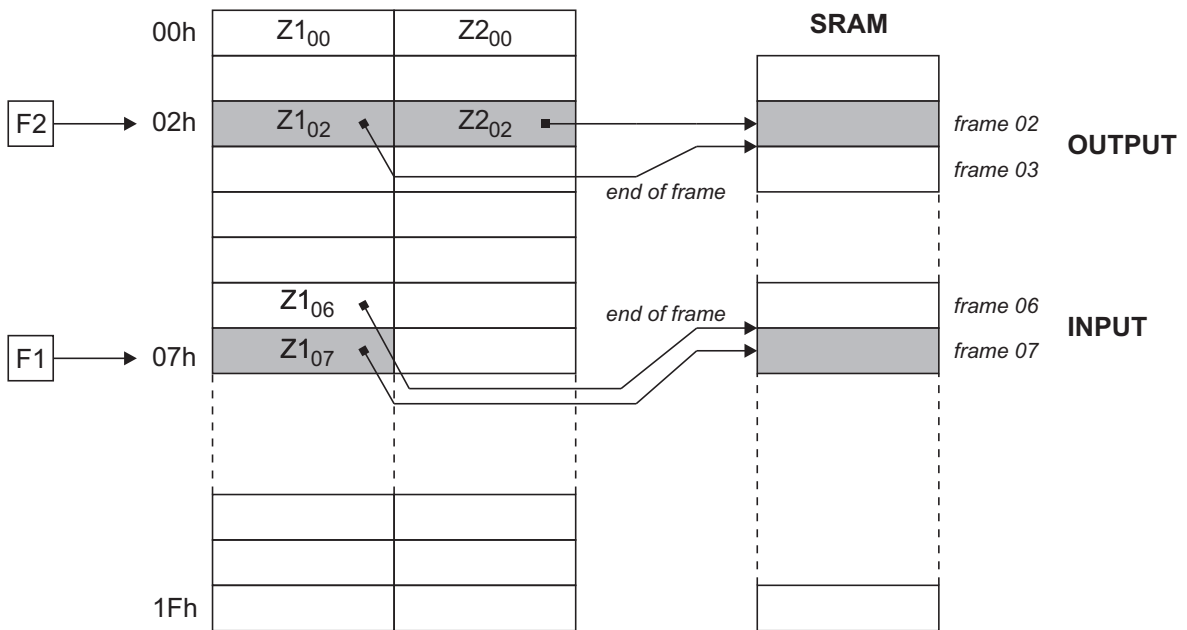


Figure 4.1: FIFO organization

or continuous '1's are sent to the ST/ U_p interface and all counters remain unchanged. If the frame counters are unequal $F2$ is incremented and XHFC-2SU tries to transmit the next frame. At the end of a frame ($Z2$ reaches $Z1$) it automatically generates the 16 bit CRC checksum and adds an ending flag. If there is another frame in the FIFO ($F1 \neq F2$) the $F2$ counter is incremented again.


With every byte being written from the host bus side to the FIFO, $Z1$ is incremented automatically. If a complete frame has been sent into the FIFO $F1$ must be incremented to transmit the next frame. If the frame counter $F1$ is incremented the Z -counters may also change because $Z1$ and $Z2$ are functions of $F1$ and $F2$. Thus there are $Z1(F1)$, $Z2(F1)$, $Z1(F2)$ and $Z2(F2)$ (see Fig. 4.1).

$Z1(F1)$ is used for the frame which is just written from the host bus side. $Z2(F2)$ is used for the frame which is just being transmitted to the PCM or ST/ U_p interface side of XHFC-2SU. $Z1(F2)$ is the end of frame pointer of the current output frame.

In the transmit HFC-channels $F1$ is only incremented from the host interface side if the software driver wants to say "end of transmit frame". This is done by setting bit V_INC_F in register $A_INC_RES_FIFO$. Then the current value of $Z1$ is stored, $F1$ is incremented and $Z1$ is used as start address of the next frame.

4.4.2 Automatic D-channel frame repetition (for S/T in TE mode only)

The D-channel transmit FIFO has a special feature. If the ST/ U_p interface signals a D-channel contention before the CRC is sent the $Z2$ counter is set to the starting address of the current frame and XHFC-2SU tries to repeat the frame automatically.

 **Please note !**

XHFC-2SU begins to transmit the bytes from a FIFO at the moment the FIFO is changed (writing R_FIFO) or the *F1* counter is incremented. Switching to the FIFO that is already selected also starts the transmission. Thus by selecting the same FIFO again transmission can be started.

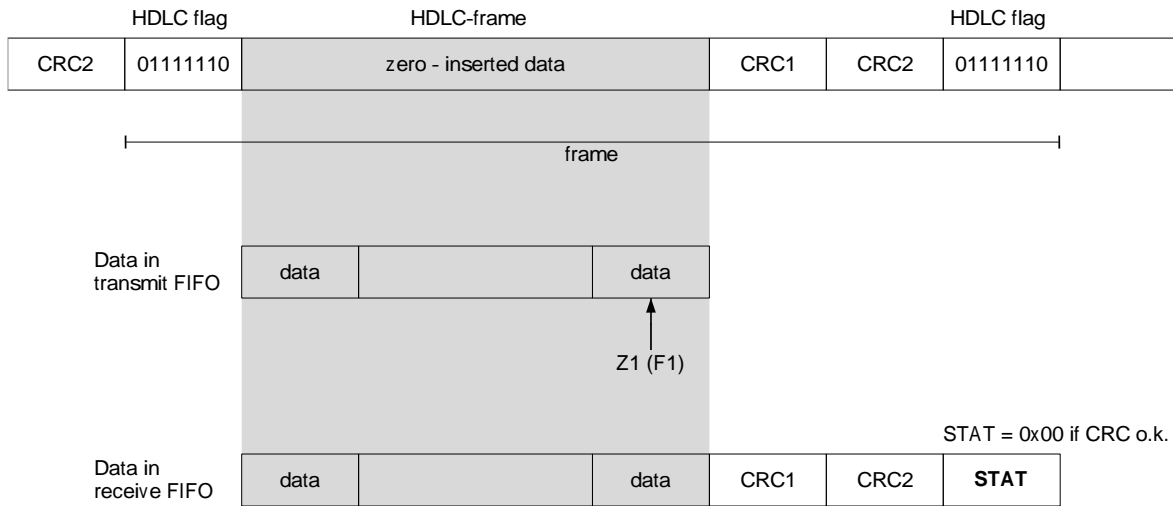


Figure 4.2: FIFO data organization in HDLC mode

4.4.3 HDLC receive FIFOs

The receive HFC-channels receive data from the ST/U_p or PCM bus interface read registers. The data is converted from HDLC into plain data and sent to the FIFO. The data can then be read via the host bus interface.

XHFC-2SU checks the HDLC data coming in. If it finds a flag or more than 5 consecutive '1's it does not generate any output data. In this case Z1 is not incremented. Proper HDLC data being received is converted by XHFC-2SU into plain data. After the ending flag of a frame, XHFC-2SU checks the HDLC CRC checksum. If it is correct one byte with all '0's is inserted behind the CRC data in the FIFO named STAT (see Fig. 4.2). This last byte of a frame in the FIFO is different from all '0's if there is no correct CRC field at the end of the frame.

If the STAT value is 0xFF, the HDLC frame ended with at least 8 bits '1's. This is similar to an abort HDLC frame condition.

The ending flag of a HDLC frame can also be the starting flag of the next frame.

After a frame is received completely, *F1* is incremented by XHFC-2SU automatically and the next frame can be received.

After reading a frame via the host bus interface *F2* has to be incremented. If the frame counter *F2* is incremented also the Z-counters may change because Z1 and Z2 are functions of *F1* and *F2*. Thus there are Z1(*F1*), Z2(*F1*), Z1(*F2*) and Z2(*F2*) (see Fig. 4.1).

Z1(*F1*) is used for the frame which is just received from the ST/U_p interface side of XHFC-2SU.

$Z2(F2)$ is used for the frame which is just being transmitted to the host bus interface. $Z1(F2)$ is the end of frame pointer of the current output frame.

To calculate the length of the current receive frame the software has to evaluate $Z1 - Z2 + 1$. When $Z2$ reaches $Z1$ the complete frame has been read.

In the receive HFC-channels $F2$ must be incremented from the host interface side after the software detects an end of receive frame ($Z1 = Z2$) and $F1 \neq F2$. Then the current value of $Z2$ is stored, $F2$ is incremented and $Z2$ is copied as start address of the next frame. This is done by setting bit V_INC_F in register $A_INC_RES_FIFO$. If $Z1 = Z2$ and $F1 = F2$ the FIFO is totally empty. $Z1(F1)$ can not be accessed.



Important !

Before reading a new frame, a change FIFO operation (write access to register R_FIFO) has to be done even if the desired FIFO is already selected. The change FIFO operation is required to update the internal buffer of XHFC-2SU. Otherwise the first byte of the FIFO will be taken from the internal buffer and may be invalid.

4.4.4 Transparent mode of XHFC-2SU

It is possible to switch off the HDLC operation for each FIFO independently by bit V_HDLC_TRP in register A_CON_HDLC . If this bit is set, data from the FIFO is sent directly to the ST/U_p or PCM bus interface and data from the ST/U_p or PCM bus interface is sent directly to the FIFO.

Be sure to switch into transparent mode only if $F1 = F2$. Being in transparent mode the F -counters remain unchanged. $Z1$ and $Z2$ are the input and output pointers respectively. Because $F1 = F2$, the Z -counters are always accessible and have valid data for FIFO input and output.

If a transmit FIFO changes to FIFO empty condition no CRC is generated and the last data byte written into the FIFO is repeated until there is new data.

Normally the last byte is undefined because of the Z -counter pointing to a previously unwritten address. To define the last byte, the last write access to the FIFO must be done without Z increment (see register $A_FIFO_DATA_NOINC$).

In receive HFC-channels there is no check on flags or correct CRCs and no status byte added.

Unlike in HDLC mode, where byte synchronization is achieved with HDLC flags, the byte boundaries are not arbitrary. The data is just the same as it comes from or is sent to the ST/U_p or PCM bus interface.

Transmit and receive transparent data can be done in two ways. The usual way is transporting FIFO data to the ST/U_p interface with the LSB first as usual in HDLC mode. The second way is transmitting the bytes in reverse bit order as usual for PCM data. So the first bit is the MSB. The bit order can be reversed by setting bit V_REV in register R_FIFO when the FIFO is selected.

**Important !**

For normal data transmission register `A_SUBCH_CFG` must be set to `0x00`. To use 56 kbit/s restricted mode for U.S. ISDN lines, register `A_SUBCH_CFG` must be set to `0x07` for B-channels.

4.5 Register description

4.5.1 Write only registers

R_FIRST_FIFO	(w)	(Reset group: H, 0, 1)	0x0B
First FIFO of the FIFO sequence			
This register is only used in <i>FIFO Sequence Mode</i> , see register R_FIFO_MD for data flow mode selection.			
Bits	Reset value	Name	Description
0	0	V_FIRST_FIFO_DIR	Data direction This bit defines the data direction of the first FIFO in FIFO sequence. '0' = transmit FIFO data '1' = receive FIFO data
4..1	0	V_FIRST_FIFO_NUM	FIFO number This bitmap defines the number of the first FIFO in the FIFO sequence.
7..5	0	(reserved)	Must be '000'.

R_FIFO_THRES	(w)	(Reset group: H, 0, 1, 2, 3)	0x0C
FIFO fill level control register			
The FIFO fill level can be controlled by a threshold which is specified separately for transmit and receive FIFOs.			
Bits	Reset value	Name	Description
3..0	1	V_THRES_TX	Threshold for all transmit FIFOs The threshold is a multiple of 16 bytes. 0 = 0 bytes 1 = 16 bytes 2 = 32 bytes ... 15 = 240 bytes
7..4	1	V_THRES_RX	Threshold for all receive FIFOs The threshold is a multiple of 16 bytes. 0 = 0 bytes 1 = 16 bytes 0 = 32 bytes ... 15 = 240 bytes

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_FIFO_MD	(w)	(Reset group: H)	0x0D
FIFO mode configuration			
This register defines the FIFO number and size. The actual FIFO size depends also on the V_UNIDIR_MD value.			
Bits	Reset value	Name	Description
1..0	0	V_FIFO_MD	<p>FIFO mode This bitmap and V_UNIDIR_MD are used to organize the FIFOs.</p> <p>FIFO mode with V_UNIDIR_RX = '0': '00' = 16 FIFOs with 64 bytes for TX and RX each '01' = 8 FIFOs with 128 bytes for TX and RX each '10' = 4 FIFOs with 256 bytes for TX and RX each '11' = not allowed</p> <p>FIFO mode with V_UNIDIR_RX = '1': '00' = 16 FIFOs with 128 bytes for TX and RX each '01' = 8 FIFOs with 256 bytes for TX and RX each '10' = not allowed '11' = not allowed</p>
3..2	0	V_DF_MD	<p>Data flow mode selection '00' = <i>Simple Mode (SM)</i> '01' = <i>Channel Select Mode (CSM)</i> '10' = not allowed '11' = <i>FIFO Sequence Mode (FSM)</i></p>
4	0	V_UNIDIR_MD	<p>Unidirectional FIFO data direction '0' = both transmit and receive FIFOs available (normal operation), V_UNIDIR_RX is ignored '1' = unidirectional FIFO mode, either transmit or receive FIFOs available due to V_UNIDIR_RX value Unidirectional FIFO data is used for voice recording, e.g., and has double FIFO size. Note: The FIFOs of the unused data direction must not be enabled.</p>
5	0	V_UNIDIR_RX	<p>FIFO data direction This bit is only used in unidirectional FIFO mode (V_UNIDIR_MD = '1'). '0' = only transmit FIFOs available '1' = only receive FIFOs available</p>
7..6	0	(reserved)	Must be '00'.

Bits	Reset value	Name	Description
Increment and reset FIFO register			
Before writing this array register the FIFO must be selected by register R_FIFO.			
0	0	V_INC_F	Increment the F-counters of the selected FIFO '0' = no increment '1' = increment This bit is automatically cleared after the counter increment has been processed.
1	0	V_RES_FIFO	FIFO reset '0' = no reset '1' = reset selected FIFO (F- and Z-counters and channel mask A_CH_MSK are reset) This bit is automatically cleared after the FIFO reset has been processed.
2	0	V_RES_LOST	LOST error bit reset '0' = no reset '1' = reset LOST This bit is automatically cleared with the LOST error bit reset.
3	0	V_RES_FIFO_ERR	FIFO error reset '0' = no operation '1' = Resets bit V_FIFO_ERR in register A_FIFO_STA This bit automatically reset to '0' after the FIFO error reset has been executed. Note: for transmit FIFOs, this bit should be set to '1' not before V_ABO_DONE = '1' in register A_FIFO_STA to ensure a completed frame abort.
7..4	0	(reserved)	Must be '0000'.

R_FIFO	(w)	(Reset group: H, 0, 1)	0x0F
FIFO selection register			
<p>This register is used to select a FIFO. Before a FIFO array register can be accessed, this index register must specify the desired FIFO number and data direction.</p> <p>Note: This register is a multi-register. It is selected with bitmap V_DF_MD less than '11' in register R_FIFO_MD (SM and CSM). In FSM (V_DF_MD = '11') some FIFO array registers are indexed by the multi-register R_FSM_IDX instead, but most FIFO array registers remain indexed by this register.</p>			
Bits	Reset value	Name	Description
0	0	V_FIFO_DIR	FIFO data direction '0' = transmit FIFO data '1' = receive FIFO data
4..1	0	V_FIFO_NUM	FIFO number
6..5	0	(reserved)	Must be '00'.
7	0	V_REV	Bit order '0' = LSB first '1' = MSB first LSB first is used in HDLC mode while MSB first is used in transparent mode. The bit order is being reversed for the data written into the FIFO or when the data is read from the FIFO.

R_FSM_IDX	(w)	(Reset group: H, 0, 1)	0x0F
Index register of the FIFO sequence			
<p>This register is used to select a list number in <i>FIFO Sequence Mode</i>. Some FIFO array registers are indexed by this list number. Before these registers can be accessed, this index register must specify the desired list number.</p> <p>Note: This register is a multi-register. It is selected with bitmap V_DF_MD = '11' in register R_FIFO_MD. In FSM only few FIFO array registers are indexed by this multi-register, but most FIFO array registers remain indexed by R_FIFO.</p>			
Bits	Reset value	Name	Description
4..0	0x00	V_IDX	List index The list index must be in the range 0..31.
7..5	0	(reserved)	Must be '000'.

4.5.2 Read only registers

A_Z1 [FIFO]	(r)	(Reset group: H, 0, 1)	0x04
FIFO input counter Z1			
Before reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		V_Z1	Counter value of Z1 The reset value is Z_{max} and depends on the FIFO configuration.

(See Table 4.2 for reset value.)

A_Z2 [FIFO]	(r)	(Reset group: H, 0, 1)	0x06
FIFO output counter Z2			
Before reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		V_Z2	Counter value of Z2 The reset value is Z_{max} and depends on the FIFO configuration.

(See Table 4.2 for reset value.)

A_F1 [FIFO]	(r)	(Reset group: H, 0, 1)	0x0C
FIFO input HDLC frame counter F1			
Before reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0	7	V_F1	Counter value Up to 7 HDLC frames can be stored in every FIFO.

A_F2 [FIFO]	(r)	(Reset group: H, 0, 1)	0x0D								
<p>FIFO output HDLC frame counter <i>F2</i></p> <p>Before reading this array register the FIFO must be selected by register R_FIFO.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Reset value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7..0</td> <td>7</td> <td>V_F2</td> <td>Counter value Up to 7 HDLC frames can be stored in every FIFO.</td> </tr> </tbody> </table>				Bits	Reset value	Name	Description	7..0	7	V_F2	Counter value Up to 7 HDLC frames can be stored in every FIFO.
Bits	Reset value	Name	Description								
7..0	7	V_F2	Counter value Up to 7 HDLC frames can be stored in every FIFO.								

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

Bits	Reset value	Name	Description
FIFO status register			
Before reading this array register the FIFO must be selected by register R_FIFO.			
0	0	V_FIFO_ERR	<p>FIFO error This status bit has different meaning for transmit and receive FIFOs.</p> <p>Transmit FIFO: There are two different situations for a FIFO to run empty. (1) A valid HDLC frame has been sent when the FIFO runs empty at the end of a frame and $F1 = F2$ gets true after $Z2$ increment. (2) An invalid HDLC frame has been sent when the FIFO runs empty within a frame, i.e. $F1 = F2$ is already valid during data transmission and $Z2$ increment cannot be executed.</p> <p>When $V_FR_ABO = '0'$ in register A_FIFO_CTRL, this bit is set to '1' to indicate an empty FIFO (either situation 1 or situation 2). Transmitted frames are valid in any case, but the frame got split in situation 2.</p> <p>When $V_FR_ABO = '1'$ in register A_FIFO_CTRL, this bit is only set to '1' to indicate an aborted frame (situation 2 only, invalid frame).</p> <p>As long as the FIFO is empty, interframe fill is repeatedly send in HDLC mode.</p> <p>Receive FIFO: This bit is set to '1' to indicate either a FIFO overflow ($Z1 = Z2$ after $Z1$ increment) or a frame counter overflow ($F1 = F2$ after $F1$ increment).</p> <p>Note: V_FIFO_ERR must be reset from host processor with $V_RES_FIFO_ERR = '1'$.</p>
3..1		(reserved)	
4	0	V_ABO_DONE	<p>Frame abort done This status bit is only used for transmit FIFOs and is not defined for receive FIFOs.</p> <p>This bit is set after sixteen consecutive '1's have been transmitted. It is reset together with V_FIFO_ERR.</p>
7..5		(reserved)	

A_USAGE [FIFO]		(r)	(Reset group: H, 0, 1)	0x14
FIFO fill level				
Before reading this array register the FIFO must be selected by register R_FIFO.				
Bits	Reset value	Name	Description	
7..0	0x00	V_USAGE	Number of bytes currently stored in the FIFO The FIFO is empty when this register has the value 0x00.	

R_FILL_BL0		(r)	(Reset group: H, 0, 1)	0x24
FIFO fill level for FIFO block 0				
When a bit is set to '1', more than the specified number of bytes is currently being in the FIFO. The threshold is separately defined for transmit and receive FIFOs, V_THRES_TX for transmit FIFOs and V_THRES_RX for receive FIFOs in register R_FIFO_THRES.				
Bits	Reset value	Name	Description	
0	0	V_FILL_FIFO0_TX	FIFO[0,TX] fill level	
1	0	V_FILL_FIFO0_RX	FIFO[0,RX] fill level	
2	0	V_FILL_FIFO1_TX	FIFO[1,TX] fill level	
3	0	V_FILL_FIFO1_RX	FIFO[1,RX] fill level	
4	0	V_FILL_FIFO2_TX	FIFO[2,TX] fill level	
5	0	V_FILL_FIFO2_RX	FIFO[2,RX] fill level	
6	0	V_FILL_FIFO3_TX	FIFO[3,TX] fill level	
7	0	V_FILL_FIFO3_RX	FIFO[3,RX] fill level	

R_FILL_BL1	(r)	(Reset group: H, 0, 1)	0x25
FIFO fill level for FIFO block 1			
<p>When a bit is set to '1', more than the specified number of bytes is currently being in the FIFO. The threshold is separately defined for transmit and receive FIFOs, V_THRES_TX for transmit FIFOs and V_THRES_RX for receive FIFOs in register R_FIFO_THRES.</p>			
Bits	Reset value	Name	Description
0	0	V_FILL_FIFO4_TX	FIFO[4, TX] fill level
1	0	V_FILL_FIFO4_RX	FIFO[4, RX] fill level
2	0	V_FILL_FIFO5_TX	FIFO[5, TX] fill level
3	0	V_FILL_FIFO5_RX	FIFO[5, RX] fill level
4	0	V_FILL_FIFO6_TX	FIFO[6, TX] fill level
5	0	V_FILL_FIFO6_RX	FIFO[6, RX] fill level
6	0	V_FILL_FIFO7_TX	FIFO[7, TX] fill level
7	0	V_FILL_FIFO7_RX	FIFO[7, RX] fill level

R_FILL_BL2	(r)	(Reset group: H, 0, 1)	0x26
FIFO fill level for FIFO block 2			
<p>When a bit is set to '1', more than the specified number of bytes is currently being in the FIFO. The threshold is separately defined for transmit and receive FIFOs, V_THRES_TX for transmit FIFOs and V_THRES_RX for receive FIFOs in register R_FIFO_THRES.</p>			
Bits	Reset value	Name	Description
0	0	V_FILL_FIFO8_TX	FIFO[8,TX] fill level
1	0	V_FILL_FIFO8_RX	FIFO[8,RX] fill level
2	0	V_FILL_FIFO9_TX	FIFO[9,TX] fill level
3	0	V_FILL_FIFO9_RX	FIFO[9,RX] fill level
4	0	V_FILL_FIFO10_TX	FIFO[10,TX] fill level
5	0	V_FILL_FIFO10_RX	FIFO[10,RX] fill level
6	0	V_FILL_FIFO11_TX	FIFO[11,TX] fill level
7	0	V_FILL_FIFO11_RX	FIFO[11,RX] fill level

R_FILL_BL3	(r)	(Reset group: H, 0, 1)	0x27
FIFO fill level for FIFO block 3			
<p>When a bit is set to '1', more than the specified number of bytes is currently being in the FIFO. The threshold is separately defined for transmit and receive FIFOs, V_THRES_TX for transmit FIFOs and V_THRES_RX for receive FIFOs in register R_FIFO_THRES.</p>			
Bits	Reset value	Name	Description
0	0	V_FILL_FIFO12_TX	FIFO[12,TX] fill level
1	0	V_FILL_FIFO12_RX	FIFO[12,RX] fill level
2	0	V_FILL_FIFO13_TX	FIFO[13,TX] fill level
3	0	V_FILL_FIFO13_RX	FIFO[13,RX] fill level
4	0	V_FILL_FIFO14_TX	FIFO[14,TX] fill level
5	0	V_FILL_FIFO14_RX	FIFO[14,RX] fill level
6	0	V_FILL_FIFO15_TX	FIFO[15,TX] fill level
7	0	V_FILL_FIFO15_RX	FIFO[15,RX] fill level

4.5.3 Read / write registers

A_FIFO_DATA [FIFO]	(r/w)	(Reset group: -)	0x80
FIFO data register			
Before writing or reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		V_FIFO_DATA	Data byte Read / write one byte from / to the FIFO selected with register R_FIFO and increment Z-counter by 1.

A_FIFO_DATA_NOINC [FIFO]	(r/w)	(Reset group: -)	0x84
FIFO data register			
Before writing or reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		V_FIFO_DATA_NOINC	Data byte Write one byte to or read one byte from the FIFO selected with register R_FIFO without incrementing Z-counter.

(This register can be used to store the last FIFO byte in transparent transmit mode. Then this byte is repeatedly transmitted.)

A_CH_MSK [FIFO]	(r*/w)	(Reset group: H, 0, 1)	0xF4								
<p>HFC-channel data mask for the selected transmit HFC-channel</p> <p>For receive FIFOs this register is ignored.</p> <p>Before writing this array register, the HFC-channel must be selected by register R_FIFO.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Reset value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7..0</td> <td>0xFF</td> <td>V_CH_MSK</td> <td> <p>Mask byte</p> <p>This bitmap defines bit values for not processed bits of a HFC-channel. All not processed bits of a HFC-channel are set to the value defined in this register.</p> <p>This register has only a meaning when $V_BIT_CNT \neq 0$ in register A_SUBCH_CFG.</p> </td> </tr> </tbody> </table>				Bits	Reset value	Name	Description	7..0	0xFF	V_CH_MSK	<p>Mask byte</p> <p>This bitmap defines bit values for not processed bits of a HFC-channel. All not processed bits of a HFC-channel are set to the value defined in this register.</p> <p>This register has only a meaning when $V_BIT_CNT \neq 0$ in register A_SUBCH_CFG.</p>
Bits	Reset value	Name	Description								
7..0	0xFF	V_CH_MSK	<p>Mask byte</p> <p>This bitmap defines bit values for not processed bits of a HFC-channel. All not processed bits of a HFC-channel are set to the value defined in this register.</p> <p>This register has only a meaning when $V_BIT_CNT \neq 0$ in register A_SUBCH_CFG.</p>								

(See Section 2.2.3.2 on page 45 for details on Read* access.)

Bits	Reset value	Name	Description
HDLC and connection settings of the selected FIFO			
Before writing this array register the FIFO must be selected by register R_FIFO.			
0	0	V_IFF	Inter frame fill '0' = write HDLC flags 0x7E as inter frame fill '1' = write all '1's as inter frame fill Note: For D-channel this bit must be '1'.
1	0	V_HDLC_TRP	HDLC mode / transparent mode selection '0' = HDLC mode '1' = transparent mode Note: For D-channel this bit must be '0'.
4..2	0	V_FIFO_IRQ	FIFO interrupt configuration This bitmap has a different meaning in HDLC and transparent mode. Transparent mode (V_HDLC_TRP = '1'): The FIFO is enabled with any value $\neq 0$. A FIFO interrupt is generated all 2^n bytes when the bits [n-1:0] of the Z2-counter (in transmit direction) or the Z1-counter (in receive direction) change from all '1's to all '0's. $n = V_FIFO_IRQ + 2$. 0 = FIFO disabled, no interrupt 1 = FIFO enabled, all $2^3 = 8$ bytes an interrupt is generated 2 = FIFO enabled, all $2^4 = 16$ bytes an interrupt is generated 3 = FIFO enabled, all $2^5 = 32$ bytes an interrupt is generated 4 = FIFO enabled, all $2^6 = 64$ bytes an interrupt is generated 5 = FIFO enabled, all $2^7 = 128$ bytes an interrupt is generated 6..7 = FIFO enabled, no interrupt HDLC mode (V_HDLC_TRP = '0'): The FIFO is enabled with any value $\neq 0$. A FIFO interrupt can be generated at end of frame. 0 = FIFO disabled, no interrupt 1..7 = FIFO enabled, interrupt enabled Note: When mixed interrupt generation is selected with V_MIX_IRQ = '1' in register A_FIFO_CTRL, FIFO interrupts occur at end of frame as well as after 2^n bytes. Note: A FIFO must be enabled even for connections between line interface and PCM interface. No data transmission is performed with disabled FIFO.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7..5	0	V_DATA_FLOW	<p>Data flow configuration</p> <p>In transmit operation (V_FIFO_DIR = '0' in register R_FIFO):</p> <p>'000', '001' = FIFO → ST/U_p, FIFO → PCM '010', '011' = FIFO → PCM '100', '101' = FIFO → ST/U_p, ST/U_p → PCM '110', '111' = ST/U_p → PCM</p> <p>In receive operation (V_FIFO_DIR = '1' in register R_FIFO):</p> <p>'000', '100' = FIFO ← ST/U_p '001', '101' = FIFO ← PCM '010', '110' = FIFO ← ST/U_p, ST/U_p ← PCM '011', '111' = FIFO ← PCM, ST/U_p ← PCM</p> <p>Note: ST/U_p ↔ PCM configurations use V_FIFO_IRQ to enable the data transmission, i.e. V_FIFO_IRQ must not be zero. As received PCM-to-ST/U_p data is stored in the FIFO, interrupt generation can be used. ST/U_p -to-PCM data transmission is connected to a transmit FIFO and here no interrupt capability is available.</p>

(See Section 2.2.3.2 on page 45 for details on Read* access.)

Bits	Reset value	Name	Description
A_SUBCH_CFG [FIFO] (r*/w) (Reset group: H, 0, 1) 0xFB			
Subchannel parameters for bit processing of the selected FIFO			
Before writing this array register the FIFO must be selected by register R_FIFO.			
Note: For D- and E-channels this register must be 0x02.			
2..0	0	V_BIT_CNT	Number of bits to be processed in the HFC-channel byte In HDLC mode, only this number of bits is read from or written into the FIFO. In transparent mode always a whole FIFO byte is read or written, but only V_BIT_CNT bits contain valid data. '000' = process 8 bits (64 kbit/s) '001' = process 1 bit (8 kbit/s) '010' = process 2 bits (16 kbit/s) '011' = process 3 bits (24 kbit/s) '100' = process 4 bits (32 kbit/s) '101' = process 5 bits (40 kbit/s) '110' = process 6 bits (48 kbit/s) '111' = process 7 bits (56 kbit/s)
5..3	0	V_START_BIT	Start bit in the HFC-channel byte This bitmap specifies the position of the bit field in the HFC-channel byte. The bit field is located at position V_START_BIT in the HFC-channel byte. V_BIT_CNT + V_START_BIT must not be greater than 7 to get the bit field completely inside the HFC-channel byte. Any value greater than 7 produces an undefined behavior of the subchannel processor.
6	0	V_LOOP_FIFO	FIFO loop '0' = normal operation '1' = repeat current FIFO data (useful only in transparent mode) Note: This bit is ignored for receive FIFOs.
7	0	V_INV_DATA	Inverted data '0' = normal data transmission '1' = inverted data transmission

(See Section 2.2.3.2 on page 45 for details on Read* access.)

A_CHANNEL [FIFO]		(r*/w)	(Reset group: H, 0, 1)	0xFC
HFC-channel assignment for the selected FIFO				
This register is only used in <i>Channel Select Mode</i> and <i>FIFO Sequence Mode</i> .				
Before writing this array register the FIFO must be selected by register R_FIFO.				
Bits	Reset value	Name	Description	
0	0	V_CH_FDIR	HFC-channel data direction '0' = HFC-channel for transmit data '1' = HFC-channel for receive data Reset value: This bitmap is reset to the same value as the current FIFO, i.e. V_CH_FDIR of A_CHANNEL[number,direction] = direction.	
4..1	0	V_CH_FNUM	HFC-channel number (0..15) Reset value: This bitmap is reset to the same value as the current FIFO, i.e. V_CH_FNUM of A_CHANNEL[number,direction] = number.	
7..5		(reserved)	Must be '000' when written.	

(See Section 2.2.3.2 on page 45 for details on Read* access.)

Bits	Reset value	Name	Description
<p>A_FIFO_SEQ [FIFO] (r*/w) (Reset group: H, 0, 1) 0xFD</p> <p>FIFO sequence list</p> <p>This register is only used in <i>FIFO Sequence Mode</i>.</p> <p>Before writing this array register the FIFO must be selected by register R_FIFO.</p>			
0	0	V_NEXT_FIFO_DIR	<p>FIFO data direction</p> <p>This bit defines the data direction of the next FIFO in FIFO sequence.</p> <p>'0' = transmit FIFO data '1' = receive FIFO data</p> <p>Reset value: This bitmap is reset to the same value as the current FIFO, i.e. V_NEXT_FIFO_DIR of A_FIFO_SEQ[number,direction] = direction.</p>
4..1	0	V_NEXT_FIFO_NUM	<p>FIFO number</p> <p>This bitmap defines the FIFO number of the next FIFO in the FIFO sequence.</p> <p>Reset value: This bitmap is reset to the same value as the current FIFO, i.e. V_NEXT_FIFO_NUM of A_FIFO_SEQ[number,direction] = number.</p>
5		(reserved)	Must be '0' when written.
6	0	V_SEQ_END	<p>End of FIFO list</p> <p>'0' = FIFO list goes on '1' = FIFO list is terminated after this FIFO (V_NEXT_FIFO_DIR and V_NEXT_FIFO_NUM are ignored)</p>
7		(reserved)	Must be '0' when written.

(See Section 2.2.3.2 on page 45 for details on Read* access.)

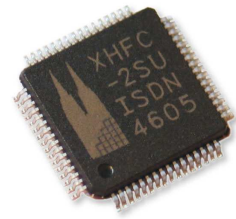
Bits	Reset value	Name	Description
A_FIFO_CTRL [FIFO] (r*/w) (Reset group: H, 0, 1) 0xFF			
Control register for the selected FIFO			
Before writing this array register the FIFO must be selected by register R_FIFO.			
0	0	V_FIFO_IRQMSK	Interrupt mask for the selected FIFO '0' = The FIFO interrupt is not used for generating a signal on the interrupt pin 14. The interrupt status can be read from registers R_FIFO_BL0_IRQ . . . R_FIFO_BL3_IRQ nevertheless. '1' = The FIFO interrupt event generates a signal on the interrupt pin 14. Note: In addition to this interrupt mask, FIFO interrupt must be enabled globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.
1	0	V_BERT_EN	BERT enable '0' = BERT disabled, normal data is transmitted and received '1' = BERT enabled, output of BERT generator is transmitted and received data is checked by BERT
2	0	V_MIX_IRQ	Mixed interrupt generation in HDLC mode This bit is only used in HDLC mode and it should be '0' in transparent mode. '0' = FIFO interrupts are generated either on <i>end of frame</i> (in HDLC mode) or periodically (in transparent mode) '1' = FIFO interrupts are generated both on <i>end of frame</i> and periodically when the bits [n-1:0] of the Z2-counter (in transmit direction) or the Z1-counter (in receive direction) change from all '1's to all '0's ($n = V_FIFO_IRQ + 3$ in register A_CON_HDLC)

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
3	0	V_FR_ABO	<p>Frame abort This bit has a different meaning for transmit and receive FIFOs.</p> <p>Transmit FIFO (FIFO underrun indication): When the selected transmit FIFO runs empty within a frame, a frame abort can be generated, i.e. at least sixteen consecutive '1's are transmitted. The receiver gets an invalid frame in this case. '0' = no frame abort after FIFO empty '1' = generate frame abort when FIFO runs empty within a frame ($Z1 = Z2$ and already $F1 = F2$, F-increment cannot be executed) Empty FIFO condition can be watched with bit V_FIFO_ERR in register A_FIFO_STA.</p> <p>Receive FIFO (aborted frame received): When the HDLC controller of the selected receive FIFO gets seven or more consecutive '1's, a frame abort condition can be signaled. '0' = no frame abort indication, FIFO status byte indicates only CRC error with any value not equal to 0x00 '1' = frame abort is indicated with FIFO status byte 0xFF, any other FIFO status byte 0x01 .. 0xFE indicates a CRC error</p> <p>Note: V_FR_ABO should be '0' in transparent mode.</p>
4	0	V_NO_CRC	<p>Suppress CRC transmission '0' = CRC is transmitted at the end of a frame (normal operation) '1' = CRC bytes are not transmitted at the end of a frame (must be done by the host processor instead)</p>
5	0	V_NO_REP	<p>No automatic repetition on HDLC frames (D-channel) '0' = After D-channel contention, the frame is automatically repeated '1' = After D-channel ontention, the frame is not repeated and it is aborted.</p>
7..6		(reserved)	Must be '00' when written.

(See Section 2.2.3.2 on page 45 for details on Read* access.)



Chapter 5

Universal ISDN Port

Table 5.1: Overview of the ST/ U_p interface registers

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x16	R_SU_SEL	193	0x13	R_AF0_OVIEW	209
0x30	A_SU_WR_STA	194	0x30	A_SU_RD_STA	210
0x31	A_SU_CTRL0	195	0x31	A_SU_DLYL	211
0x32	A_SU_CTRL1	197	0x32	A_SU_DLYH	212
0x33	A_SU_CTRL2	198	0x34	A_MS_RX	213
0x34	A_MS_TX	200	0x35	A_SU_STA	214
0x35	A_ST_CTRL3	201	0x3C	A_B1_RX	215
0x35	A_UP_CTRL3	202	0x3D	A_B2_RX	216
0x36	A_MS_DF	204	0x3E	A_D_RX	217
0x37	A_SU_CLK_DLY	205	0x3F	A_E_RX	218
0x3C	A_B1_TX	206			
0x3D	A_B2_TX	206			
0x3E	A_D_TX	207			
0x3F	A_BAC_S_TX	208			

5.1 General overview of the S/T and U_p interfaces

The Universal ISDN Port consists of a line interface with both S/T and U_p signaling capability. It can be configured to TE or NT/LT mode.

5.1.1 Array registers and multiregisters

Every line interface can be switched either into S/T or U_p mode separately; so they are called *Universal ISDN Port*. For both modes there is a complete set of registers. These registers are multi-registers, which means they have the same address and I/O functionality (write or read) but different meaning or bitmap structure. Yet, most of them are very similar in S/T and U_p mode.

Furthermore, all registers are described to be array registers because there are several line interfaces in XHFC-2SU and each has the complete set of registers. So, the Universal ISDN Port module – with multiple entities – has registers which are multi-registers *and* array registers at the same time.

Register accesses concerning the line interfaces have to be done as follows:

- 1. Array register selection:** The line interface number must be selected first by writing the appropriate value into bitmap V_{SU_SEL} of register R_{SU_SEL}.
 - XHFC-2S4U: S/T interfaces 0..1 available, U_p interfaces 0..3 available
 - XHFC-4SU: S/T interfaces 0..3 available, U_p interfaces 0..3 available
- 2. Multiregister selection:** Then the interface mode can be chosen (only in the initialization procedure). The selected line interface works in S/T mode with the bitmap value V_{ST_SEL} = '0' in register A_{ST_CTRL3}. The line interface can be switched into U_p mode with V_{ST_SEL} = '1'¹. It is strongly recommended to select the interface mode before any other register of the selected line interface is written.
- 3. Any other register access:** Now all registers of the selected line interface in its chosen mode (S/T or U_p) can be accessed.

These three steps describe the initialization procedure after reset. Afterwards, step 2 must not be executed again, of course.



Important !

The Universal ISDN Port is in S/T mode after reset. When U_p mode is required, it should be selected after reset, i.e. no other register of the line interface module should be written before.

5.1.2 Block diagram of the Universal ISDN Port module

The line interface module consists of the receive and transmit data pathes with a clock processing unit each, the clock distribution unit and the state machine. The overall connections of these units are shown in Figure 5.1.

¹Please note, that there is an array register at address 0x35 which is also a multi-register. For a line interface in S/T mode, A_{ST_CTRL3} is used while a line interface in U_p mode uses A_{UP_CTRL3}.

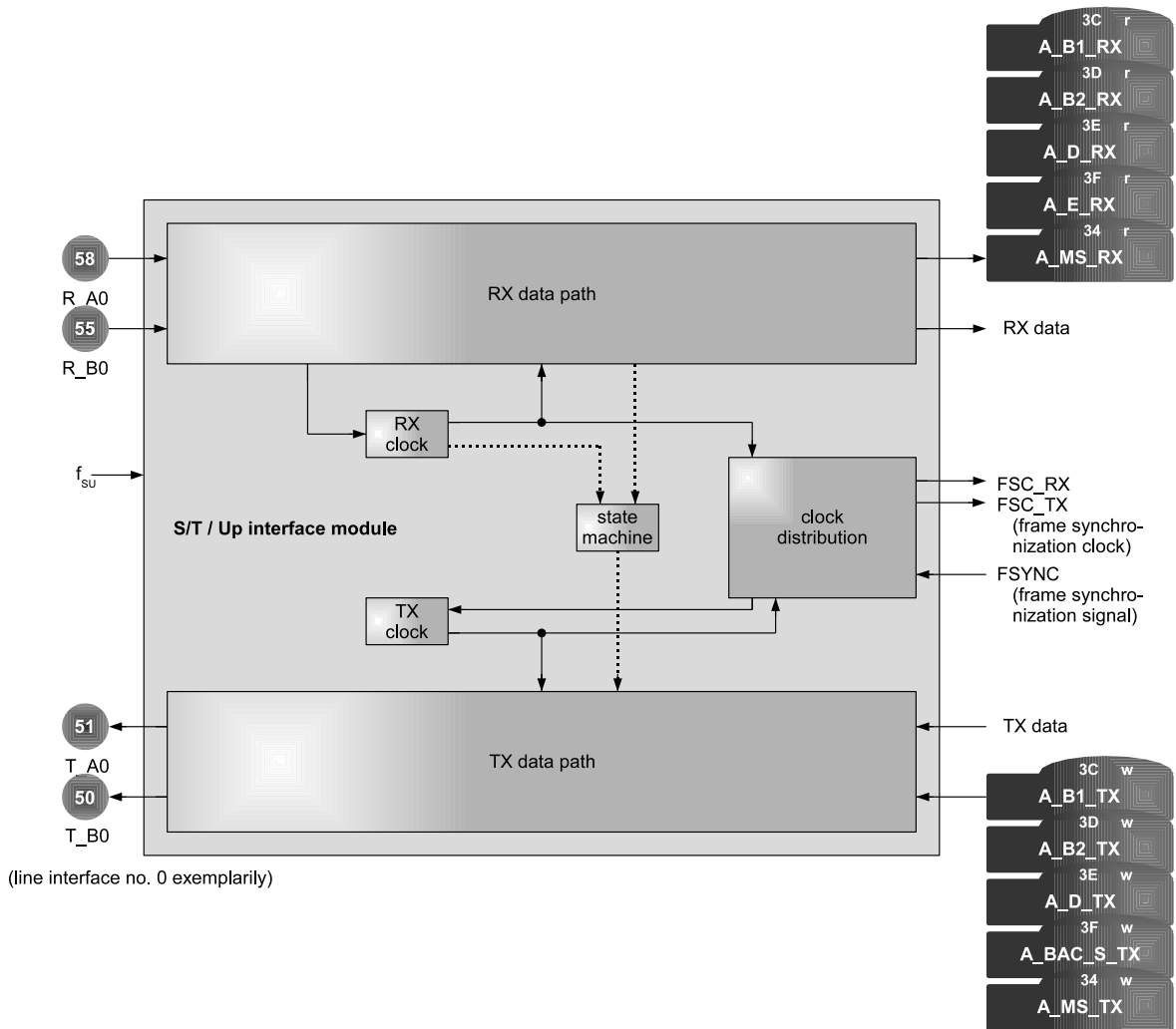


Figure 5.1: Overview of the Universal ISDN Port module (For details see Figures 5.13 or 5.5)

5.2 S/T interface description

5.2.1 Overview

The Universal ISDN Port is able to provide an S/T line interface in TE or NT mode according to ITU-T I.430 [9] and ETSI TBR 003[4] specifications.

The line interface is a four-wire interface and has separated transmitter and receiver with configurable behaviour. The ISDN data frame structure is handled by hardware. Thus plain data is processed on the host side of the S/T interface.

A specification conform state machine for TE and NT mode is implemented (see Section 5.2.6).

The S/T interface uses the modified AMI coding for input and output signals. This pseudo-ternary coding converts logical ones to 0 V level. Logical zeros are coded by alternating positive and negative voltage with 750 mV nominal amplitude on the line.

5.2.2 Frame Structure

The frame structures on the S/T interface are different for each direction of transmission. Both structures are illustrated in Figure 5.2. The raw data bit rate is 192 kBit/s in transmit and receive direction.

HDLC B-channel data starts with the LSB, PCM B-channel data starts with the MSB.

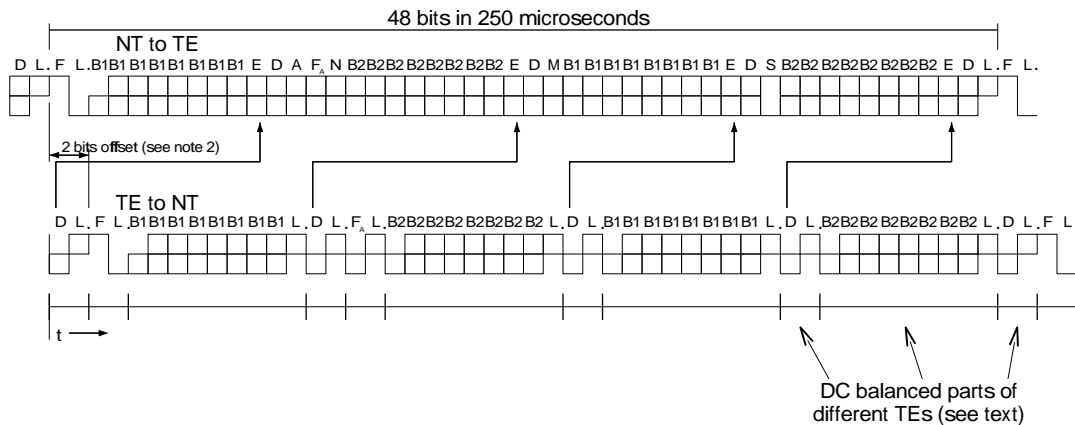


Figure 5.2: Frame structure at reference points S and T (see legend in Table 5.2 and specification [9])

The nominal 2 bit offset is as seen from the TE in Figure 5.2. The offset can be adjusted for TE mode with bitmap V_SU_CLK_DLY in register A_SU_CLK_DLY. The corresponding offset at the NT is not fixed and may be greater due to delay in the interface cable and varies by configuration.

The TE-to-NT transmission has 10 balancing bits within every frame to achieve independent DC balanced parts for different TEs. This is indicated by lines below the frame structure in Figure 5.2.

In the NT-to-TE direction there is only one real DC-balance bit at the end of the frame because all data comes from the same source. Another L-bit at the beginning of the frame belongs to the preceding F-bit and is used for code violation.

Table 5.2: Legend for Figure 5.2

NT-to-TE & TE-to-NT:		NT-to-TE only:	
Code	Description	Code	Description
F	Framing bit, marks the start of the frame (1 bit/frame)	E	Bit within the E-channel (D-echo-channel, 4 bit/frame)
B1	Bit within the B1-channel (2 byte/frame)	M	Multiframe bit, marks the start of the multiframe in every 20th frame (1 bit/frame)
B2	Bit within the B2-channel (2 byte/frame)	N	Boolean complementation of the auxiliary framing bit F_A , $N = \bar{F}_A$ (1 bit/frame)
D	Bit within the D-channel (4 bit/frame)	S	S-bit of the multiframe (1 bit/frame)
L	DC balancing bit (NT-to-TE: 2 bit/frame, TE-to-NT: 10 bit/frame)	A	Activation bit (1 bit/frame)
F_A	NT-to-TE: Auxiliary framing bit, marks the start of subchannel 1 in every 5th frame, a multiframe bit (S-bit) is transmitted in the same frame (1 bit/frame) TE-to-NT: Q-bit of the multiframe (1 bit/frame)		

5.2.3 Multiframe structure

There is a higher frame structure called *multiframe*. A multiframe has the length of 4 bits and consists of the bits Q1, Q2, Q3 and Q4 (TE-to-NT) or S1, S2, S3 and S4 (NT-to-TE). Q1 and S1 are transmitted first. As there is one multiframe bit transferred every fifth 250 μ s cycle, a complete multiframe is transferred every 5 ms. This means that a multiframe has a length of 20 S/T frames.

The F_A - and M-bits are used to identify the multiframe. Table 5.3 shows the position of the multiframe bits. A detailed specification of the multiframe structure is given in [9].

Multiframe transmission must be enabled with $V_ST_SQ_EN = '1'$ in register A_SU_CTRL0.

5.2.4 Data transmission

B-channel data on the line interface must be enabled for transmit and receive direction separately.

$V_B1_TX_EN = '1'$ in register A_SU_CTRL0 enables data transmission for the B1-channel and $V_B2_TX_EN = '1'$ in the same register enables data transmission for the B2-channel.

$V_B1_RX_EN = '1'$ in register A_SU_CTRL2 enables data receive for the B1-channel and $V_B2_RX_EN = '1'$ in the same register enables data receive for the B2-channel.

Disabled B-channel data means that all bits are forced to '1' on the line.

Figures 5.3 and 5.4 show the composition and decomposition of the S/T frames. B1-, B2-, D- and E-channel data is normally handled by the data flow controller. The HDLC controller as well as the PCM interface deliver data to the S/T interface and receive data from the S/T interface. For this reason, registers A_B1_TX, A_B2_TX, A_D_TX and A_BAC_S_TX as well as A_B1_RX, A_B2_RX, A_D_RX and A_E_RX are normally not written or read from the application software.

Multiframe bits can be handled by the data flow controller (involving registers A_BAC_S_TX and A_E_RX) or manually from the application software with registers A_MS_TX and A_MS_RX.

Table 5.3: *Multiframe structure of the Q- and S-bits*

Frame number	NT-to-TE frame synchronization		TE-to-NT multiframe	NT-to-TE multiframe
	F _A -bit	M-bit	Q-bits in F _A bit position ^{*1}	S-bits ^{*2}
1	'1'	'1'	Q1	S1
2	'0'	'0'	'0'	'0'
3	'0'	'0'	'0'	'0'
4	'0'	'0'	'0'	'0'
5	'0'	'0'	'0'	'0'
6	'1'	'0'	Q2	S2
7	'0'	'0'	'0'	'0'
8	'0'	'0'	'0'	'0'
9	'0'	'0'	'0'	'0'
10	'0'	'0'	'0'	'0'
11	'1'	'0'	Q3	S3
12	'0'	'0'	'0'	'0'
13	'0'	'0'	'0'	'0'
14	'0'	'0'	'0'	'0'
15	'0'	'0'	'0'	'0'
16	'1'	'0'	Q4	S4
17	'0'	'0'	'0'	'0'
18	'0'	'0'	'0'	'0'
19	'0'	'0'	'0'	'0'
20	'0'	'0'	'0'	'0'
1	'1'	'1'	Q1	S1
2	'0'	'0'	'0'	'0'
...				

^{*1}: If the Q-bits are not used by a TE, the Q-bits shall be set to '1' (i.e. echoing of the received F_A bits).

^{*2}: The specification [9] defines five subchannels for the S-multiframe. Only subchannel 1 is used from XHFC-2SU. All other subchannels are set to '0'.

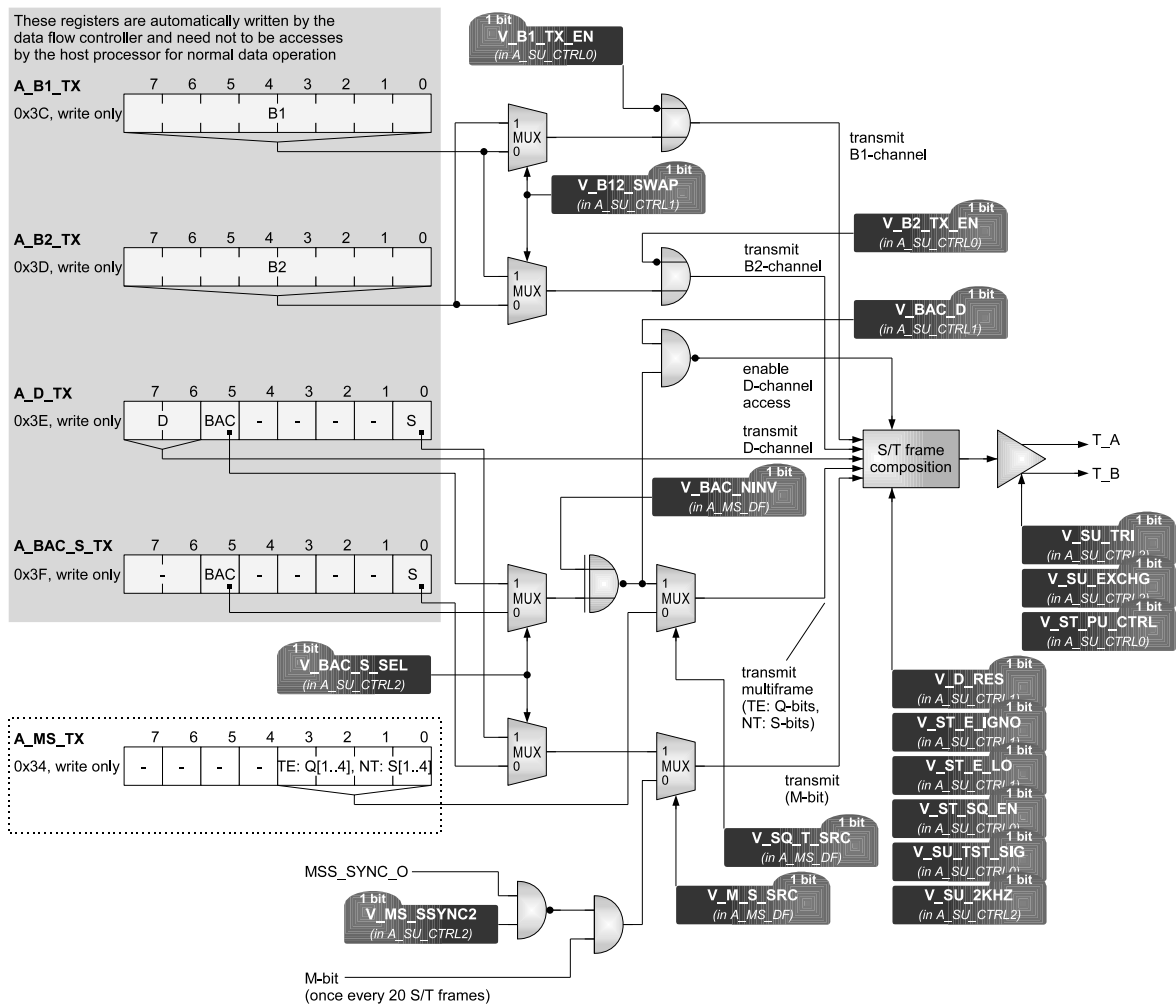


Figure 5.3: S/T frame composition for B1-, B2-, D- and multiframe bits (S/T interface mode, transmit direction)

5.2.5 INFO signals

Signals which are transmitted on the interface line are called *INFO signals*. INFO 0 is defined for both TE-to-NT and NT-to-TE directions. All other INFO signals are either for TE-to-NT signaling (INFO 1, INFO 3) or NT-to-TE signaling (INFO 2, INFO 4). The INFO signals are defined as follows²:

INFO 0: No signal on line.

INFO 1: Continuous signal at nominal bit rate with a '0011 1111' pattern which has a positive zero first and a negative zero following.

INFO 2: Frames with A-bit and all B-, D- and E-bits in the frame are set to binary zero. The F_A-, N- and L-bits are set according to the normal coding rule.

INFO 3: Synchronised frames with 2 bit offset and operational data on B- and D-channels.

INFO 4: Frames with operational data on B-, D- and E-channels. The A-bit is set to binary one.

²Please see [9] for a detailed description of the INFO signals.

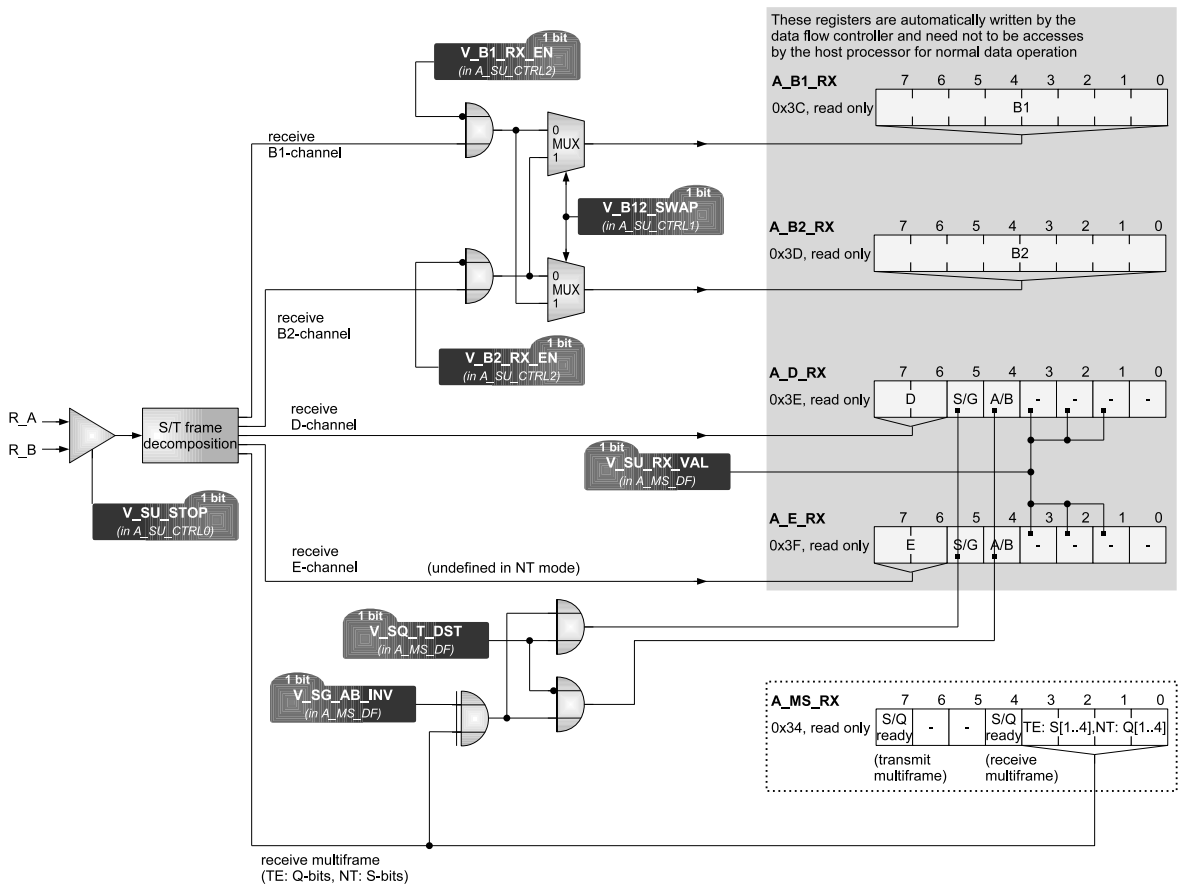


Figure 5.4: S/T frame decomposition for B1-, B2-, D-, E-channel and multiframe bits (S/T interface mode, receive direction)

5.2.6 State machine

A specification conform state machine for TE and NT mode is implemented [9]. So the current Fx or Gx state can be read out of register A_SU_RD_STA. However, it is possible to overwrite the current state by setting bit V_SU_LD_STA in register A_SU_WR_STA.

Activation and deactivation can be initiated by writing bitmap V_SU_ACT in the same register. This bitmap can be used for TE and NT mode and can start activation or deactivation from any state. Even in TE mode it can be used to initiate a deactivation from any state to F3. Such a deactivation should only be initiated if the state machine is not in F6 or F7, of course. Writing '11' (start activation) when the state machine is already activated (G2/G3 or F6/F7), will not change the current state.

Before starting the state machine, register A_SU_CLK_DLY of its S/T interface must be set. The default value is 0x0E for TE and 0x6C for NT mode.



Important !

The S/T state machine is stuck at F0 or G0 after a reset. The interface sends no signal on the S/T line (INFO 0) and is not able to activate it by incoming INFO x in this state. Writing '0' into bit V_SU_LD_STA of register A_SU_WR_STA starts the state machine.

NT mode: The NT state machine does not change automatically from G2 to G3 if the TE side sends INFO 3 frames. This transition must be activated each time by V_G2_G3 in register A_SU_RD_STA or it can permanently be enabled by setting bit V_G2_G3_EN in register A_SU_CTRL1.

Incoming INFO 0 at state F6 cause a state change to F3 normally. Sometimes an intermediate state F7 occurs which stops timer T3. In this case another state change to F3 comes up within 1 ms and F7 can be ignored. V_SU_INFO0 in register A_SU_RD_STA should be checked with every state change from F6 to F7. When this bit is set, the state should be checked again after about 1 ms. When it is F3, the intermediate state F7 has to be ignored.³

Tables 5.4 and 5.5 show the S/T interface activation and deactivation layer 1 of the finite state matrix in NT and TE mode.

³It might be useful to start a timer of approximately 1 ms to detect the F6 – F7 – F3 state changes.

Table 5.4: S/T interface activation/deactivation layer 1 matrix for NT mode

State name:	Reset	Deactivated	Pending activation	Active	Pending deactivation
State number:	G 0	G 1	G 2	G 3	G 4
INFO sent:	INFO 0	INFO 0	INFO 2	INFO 4	INFO 0
Event:					
State machine release ^{*3}	G 1				
Activate request	start T 1 ^{*1} G 2	start T 1 ^{*1} G 2			start T 1 ^{*1} G 2
Deactivate request	—		start T 2 G 4	start T 2 G 4	
Expiry T 1 ^{*1}	—	—	start T 2 G 4	/	—
Expiry T 2 ^{*2}	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1	—	start T 1 ^{*1} G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 ^{*1,4} G 3	—	—
Lost framing	—	/	/	G 2	—
Legend:	—	No state change			
	/	Impossible situation			
		Impossible by the definition of the layer 1 service			

^{*1}: Timer T 1 is not implemented and must be implemented in software. T 1 is started with entering G2, runs during G2 state and is stopped when entering G3 or expiry. T 1 should expire after 100 ms . . 1000 ms [5].

^{*2}: Timer T 2 prevents unintentional reactivation. Its value is $256 \cdot 125 \mu\text{s} = 32 \text{ms}$. This implies that a TE has to recognize INFO 0 and to react on it within this time.

^{*3}: After reset the state machine is fixed to G 0.

^{*4}: Bit V_SU_SET_G2_G3 in register A_SU_WR_STA must be set to allow this transition or V_G2_G3_EN in register A_SU_CTRL1 must be set to allow automatic transition G 2 → G 3.

Table 5.5: S/T interface activation/deactivation layer 1 matrix for TE mode

State name:	Reset	Sensing	Deactivated	Awaiting signal	Identifying input	Synchronized	Activated	Lost framing
State number:	F0	F2	F3	F4	F5	F6	F7	F8
INFO sent:	INFO 0	INFO 0	INFO 0	INFO 1	INFO 0	INFO 3	INFO 3	INFO 0
Event:								
State machine release ^{*1}	F2	/	/	/	/	/	/	/
Activate request, receiving any signal	—		F5			—		—
receiving INFO 0	—		start T3 ^{*5} F4			—		—
Expiry T3 ^{*5}	—	/	—	F3	F3	—	—	F3
Receiving INFO 0	—	F3	—	—	—	F3	F3	F3
Receiving any signal ^{*2}	—	—	—	F5	—	/	/	—
Receiving INFO 2 ^{*3}	—	F6	F6	F6	F6	—	F6	F6
Receiving INFO 4 ^{*3}	—	F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	—	stop T3 ^{*5} F7
Lost framing ^{*4}	—	/	/	/	/	F8	F8	—

Legend: — No state change
 / Impossible situation
 | Impossible by the definition of the layer 1 service

^{*1}: After reset the state machine is fixed to F0.

^{*2}: This event reflects the case where a signal is received and the TE has not (yet) determined whether it is INFO 2 or INFO 4.

^{*3}: Bit and frame synchronization achieved.

^{*4}: Loss of Bit or frame synchronization.

^{*5}: Timer T3 is not implemented and must be implemented in software.

5.2.7 Clock synchronization

A detailed view inside the line interface block diagram of Figure 5.1 is shown for the S/T interface mode in Figure 5.5. All clocks are derived from a 6.144 MHz clock which is $f_{SU}/2$. Frame synchronization is accomplished by evaluating the code violations in the S/T frame.

Received data from the pins R_A0/R_A1 and R_B0/R_B1 is passed through the RX data path to the switching buffer (see Figures 3.3 and 3.4 in Section 3.2). A bit clock and a frame clock are derived from the received data stream. These clocks are used to synchronize the RX data path timing to the incoming data stream. The frame clock can be passed for synchronization purposes to the TX data path and to the PCM timing control as well.

The transmit data clock has different sources in TE and NT mode:

NT mode: The 192 kHz bit clock as well as the 8 kHz frame clock are derived from FSYNC in NT mode. This signal is either F0IO input or F1_7 (see register R_SL_SEL7 and Figure 6.5 on page 229).

TE mode: A TE is always taken as synchronization source for ISDN applications because it delivers the clock from the central office switch. Thus both clocks are taken from the RX clock unit.

The state machine takes several signals from the RX data path and the RX clock unit. The TX data path is controlled by the state machine's output signals.

The multiframe transmission can be synchronized to the PCM interface. The MSS controller (multi-frame/superframe synchronization controller) delivers the MSS_SYNC_O signal to force the 'start of multiframe' in NT mode. The MSS controller is described in Section 6.6 from page 235.

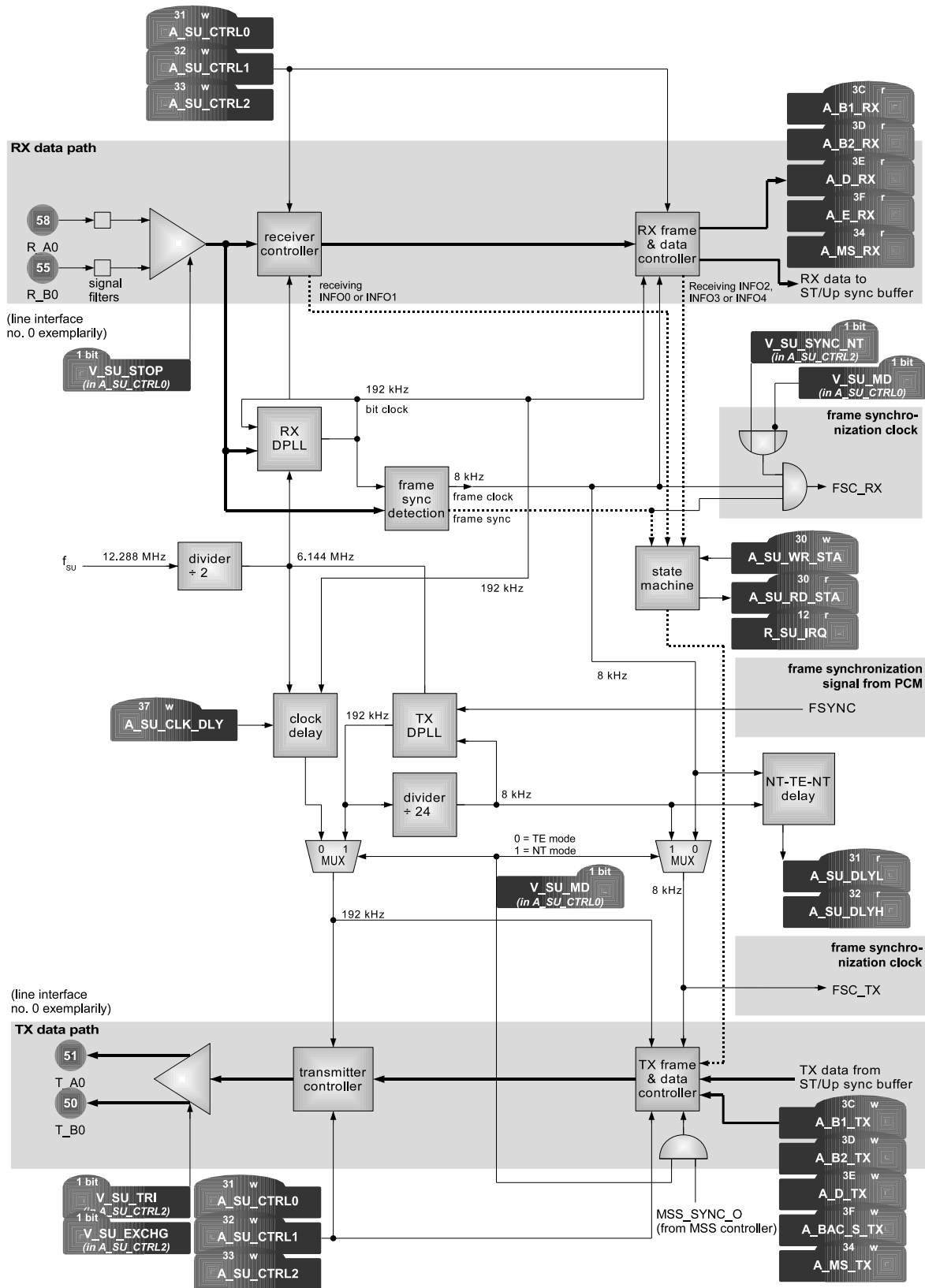


Figure 5.5: S/T clock synchronization

5.2.8 External circuitries

5.2.8.1 External receive circuitry

The standard external receive circuitry for TE and NT mode is shown in Figure 5.6.

Figure 5.6 connects pins R_A0/L_A0 through the transformer to a minus (-) pin of the RJ-45 jack, while pins R_B0/L_B0 are connected to a RJ-45 plus (+) pin. Due to the automatic polarity detection of the XHFC-2SU receiver, it is allowed to swap the pairs R_A0/L_A0 and R_B0/L_B0.

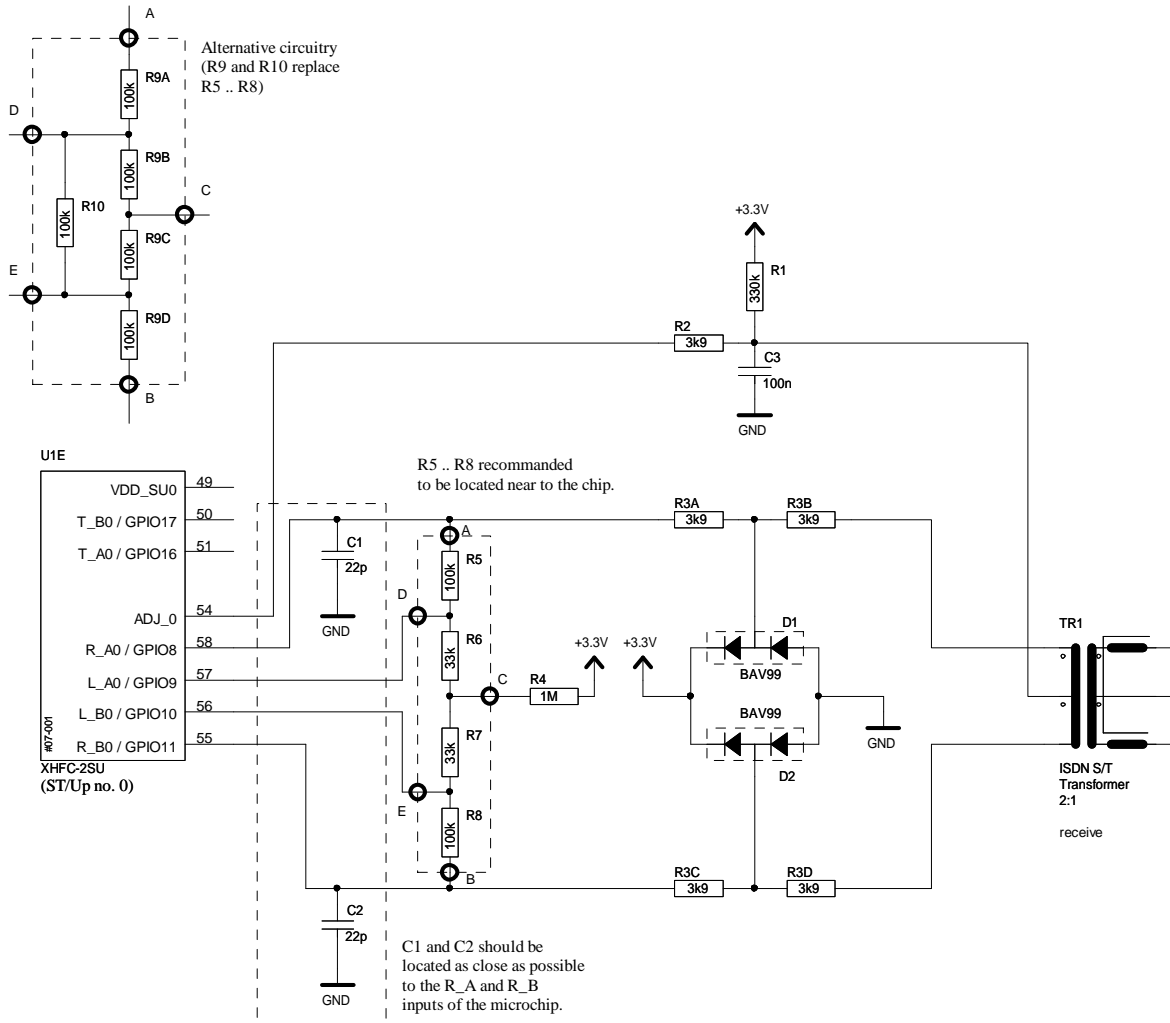


Figure 5.6: External S/T receive circuitry for TE and NT mode

XHFC-2SU has two S/T interfaces. For a not used S/T interface, the level adjustment pin ADJ_0 or ADJ_1 should be left open.

5.2.8.2 External transmit circuitry

The standard external transmit circuitry for TE and NT mode is shown in Figure 5.7.

Figure 5.7 connects pin T_A0 through the transformer to a minus (-) pin of the RJ-45 jack, while pin T_B0 is connected to a RJ-45 plus (+) pin. This is important for interoperability with other devices. Mainly for test purposes, the transmit lines can be swapped internally with V_SU_EXCHG = '1' in register A_SU_CTRL2.

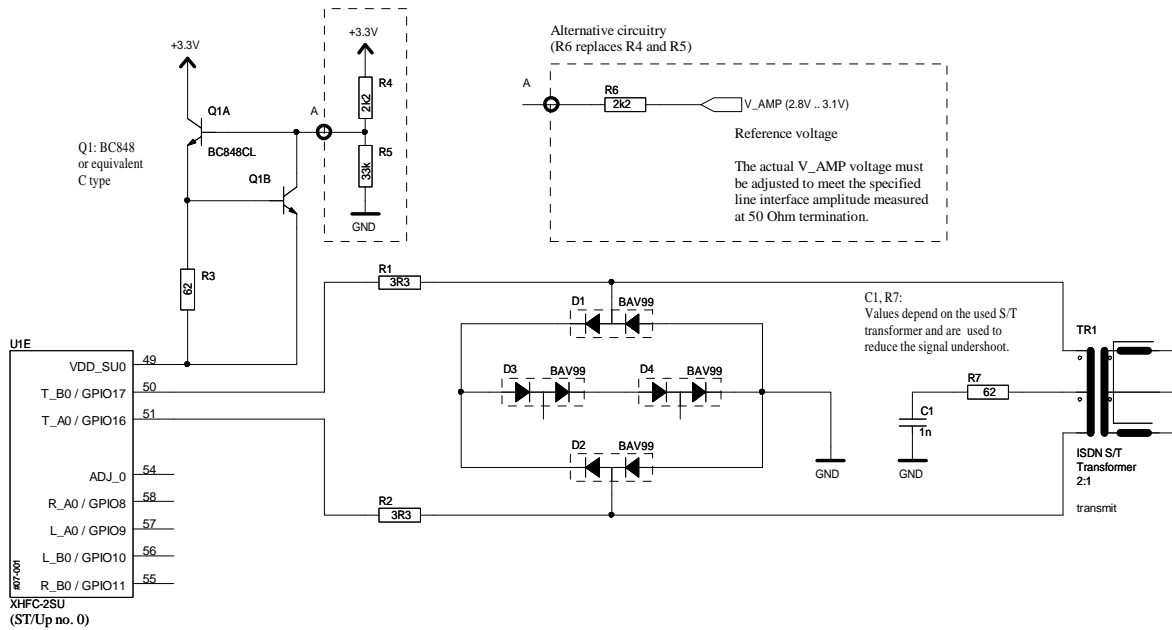


Figure 5.7: External S/T transmit circuitry for TE and NT mode

The signal level of the transmit circuitry has to be adjusted by VDD_SU0/VDD_SU1. The exact voltage of these pins depends on the used transformer and circuitry dimensioning.

5.2.8.3 Transformer and ISDN jack connection

Figure 5.8 and 5.9 show the connection circuitry of the transformer and the ISDN jack in TE mode⁴. The termination resistors R1 and R2 are optional.

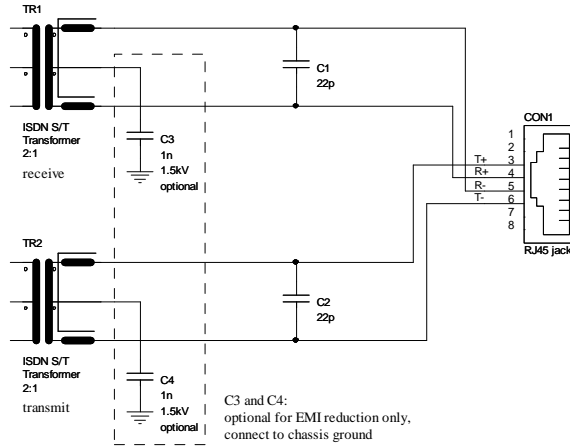


Figure 5.8: Transformer and connector circuitry in TE mode

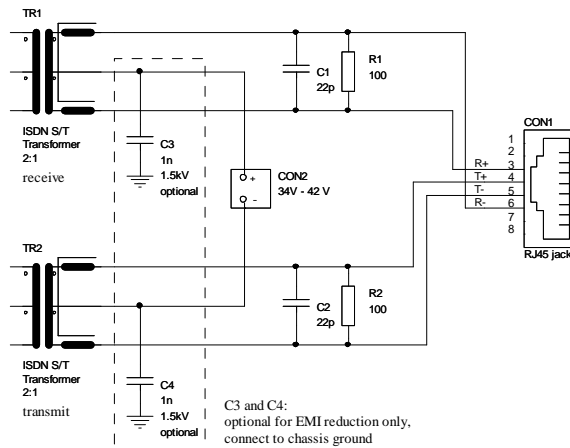
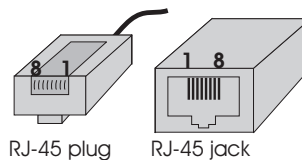


Figure 5.9: Transformer and connector circuitry in NT mode (shown with optional 100Ω termination, whole bus termination must be 50Ω)

⁴The ISDN jack RJ-45 has 8 pins and carries two pairs of wires. Standard configuration is

- pin 3: TE (+) transmit → NT (+) receive,
- pin 4: NT (+) transmit → TE (+) receive,
- pin 5: NT (-) transmit → TE (-) receive,
- pin 6: TE (-) transmit → NT (-) receive.



5.2.9 S/T transformers

Customers of Cologne Chip can choose from a variety of S/T transformers for ISDN Basic Rate Interface. All S/T transformers are compatible to the XHFC series of Cologne Chip that fulfill two criteria:

- Turns ratio of 1:2 (line side : chip side)
- Center tap on the chip side (required for Cologne Chip transmitter and receiver circuitries)

Several companies provide *transformers* and *transformer modules* that can be used with our ISDN Basic Rate Interface controllers. Most popular are SMD dual transformer modules with choke for EMI reasons. Part numbers and manufacturers are listed in Table 5.6. A more extensive and regularly updated list can be found on Cologne Chip's website <http://www.colognechip.com>.

The transformer list has not been compiled under aspects of RoHS compliance. For the current RoHS status of the listed parts, please contact the transformer manufacturers straight.

Table 5.6: S/T transformer part numbers and manufacturers

Bel Fuse Inc., United Kingdom, http://www.belfuse.com			
<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Dual Transformer Module without choke:		Dual Transformer Module with choke:	
	2798B (SMD)		APC 48301 (THT)
	2798C (SMD)		
	2798D (SMD)		
Pulse Engineering, Inc., United States, http://www.pulseeng.com			
<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Single transformer:		Dual Transformer Module without choke:	
	T5003 (SMD, PCMCIA)		T5006 (SMD)
	T5020 (SMD)		T5007 (SMD)
	T5023 (SMD)		T5042 (SMD, 3 kV)
	T5024 (SMD, 3 kV)		PE-65495 (THT)
	T5033 (SMD)		PE-65499 (THT)
	T5035 (THT, 3 kV)		PE-65795 (SMD)
	T5036 (SMD, 3 kV)		PE-65799 (SMD)
	PE-64995 (THT)		
	PE-64999 (THT)	Dual Transformer Module with choke:	
	PE-68992 (THT)		T5012 (THT)
	ST-5069 (SMD)		T5034 (SMD)
			T5038 (SMD)

(continued on next page)

Table 5.6: S/T transformer part numbers and manufacturers

(continued from previous page)

Talema Elektronik GmbH, Germany, <http://www.talema.net>

<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Single Transformer:		Dual Transformer Module with choke:	
	ISF-140B1 (THT)		HVM-140C1 (THT)
	ISV-140B1 (THT)		ISM-140C1 (THT)
	ISHF-240B1 (THT, 3 kV)		MUJ-103A-500(SMD, miniature)
	SHJ-240B (SMD, 3 kV)		MUJ-103A-101(SMD, miniature)
	SMJ-140B (SMD)		MUJ-103A-501(SMD, miniature)
	SWJ-140B (SMD)		MUJ-103A-502(SMD, miniature)
Dual Transformer Module without choke:			MAJ-403A-470 (SMD)
	MUJ-103A-000 (SMD, miniature)		MAJ-403A-101 (SMD)
			MAJ-403A-501 (SMD)
			MAJ-403A-502 (SMD)
			MSJ-403A-470 (SMD)
			MSJ-403A-101 (SMD)
			MSJ-403A-501 (SMD)
			MSJ-403A-502 (SMD)
			MHJ-240B1-470 (SMD, 3 kV)
			MHJ-240B1-101 (SMD, 3 kV)
			MHJ-240B1-501 (SMD, 3 kV)
			MHJ-240B1-502 (SMD, 3 kV)
			MHJ-240B1-123 (SMD, 3 kV)

(continued on next page)

Table 5.6: S/T transformer part numbers and manufacturers

(continued from previous page)

UMEC GmbH, Germany, Taiwan, United States, <http://www.umec.de>

<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Single transformer:		Dual Transformer Module without choke:	
	UT 20995 (THT)		UT 20495 (THT)
	UT 20999 (THT)		UT 20495-TS (SMD)
	UT 21023 (THT)		UT 20765-00 (SMD, 3 kV)
	UT 21595 (THT)		UT 20795-00TS (SMD)
	UT 28166 (THT)		
	UT 28166-TS (SMD)	Dual Transformer Module with choke:	
	UT 28428-TS (SMD)		UT 20495 CV-TS (SMD)
	UT 28729 (THT, 4 kV)		UT 20765-05TS (SMD, 3 kV)
			UT 20765-10TS (SMD, 3 kV)
			UT 20765-50TS (SMD, 3 kV)
			UT 20795-05TS (SMD)
			UT 20795-10TS (SMD)
			UT 20795-50TS (SMD)
			UT 20795-5M-TS (SMD)
			UT 21644S (SMD, miniature)
			UT 28624 (THT)
			UT 28624A (THT)
			UT 28624A-T (SMD)

Vacuumschmelze GmbH & Co. KG, Germany, <http://www.vacuumschmelze.com>

<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Single transformer:		Dual Transformer Module without choke:	
	3-L4021-X066 (THT)		7-M4035-X001 (THT)
	3-L4025-X095 (THT)		7-M5014-X001 (SMD, miniature)
	3-L4031-X001 (THT)		7-M5026-X001 (SMD)
	3-L4097-X029 (THT, 3 kV)		7-M5026-X002 (SMD, 3 kV)
	3-L5024-X028 (SMD)		7-M5054-X001 (SMD)
	3-L5032-X040 (SMD, 3 kV)		
		Dual Transformer Module with choke:	
			7-L5026-X010 (SMD)
			7-L5026-X011 (SMD, 3 kV)
			7-L5026-X017 (SMD)
			7-L5051-X014 (THT)
			7-L5054-X005 (SMD, 3 kV)
			7-L5054-X006 (SMD, 3 kV)

(continued on next page)

Table 5.6: S/T transformer part numbers and manufacturers

(continued from previous page)

Sumida AG, Germany, <http://www.sumida-eu.com> (formerly known as Vogt electronic AG)

<u>Type</u>	<u>Device</u>	<u>Type</u>	<u>Device</u>
Single transformer:		Dual Transformer Module without choke:	
	503 05 901 00 (SMD)		503 16 504 00 (SMD)
	503 10 009 00 (SMD)		503 16 513 00 (SMD, 4 kV)
	503 12 001 00 (SMD, PCMCIA)		503 20 981 00 (SMD)
	503 20 010 00 (SMD)		503 74 003 00 (SMD, 4 kV)
	503 20 019 00 (SMD, 4 kV)		503 74 006 00 (SMD)
	543 80 008 00 (THT, 4 kV)	Dual Transformer Module with choke:	
			503 16 017 00 (SMD, miniature)
			503 16 501 00 (SMD)
			503 16 502 00 (SMD)
			503 16 505 00 (SMD)
			503 16 506 00 (SMD)
			503 20 985 00 (SMD)
			543 76 006 00 (SMD)

Please note: Cologne Chip cannot take any liability concerning the product names, characteristics and availability. Products can change without notice. Please refer to the manufacturer in case of doubt.

5.3 U_p interface description

5.3.1 Overview

The Universal ISDN Port is able to provide a ping-pong type 2-wire interface according to U_{p0} and U_{pN} specifications [3] known from SIEMENS Corporation and German Electrical and Electronic Manufacturers' Association (ZVEI).

The line interface is a four-wire interface and has separated transmitter and receiver with configurable behaviour. The ISDN data frame structure is handled by hardware. Thus plain data is processed on the host side of the U_p interface.

A specification conform state machine for TE and LT mode is implemented (see Section 5.3.6). This is very similar to the S/T state machine (see Section 5.2.6 on page 163).

The U_p interface uses AMI coding for input and output signals. This pseudo-ternary coding converts logical zeros to 0 V level. Logical ones are coded by alternating positive and negative voltage with 2 V nominal amplitude on the line.

5.3.2 Frame Structure

The U_p frame structure has a length of 250 μ s. Within this period, a transmit phase and a receive phase with 99 μ s length each is placed as shown in Figure 5.10. The 38 bit frame has the same structure for LT-to-TE and TE-to-LT transmissions.

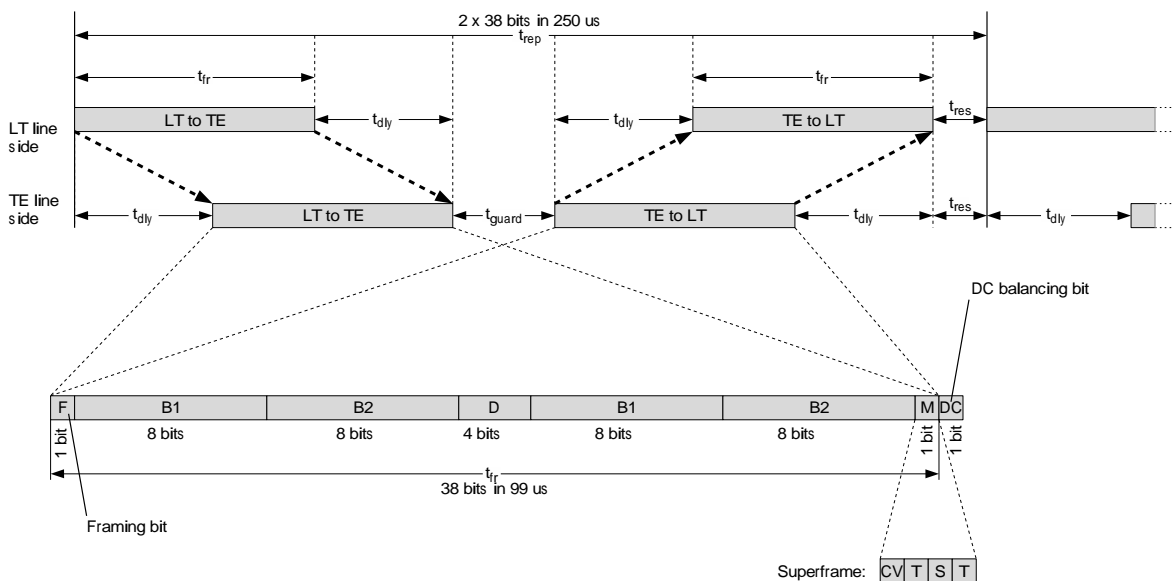


Figure 5.10: U_p interface frame structure

5.3.3 Superframe structure

There is a higher frame structure called *superframe* (also called *M-channel*). A superframe has the length of 4 bits and consists of the bits CV, T and S as shown in Figure 5.10 and Table 5.8. As there is one superframe bit within every frame, a complete superframe is transmitted and received every 1 ms.

Table 5.7: Legend for Figure 5.10

Symbol	min / μ s	typ / μ s	max / μ s	Characteristic
t_{rep}		250		Burst repetition period
t_{fr}		99		Frame time
t_{dly}	0		20.8	Line delay
t_{guard}		5.2		Guard time (2 bits)
t_{res}	0		41.6	Residual time

The CV-bit is used for the superframe synchronization. It has always the value '1' and produces a code violation.

Superframe data is transferred through the T-bits while the S-bit can be used for service bits. Received bits are stored in register A_MS_RX. When two complete superframes are received, bit V_MS_RX_RDY is set to '1' in the same register. Then the four T-bits in V_MS_RX – received within two superframes – are valid. V_MS_RX_RDY is reset to '0' with every read access to register A_MS_RX. The first received T-bit is stored in V_MS_RX[3], the fourth is stored in V_MS_RX[0].

Superframe data to be transmitted must be stored in bitmap V_MS_TX of register A_MS_TX. Four T-bits must be stored together. They are transmitted with the next two superframes. The first T-bit is V_MS_TX[3] and the fourth is V_MS_TX[0]. When all T-bits are transferred to the output shift register, bit V_MS_TX_RDY in register A_MS_RX changes to '1' to signal 'next data required'. This bit is automatically reset to '0' with a read access to register A_MS_RX.

The received service bit S can be read from register V_UP_S_RX. In transmit data direction, the bit value V_UP_S_TX in register A_MS_TX will be send in the next superframes until another value is written into V_UP_S_TX.

Table 5.8: Superframe construction

Number in sequence	Bit name	Meaning	Bit rate
1	CV	Code violation bit	1 kbit/s
2, 4	T	Transparent channel bit	2 kbit/s
3	S	Service channel bit	1 kbit/s

5.3.4 Data transmission

B-channel data on the line interface must be enabled for transmit and receive direction separately.

V_B1_TX_EN = '1' in register A_SU_CTRL0 enables data transmission for the B1-channel and V_B2_TX_EN = '1' in the same register enables data transmission for the B2-channel.

V_B1_RX_EN = '1' in register A_SU_CTRL2 enables data receive for the B1-channel and V_B2_RX_EN = '1' in the same register enables data receive for the B2-channel.

Disabled B-channel data means that all bits are forced to '1' on the line. Due to the fact that the bit scramblers are functionally arranged in front of the B-channel enable/disable logic (V_B1_TX_EN

and V_B2_TX_EN), it is important always to enable the B-channels during U_p operation even if no data is send.

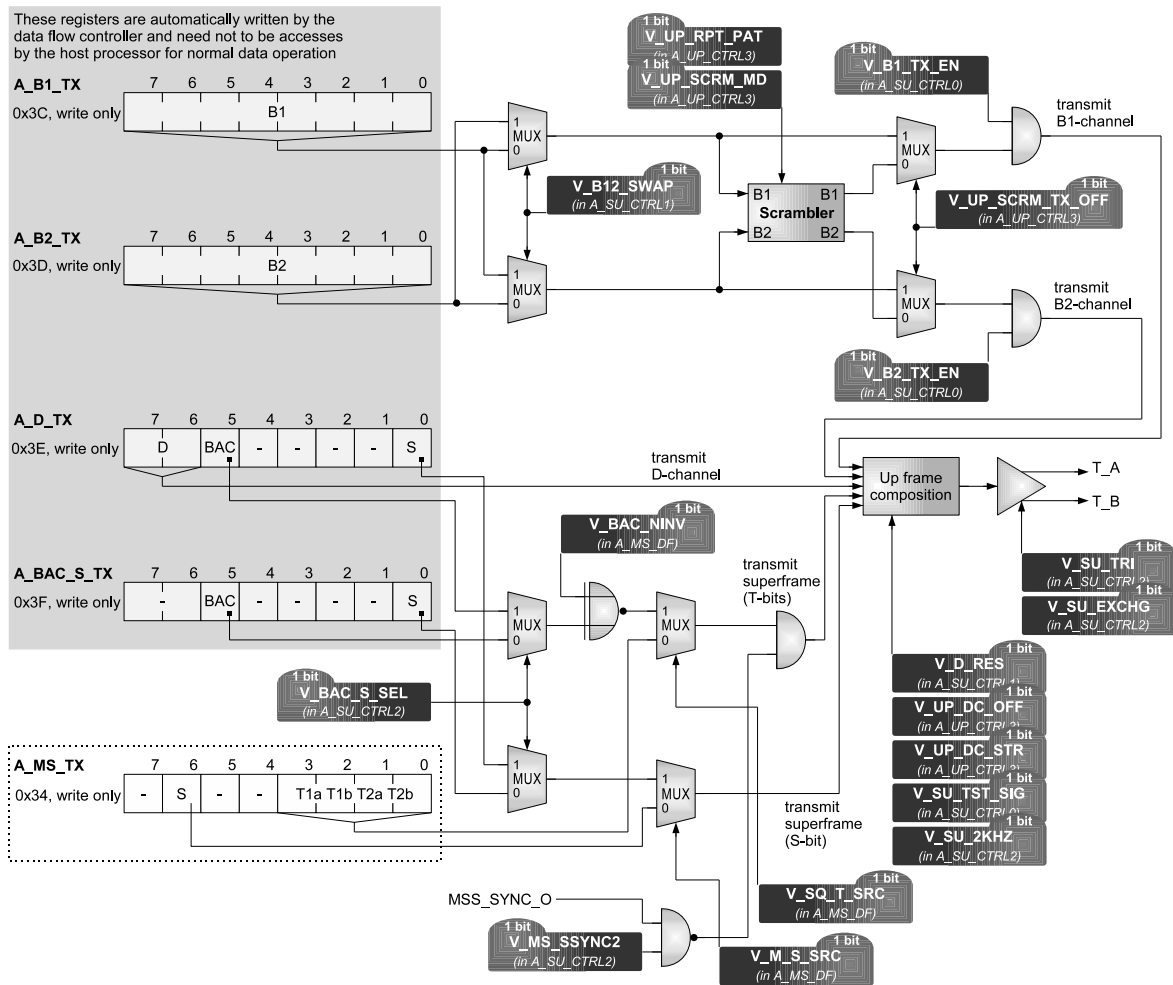


Figure 5.11: U_p frame composition for B1-, B2-, D-channel and superframe bits (U_p interface mode, transmit direction)

Figures 5.11 and 5.12 show the composition and decomposition of the U_p frames. B1-, B2- and D-channel data is normally handled by the data flow controller. The HDLC controller as well as the PCM interface deliver data to the U_p interface and receive data from the U_p interface. For this reason, registers A_B1_TX, A_B2_TX, A_D_TX and A_BAC_S_TX as well as A_B1_RX, A_B2_RX, A_D_RX and A_E_RX⁵ are normally not written or read from the application software.

Superframe bits can be handled by the data flow controller (involving registers A_BAC_S_TX and A_E_RX) or manually from the application software with registers A_MS_TX and A_MS_RX. Please note that A_BAC_S_TX contains E-channel data only in S/T interface mode.

5.3.5 INFO signals

Signals which are transmitted on the interface line are called *INFO signals*. INFO0 is defined for upstream (TE-to-LT) and downstream (LT-to-TE) direction. All other INFO signals are either for

⁵Please note, that A_E_RX does only contain E-channel bits in S/T interface mode.

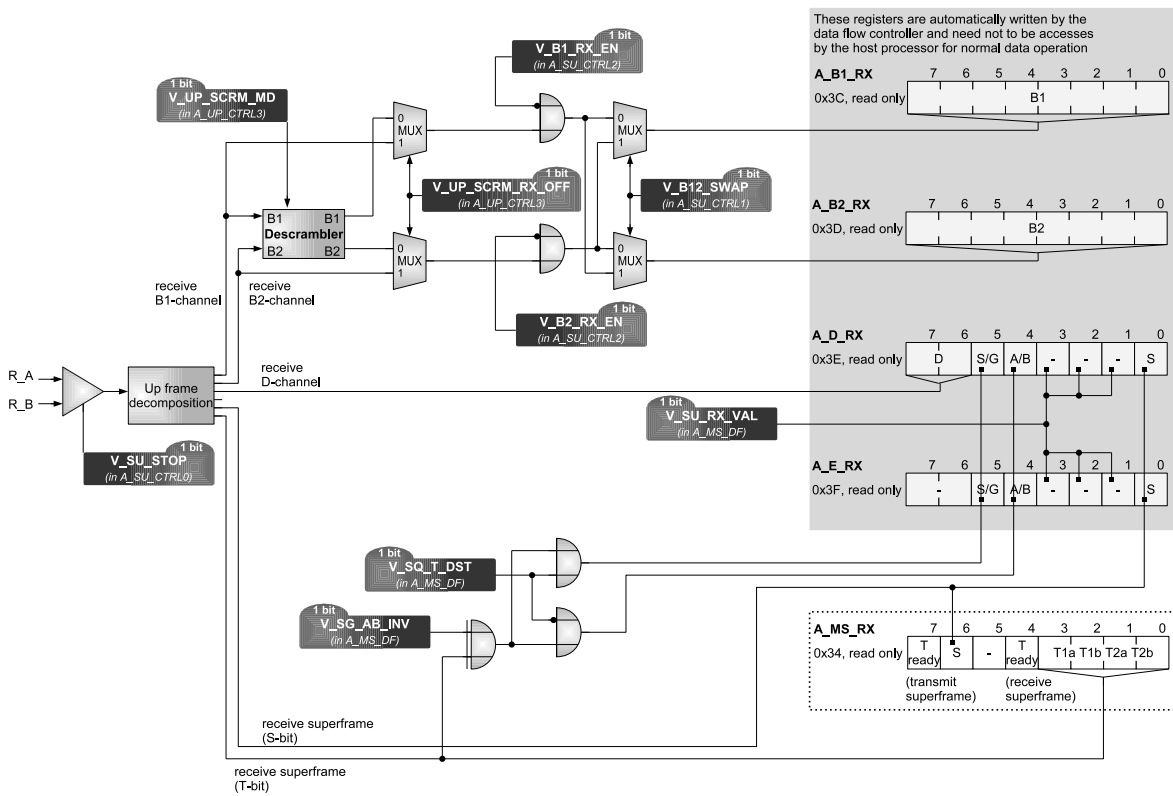


Figure 5.12: U_p frame decomposition for B1-, B2-, D-channel and superframe bits (U_p interface mode, receive direction)

upstream (INFO 1W, INFO 1, INFO 3) or downstream (INFO 2, INFO 4) direction. The INFO signals are defined as follows [3]:

INFO 0: No signal on line.

INFO 1W: Asynchronous awake signal with a 2 kHz burst rate (every second frame used). The frame has the contents

```
'F 00010001 00010001 0001 01010100 01011111 1'
```

with code violation in the framing bit F. F is always '1'.

INFO 1: 4 kHz burst rate (every frame used). The frame has the contents

```
'F 00010001 00010001 0001 01010100 01011111 M DC'
```

with code violation in the framing bit F.

INFO 2: 4 kHz burst rate (every frame used). The frame has the contents

```
'F 00010001 00010001 0001 01010100 01011111 M'
```

with code violation in the framing bit F.

INFO 3: 4 kHz burst rate (every frame used) with user data in B-, D- and M-channels. The B-channels are scrambled. The framing bit F has no code violation. The DC-bit is used.

INFO 4: 4 kHz burst rate (every frame used) with user data in B-, D- and M-channels. The B-channels are scrambled. The framing bit F has no code violation. The DC-bit is used.

The F-bit polarity (AMI-violation or no AMI-violation) is calculated in relation to the last bit of the preceding frame in the same direction. The DC-balancing bit is included in the F-bit polarity calculation algorithm when it is present.

5.3.6 State machine

A specification conform state machine for TE and LT mode is implemented. So the current Fx or Gx state of the state machine can be read out of register A_SU_RD_STA. However, it is possible to overwrite the state machine by setting bit V_SU_LD_STA in register A_SU_WR_STA.

Activation and deactivation can be initiated by writing bitmap V_SU_ACT in the same register. This bitmap can be used for TE and LT mode and can start activation or deactivation from any state. Even in TE mode it can be used to initiate a deactivation from any state to F3. Such a deactivation should only be initiated if the state machine is not in F6 or F7, of course. Writing '11' (start activation) when the state machine is already activated (G2/G3 or F6/F7), will not change the current state.

Before starting the state machine in TE mode, register A_SU_CLK_DLY of its U_p interface must be set. The default value is 0xF for TE mode.

Please note that in contrast to the S/T interface mode, an U_p device cannot be linked to an already activated U_p line for monitoring e.g., because the device must pass the entire activation sequence.

Tables 5.9 and 5.10 show the U_p interface activation and deactivation layer 1 of the finite state matrix in TE and LT mode. They are adopted from the S/T state machine specification according to ITU-T I.430 [9].

Table 5.9: U_p interface activation/deactivation layer 1 matrix for LT mode

State name:	Reset	Deactivated	Pending activation	Active	Pending deactivation
State number:	G 0	G 1	G 2	G 3	G 4
INFO sent:	INFO 0	INFO 0	INFO 2	INFO 4	INFO 0
Event:					
State machine release ^{*3}	G 1				
Activate request	start T 1 ^{*1} G 2	start T 1 ^{*1} G 2			start T 1 ^{*1} G 2
Deactivate request	—		start T 2 G 4	start T 2 G 4	
Expiry T 1 ^{*1}	—	—	start T 2 G 4	/	—
Expiry T 2 ^{*2}	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1 or INFO 1W	—	start T 1 ^{*1} G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 ^{*1,4} G 3	—	—
Lost framing	—	/	/	G 2	—
Legend:	—	No state change			
	/	Impossible situation			
		Impossible by the definition of the layer 1 service			

^{*1}: Timer T 1 is not implemented and must be implemented in software. T 1 is started with entering G2, runs during G2 state and is stopped when entering G3 or expiry. T 1 should expire after 100 ms . . . 1000 ms [5].

^{*2}: Timer T 2 prevents unintentional reactivation. Its value is $256 \cdot 125\mu s = 32ms$. This implies that a TE has to recognize INFO 0 and to react on it within this time.

^{*3}: After reset the state machine is fixed to G 0.

^{*4}: Bit V_SU_SET_G2_G3 in register A_SU_WR_STA must be set to allow this transition or V_G2_G3_EN in register A_SU_CTRL1 must be set to allow automatic transition G 2 → G 3.

Table 5.10: U_p interface activation/deactivation layer 1 matrix for TE mode

State name:	Reset	Sensing	Deactivated	Awaiting signal	Identifying input	Synchronized	Activated	Lost framing
State number:	F0	F2	F3	F4	F5	F6	F7	F8
INFO sent:	INFO 0	INFO 0	INFO 0	INFO 1W	INFO 0	INFO 1	INFO 3	INFO 0
Event:								
State machine release ^{*1}	F2	/	/	/	/	/	/	/
Activate request, receiving any signal	—		F5			—		—
receiving INFO 0	—		start T3 ^{*5} F4			—		—
Expiry T3 ^{*5}	—	/	—	F3	F3	—	—	F3
Receiving INFO 0	—	F3	—	—	—	F3	F3	F3
Receiving any signal ^{*2}	—	—	—	F5	—	/	/	—
Receiving INFO 2 ^{*3}	—	F6	F6	F6	F6	—	F6	F6
Receiving INFO 4 ^{*3}	—	F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	stop T3 ^{*5} F7	—	stop T3 ^{*5} F7
Lost framing ^{*4}	—	/	/	/	/	F8	F8	—

Legend: — No state change
 / Impossible situation
 | Impossible by the definition of the layer 1 service

^{*1}: After reset the state machine is fixed to F0.

^{*2}: This event reflects the case where a signal is received and the TE has not (yet) determined whether it is INFO 2 or INFO 4.

^{*3}: Bit and frame synchronization achieved.

^{*4}: Loss of Bit or frame synchronization.

^{*5}: Timer T3 is not implemented and must be implemented in software.

**Important !**

The U_p state machine is stuck at F0 or G0 after a reset. The interface sends no signal on the U_p line and is not able to activate it by incoming INFO x in this state. Writing '0' into bit V_{SU_LD_STA} of register A_{SU_WR_STA} starts the state machine.

LT mode: The LT state machine does not change automatically from G2 to G3 if the TE side sends INFO 3 frames. This transition must be activated each time by V_{G2_G3} in register A_{SU_RD_STA} or it can permanently be activated by setting bit V_{G2_G3_EN} in register A_{SU_CTRL1}.

5.3.7 Clock synchronization

A detailed view inside the line interface block diagram of Figure 5.1 is shown for the U_p interface mode in Figure 5.13. All clocks are derived from the 12.288 MHz clock f_{SU} . Frame synchronization is accomplished by evaluating the code violations in the U_p frame.

Received data from the pins R_{A0}/R_{A1} and R_{B0}/R_{B1} is passed through the RX data path to the switching buffer. A bit clock and a frame clock are derived from the received data stream. These clocks are used to synchronize the RX data path timing to the incoming data stream. The frame clock can be passed for synchronization purposes to the TX data path and the PCM timing control as well.

The transmit data clock has different sources in TE and LT mode:

LT mode: The 384 kHz bit clock as well as the 8 kHz frame clock are derived from FSYNC in NT mode. This signal is either F0IO input or F1_7 (see register R_{SL_SEL7} and Figure 6.5 on page 229).

TE mode: A TE is always taken as synchronization source for ISDN applications because it delivers the clock from the central office switch. Thus both clocks are taken from the RX clock unit.

The state machine takes several signals from the RX data path and the RX clock unit. The TX data path is controlled by the state machine's output signal.

Bit scramblers are inserted into the receive and the transmit data paths. The scramblers can be switched off with V_{UP_SCRM_RX_OFF} = '1' (for the receive data path) and V_{UP_SCRM_TX_OFF} = '1' (for the transmit data path) in register A_{UP_CTRL3}. Bit scramblers should only be disabled for test purposes. They shall mandatorily be enabled for normal U_p operation.

The scrambler mode is configurable according to ITU-T V.27 specification [8] or OCTAT-P compatibility [6] with V_{UP_SCRM_MD} in the same register.

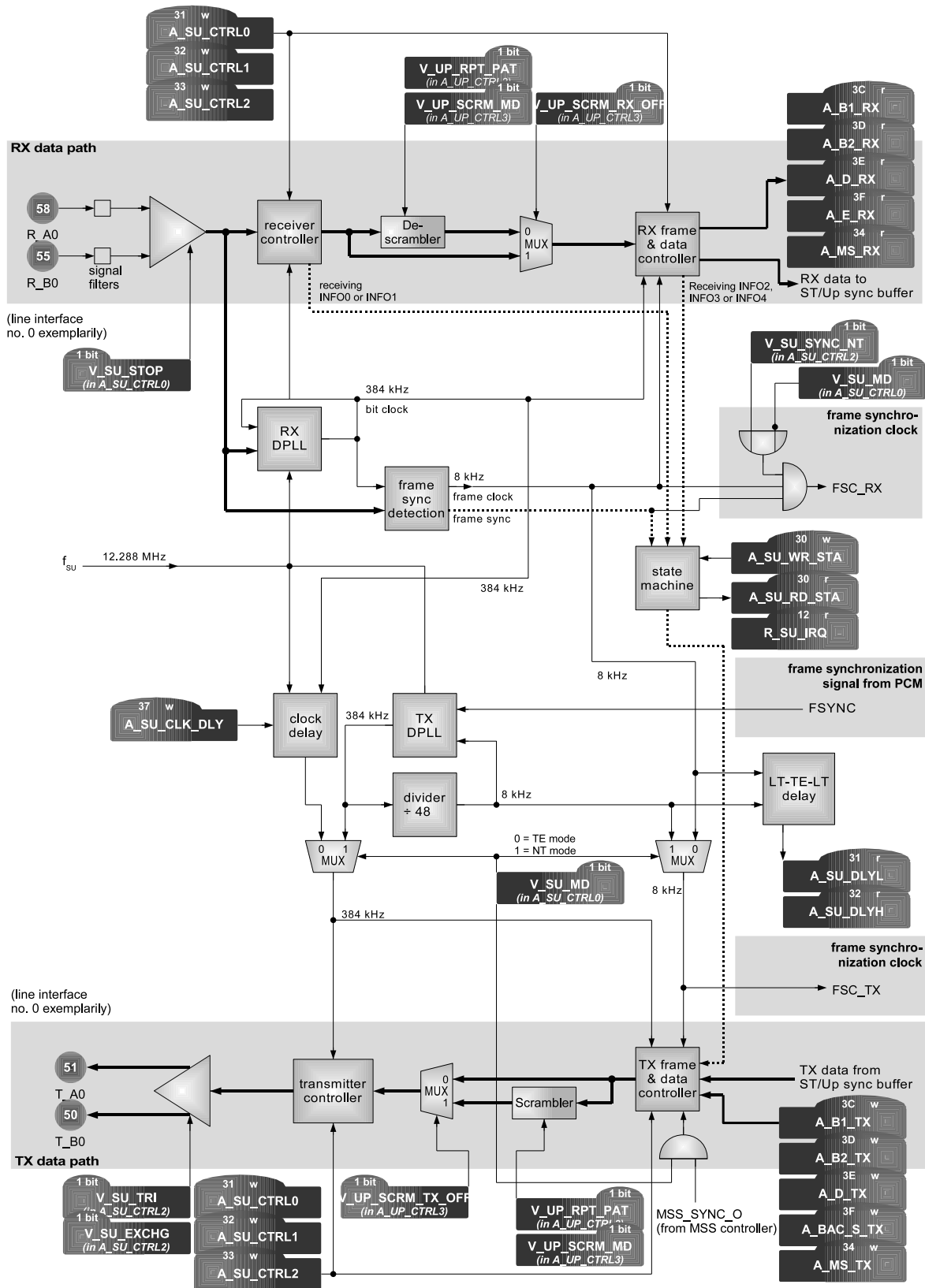


Figure 5.13: U_p clock synchronization

5.3.8 External circuitry

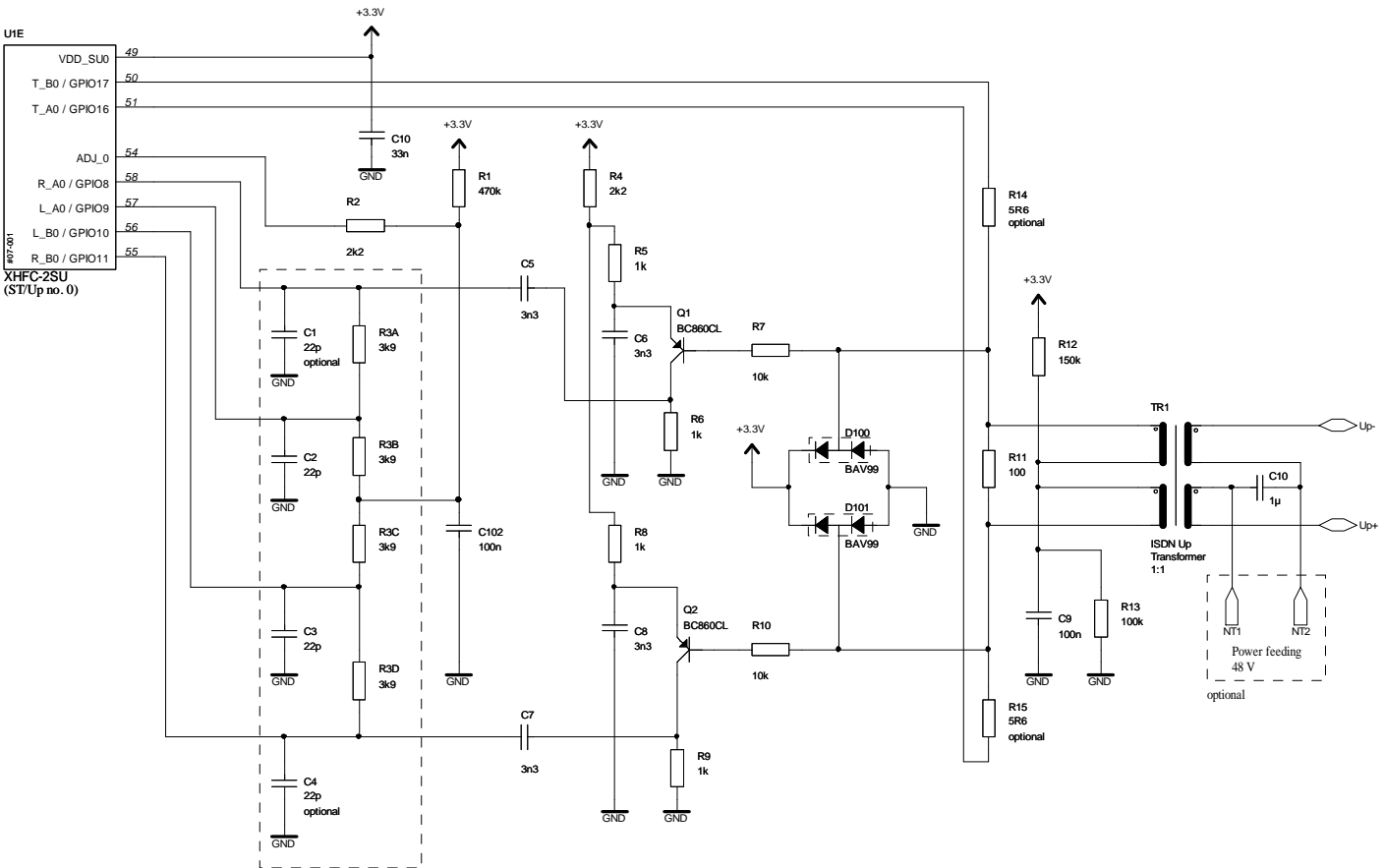


Figure 5.14: External U_p circuitry for TE and LT mode

C1..C4 must be located as close as possible to the input pins of the microchip.
R3 is recommended to be located near to the chip.

Q1, Q2:
BC860CL or equivalent C type

R14, R15:
Value depends on the used Up transformer
and the desired Up amplitude.

5.3.9 U_p transformers

Customers of Cologne Chip can choose from a variety of U_p transformers for two-wire ISDN U_p interface (U_{p0}/U_{pN}). All U_p transformers are compatible to the XHFC series of Cologne Chip that fulfill two criteria:

- Turns ratio of 1:1
- Dual winding on line side (required for power feeding)

Several companies provide transformers that can be used with our ISDN Basic Rate Interface controllers. Part numbers and manufacturers are listed in Table 5.11. Please ask our support team for more information.

The transformer list has not been compiled under aspects of RoHS compliance. For the current RoHS status of the listed parts, please contact the transformer manufacturers straight.

Table 5.11: *Up transformer part numbers and manufacturers*

UMEC GmbH, Germany, Taiwan, United States, <http://www.umec.de>

Type Device

Single transformer:

UT 21434A-TS (SMD)

Vacuumschmelze GmbH & Co. KG, Germany, <http://www.vacuumschmelze.com>

Type Device

Single transformer:

3-M5024-X008 (SMD)

Sumida AG, Germany, <http://www.sumida-eu.com> (formerly known as Vogt electronic AG)

Type Device

Single transformer

503 10 903 00 (SMD)

Please note: Cologne Chip cannot take any liability concerning the product names, characteristics and availability. Products can change without notice. Please refer to the manufacturer in case of doubt.

5.4 Common features of the S/T and U_p interfaces

5.4.1 Direct data access for test purposes

Data accesses from the host processor are normally write operations to the FIFOs or read operations from the FIFOs. For test purposes it is also possible to access directly internal data registers. These registers

- A_B1_TX, A_B2_TX, A_D_TX and A_BAC_S_TX in transmit data direction and
- A_B1_RX, A_B2_RX, A_D_RX and A_E_RX in receive data direction

can only be read or written during non-processing phase of the data flow processor. This is indicated with V_PROC = '0' in register R_STATUS. During processing phase (V_PROC = '1'), read values can be invalid and written values might be overwritten by the data flow processor.

Interrupt capability

Every line interface can cause an interrupt when a state change occurs. State changes can be read from V_SU0_IRQ..V_SU1_IRQ in register R_SU_IRQ for each line interface separately. Both bits in R_SU_IRQ are cleared after reading the register.

Any '1' in these bits causes an interrupt if the associated mask bit V_SU0_IRQMSK..V_SU1_IRQMSK in register R_SU_IRQMSK enables the interrupt. Register R_SU_IRQ can be read even if interrupts are disabled with V_SU0_IRQMSK..V_SU1_IRQMSK = '00' and must be polled to watch the interrupt status in this case.

5.4.2 Clock synchronization with several TEs connected to different CO switches

Both TEs of the XHFC-2SU line interfaces can be interconnected to different central offices (CO). These have different clock phases and, typically, slightly different clock frequencies. An example of this scenario is illustrated in Figure 5.15. XHFC-2SU is able to synchronize both line interfaces as it is described in this section.

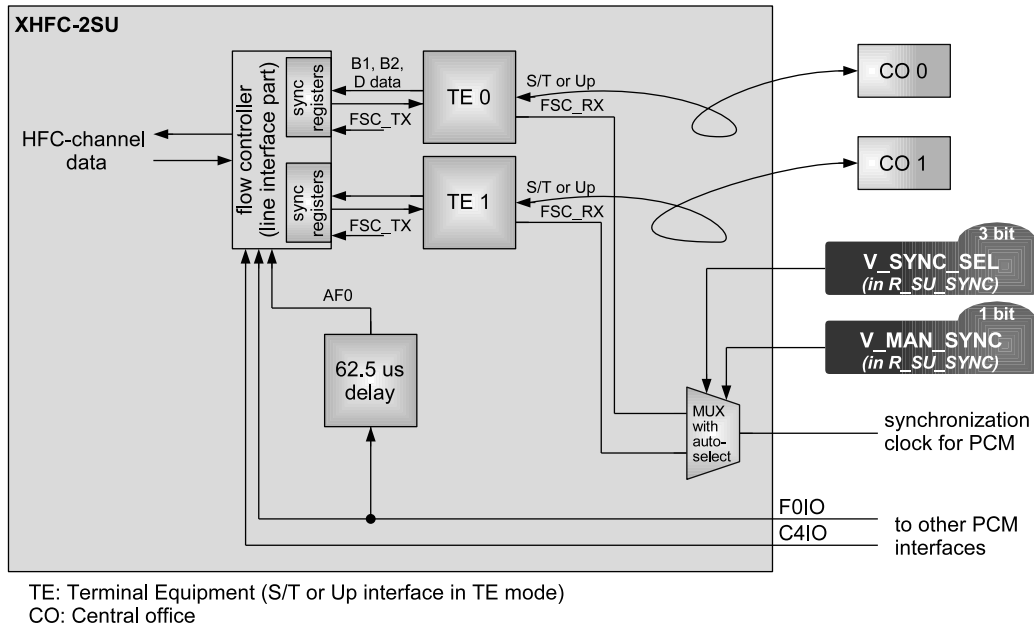


Figure 5.15: Synchronization scenario with TEs connected to several central office switches

The synchronization registers of Figure 5.15 are shown in detail in Figure 5.16. Received and transmitted data is always buffered twice to achieve a synchronization on both sides, the HFC-channel and the line interface. The line interface data is always synchronous to its FSC pulse.

HFC-channel data is latched either by the F0IO signal or by the delayed AF0 signal. If there is a central clock supply from an external PLL, it can be used to provide the timing for XHFC-2SU as shown in Figure 5.15. In the other case, the internal PLL can be used as master PLL of the ISDN system.

The window detection block changes its output signal when the phase offset between FSC_TX and F0 is smaller than approximately 25 μ s (guard window). So the phase offset between FSC_TX and F0 is always 25 μ s .. 100 μ s.

Timing without frequency drift

The timing characteristics of two TEs with a phase offset and the signals F0IO and AF0 are shown in Figure 5.17. In this example TE 0 is synchronization source for the PLL. Thus the timing offset between FSC_TX_0 and F0IO is 62.5 μ s (caused by the PLL). The figure shows one sample transmit data flow and one sample receive data flow on TE 0 and TE 1 each. In fact, both data transmissions happen every 125 μ s.

Figure 5.17 is divided into three parts. The upper and lower part show the line interface oriented signals of TE 0 and TE 1 respectively. In the middle part, HFC-channel oriented signals are shown

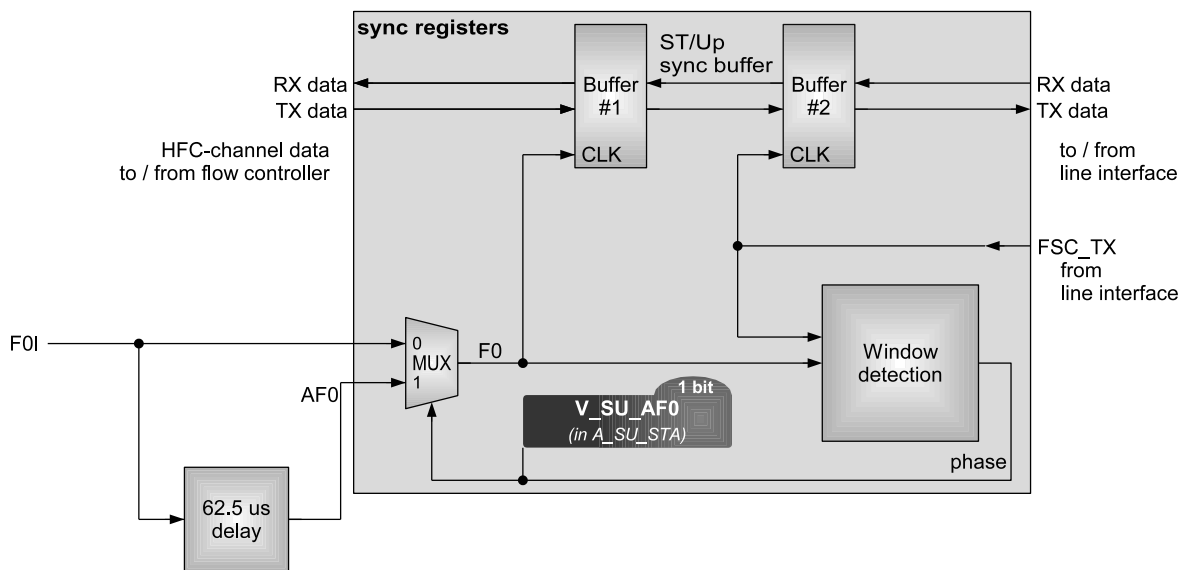


Figure 5.16: Synchronization registers (detail of Figure 5.15)

Table 5.12: Symbols of Figures 5.17

Symbol	Characteristic
t_{PLL}	PLL-generated frame pulse offset between FSC_TX_0 and F0IO (62.5 μ s)
t_{delay}	Frame pulse delay between F0IO and AF0 (62.5 μ s)
t_{phase}	Frame offset between interface TE 0 and TE 1
TX _{data_F0_0}	Time from transmit data valid to next F0_0 pulse
TX _{F0_0_FSC0}	Time from F0_0 pulse to next FSC_TX_0 pulse
RX _{FSC0_F0_0}	Time from FSC_TX_0 pulse to next F0_0 pulse
RX _{F0_0_data}	Time from F0_0 pulse to receive data valid
TX _{data_F0_1}	Time from transmit data valid to next F0_1 pulse
TX _{F0_1_FSC1}	Time from F0_1 pulse to next FSC_TX_1 pulse
RX _{FSC1_F0_1}	Time from FSC_TX_1 pulse to next F0_1 pulse
RX _{F0_1_data}	Time from F0_1 pulse to receive data valid

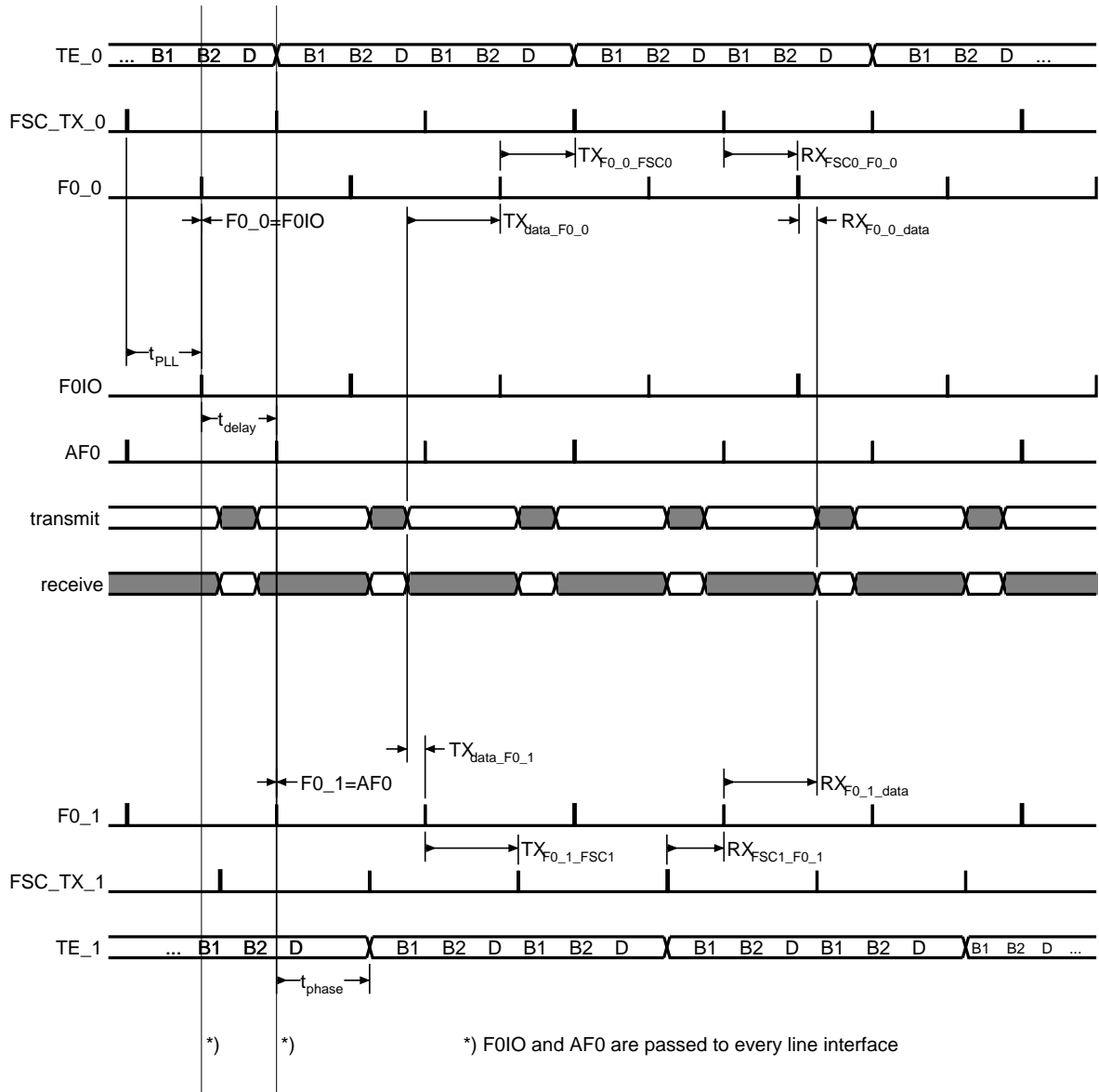


Figure 5.17: Transmit and receive data transmission examples of two TEs with phase offset (see explanation in the text)

which are common for both line interfaces.

A PLL generates the F0IO signal from the FSC pulse of the synchronization source with $t_{PLL} = 62.5\mu s$. AF0 has a fixed $62.5\mu s$ delay to F0IO. The pseudo signals *transmit* and *receive* in Figure 5.17 represent the valid and invalid states of the HFC-channel data, strictly speaking input data of buffer #1 in transmit direction and output data from buffer #1 in receive direction.

As TE0 is the synchronization source in this example, FSC_TX_0 is the reference signal for the PLL to generate the F0IO signal. A data transmission from TE0 has no choice of synchronization signals. Buffer #1 gets the data byte with a F0IO pulse and buffer #2 takes it with the next FSC_TX_0 pulse.

In receive direction, the incoming data byte is stored in buffer #2 first with the FSC_TX_0 pulse. After $RX_{FSC0_F0_0}$, buffer #1 takes the data byte where it becomes valid for the HFC-channel.

TE1 gets the transmit data from the HFC-channel with the F0_1 pulse which comes $TX_{data_F0_1}$ after data became valid. The internal data transfer of each ST/U_p interface is controlled either by F0IO or by AF0. In this example $F0_1 = AF0$ is shown. $TX_{F0_1_FSC1}$ after F0_1, the FSC_TX_1 pulse stores the data in buffer #2 so that the data byte is available at the line interface.

Received data is first stored in buffer #2 with the FSC_TX_1 pulse. After $RX_{FSC1_F0_1}$ buffer #1 takes the data byte and it receives the HFC-channel.

For the TE which acts as synchronization source, the clock pulses of buffer #1 and #2 have always a $62.5\mu s$ delay. Unsynchronized ST/U_p interfaces have clock pulses of buffer #1 and #2 that are delayed $25\mu s \dots 100\mu s$. The value depends on the phase offset t_{phase} between the synchronization source and the unsynchronized interface.

Timing with frequency drift

When there is a frequency drift between FSC_TX_0 and FSC_TX_1, the window detection block changes its output level from time to time and the synchronization multiplexer output shown in Figure 5.16 switches to the other clock signal. When this happens, a data error might happen.

Figure 5.18 shows the synchronization process for $f_{FSC_TX_1} > f_{FSC_TX_0}$ in transmit data direction. FSC_TX_0 is assumed to be the synchronization signal which is the source for F0IO. F0_1 is either F0IO or AF0. In this case FSC_TX_1 is too fast which leads to a *byte doubling* in case of transmission error.

Every time, when the detection window reaches the FSC_TX_1 pulse, F0_1 jumps to the alternative signal. Every second jump a data error occurs as shown with byte 3 which is transmitted twice.

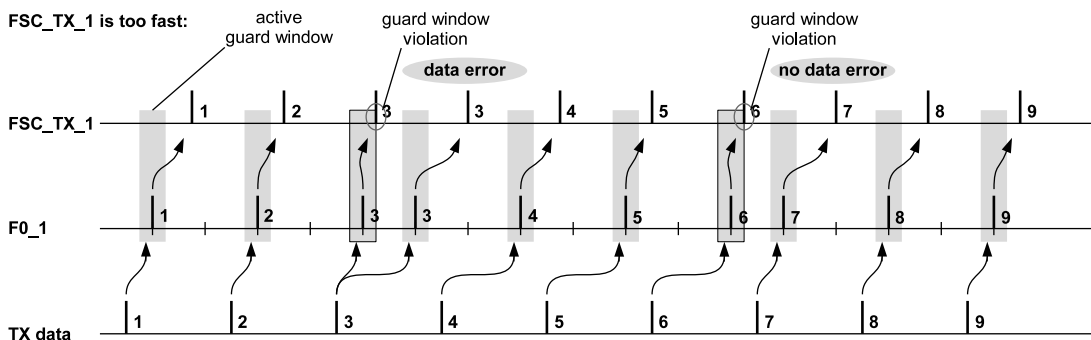


Figure 5.18: Data transmission with $f_{FSC_TX_1} > f_{FSC_TX_0}$ (i.e. too fast FSC_TX from unsynchronized TE)

Figure 5.19 shows the synchronization process for $f_{FSC_TX_1} < f_{FSC_TX_0}$ in transmit data direction. Again, FSC_TX_0 is assumed to be the synchronization signal. In this case FSC_TX_1 is too slow which leads to a *byte skip* in case of transmission error.

Every time, when the detection window reaches the FSC_TX_1 pulse, F0_1 jumps to the alternative signal. Every second jump an error occurs as shown with the byte pair 3 and 4, where byte 3 is not transmitted.

The shown examples consider only the transmit data direction. A similar effect exists in receive data direction, of course. A too fast FSC_TX_1 leads to *byte skip* errors and a too slow FSC_TX_1 causes *byte doubling* errors from time to time.

The time between two errors is given by

$$T_{error} = \frac{1}{f_{FSC_TX_0}} \cdot \frac{1 + \Delta f_{rel}}{\Delta f_{rel}} \approx \frac{125 \mu s}{\Delta f_{rel}} \text{ for } \Delta f_{rel} \ll 1$$

with the precise frame clock $f_{FSC_TX_0} = 8 \text{ kHz}$ and the relative frequency error

$$\Delta f_{rel} = \frac{|f_{FSC_TX_1} - f_{FSC_TX_0}|}{f_{FSC_TX_0}}$$

For example, with $\Delta f_{rel} = 0.01 \text{ ppm} = 10^{-8}$ the error-to-error time is $T_{error} = 208 \text{ minutes}$.

Frequency jitter

Even if both TEs have exactly the same frequency, there might be a F0-jump as well. Due to FSC jitter, the synchronization multiplexer can switch to the alternative signal. But this will happen only one time. Then, the guard window is centered between two consecutive FSC pulses and is far away from another jump condition. This single-jump condition might cause a byte error or not. It depends on the question, which one of the four jump situations shown in Figures 5.18 and 5.19 occurs and therefore it is a random event.

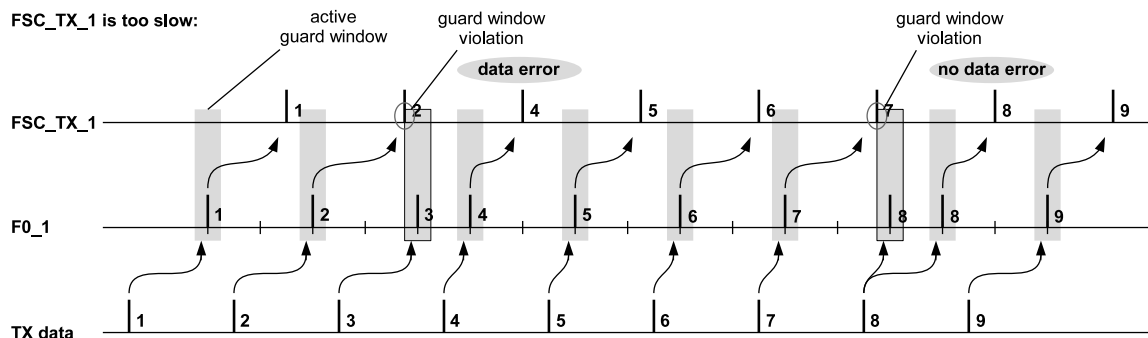


Figure 5.19: Data transmission with $f_{FSC_TX_1} < f_{FSC_TX_0}$ (i.e. too slow FSC_TX from unsynchronized TE)

5.4.3 Combined S/T and U_p circuitry

An external circuitry of the Universal ISDN Port can be set up which can be used for both S/T and U_p interface mode. This circuitry requires only one S/T transformer module which is used for U_p operating mode as well. The circuitry description is available on request.

5.5 Register description

 **Please note !**

The name fragment SU of registers and bitmaps indicates those registers and bitmaps which are valid in both S/T and U_p interface mode. SU means 'ST/U_p'.

The name fragments ST or UP are used when a register or bitmap is either valid in S/T or U_p interface mode.

5.5.1 Write only registers

R_SU_SEL	(w)	(Reset group: H, 0, 3)	0x16
S/T or Up interface selection register			
This register is used to select an S/T or Up interface. Before a line interface array register can be accessed, this index register must specify the desired line interface number.			
Bits	Reset value	Name	Description
1..0	0	V_SU_SEL	Single line interface selection '000' = ST/Up interface 0 '001' = ST/Up interface 1 '010' = ST/Up interface 2 '011' = ST/Up interface 3
2	0	(reserved)	Must be '0'.
3	0	V_MULT_SU	multiple line interface selection All line interfaces are selected together. This is only useful for write access. '0' = interface selection by V_SU_SEL '1' = select all line interfaces for write accesses
7..4	0	(reserved)	Must be '0000'.

A_SU_WR_STA [ST/Up] (w) (Reset group: H, 0, 3) **0x30**

ST/Up state machine register

This register is used to set a new state. The current state can be read from register A_SU_RD_STA.

Before writing this array register the line interface must be selected by register R_SU_SEL.

Bits	Reset value	Name	Description
3..0	0	V_SU_SET_STA	Binary value of the new state (NT/LT: Gx, TE: Fx) V_SU_LD_STA must also be set to load the state.
4	1	V_SU_LD_STA	Load the new state '1' = load the prepared state (V_SU_SET_STA) and stops the state machine. This bit needs to be set for a minimum period of 5.21 µs and must be cleared by software. '0' = enable the automatic state machine (V_SU_SET_STA is ignored). Note: After writing an invalid state, the state machine goes to deactivated state (G1, F2).
6..5	0	V_SU_ACT	Start activation/ deactivation '00' = no operation '01' = no operation '10' = start deactivation '11' = start activation These bits are automatically cleared after reaching the activated / deactivated state.
7	0	V_SU_SET_G2_G3	Allow G2 to G3 transition '0' = no operation '1' = allows transition from G2 to G3 in NT/LT mode This bit is automatically cleared after the transition and has no function in TE mode.

Bits	Reset value	Name	Description
A_SU_CTRL0 [ST/Up] (w) (Reset group: H, 0, 3) 0x31			
Control register of the selected line interface, register 0			
Before writing this array register the line interface must be selected by register R_SU_SEL.			
0	0	V_B1_TX_EN	B1-channel transmit '0' = B1 send data disabled (permanent '1's sent when the line interface is activated) '1' = B1 send data enabled Note: When the bit scrambler is activated in Up line interface mode (default operation mode), this bit must be set to '1' even if the B1-channel is not in use.
1	0	V_B2_TX_EN	B2-channel transmit '0' = B2 send data disabled (permanent '1's sent when the line interface is activated) '1' = B2 send data enabled Note: When the bit scrambler is activated in Up line interface mode (default operation mode), this bit must be set to '1' even if the B2-channel is not in use.
2	0	V_SU_MD	Line interface mode '0' = TE mode '1' = NT/LT mode
3	0	V_ST_D_LPRI0	D-channel priority '0' = high priority 8/9 '1' = low priority 10/11 Note: This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
4	0	V_ST_SQ_EN	S/Q bits transmission '0' = S/Q bits disabled '1' = S/Q bits (multiframe) enabled Note: This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
5	0	V_SU_TST_SIG	Send test signal '0' = normal operation '1' = send test signal The test signal depends on the selected line interface mode and V_SU_2KHZ in register A_SU_CTRL2: S/T interface mode: Test signal is 96 kHz (alternating '0's) or 2 kHz (one alternating '0' per frame) Up interface mode: Test signal is 192 kHz (alternating '1's) or 2 kHz (one alternating '1' per frame)

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
6	0	V_ST_PU_CTRL	End of pulse control The end of pulse edge of the transmit signal can be adjusted in S/T interface mode. The programming value of V_ST_PULSE in register A_ST_CTRL3 specifies the end of pulse edge. '0' = no end of pulse control '1' = end of pulse control enabled Note: This bit is only used for line interfaces in S/T mode. It is not used in Up interface mode and must be '0'.
7	0	V_SU_STOP	Power down '0' = external receiver activated '1' = power down, external receiver disabled

Bits	Reset value	Name	Description
A_SU_CTRL1 [ST/Up] (w) (Reset group: H, 0, 3) 0x32			
Control register of the selected line interface, register 1			
Before writing this array register the line interface must be selected by register R_SU_SEL.			
0	0	V_G2_G3_EN	Force automatic transition from G2 to G3 '0' = V_SU_SET_G2_G3 in register A_SU_WR_STA must be set again for every transition from G2 to G3 '1' = transitions from G2 to G3 are always allowed and V_SU_SET_G2_G3 is ignored
1	0	(reserved)	Must be '0'.
2	0	V_D_RES	D-channel reset '0' = normal operation '1' = D-channel is reset and its bits are forced to '1' in transmit direction
3	0	V_ST_E_IGNO	Ignore E-channel data This bit is only used for line interfaces in S/T mode. D-channel data is immediately send in Up interface mode. '0' = normal operation '1' = D-channel always sends data regardless of the received E-channel bit Note: This bit is only used in TE mode and is ignored in NT mode.
4	0	V_ST_E_LO	Force E-channel to low (only in NT mode) '0' = normal operation, E-channel bits echo received D-channel data '1' = E-channel bits are forced to '0' Note: This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
5	0	(reserved)	Must be '0'.
6	0	V_BAC_D	BAC-bit disables D-channel transmit '0' = D-channel transmission is enabled '1' = Bac-bit is used to enable or disable D-channel transmission Note: This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
7	0	V_B12_SWAP	Swap B-channels '0' = normal operation '1' = swap B1- and B2-channel registers of the line interface

A_SU_CTRL2 [ST/Up]

(w)

(Reset group: H, 0, 3)

0x33

Control register of the selected line interface, register 2

Before writing this array register the line interface must be selected by register R_SU_SEL.

Bits	Reset value	Name	Description
0	0	V_B1_RX_EN	Enable B1-channel receive '0' = B1 receive bits are forced to '1' '1' = normal operation
1	0	V_B2_RX_EN	Enable B2-channel receive '0' = B2 receive bits are forced to '1' '1' = normal operation
2	0	V_MS_SSYNC2	Multiframe / superframe single synchronization pulse The multiframe / superframe synchronization pulse can be configured to occur only once when this bit is set to '1'. S/T interface mode: '0' = normal multiframe bit M is transmitted (once every 20 S/T frames) '1' = single multiframe bit can be transmitted Up interface mode: '0' = normal T-bit is transmitted in Up superframe '1' = single T-bit = '0' can be transmitted at T-bit position in the Up superframe (normal T-bit sources must be '1') Note: When this bit is set to '1', usually also V_MS_SSYNC1 has to be set in register R_MSS1.
3	0	V_BAC_S_SEL	Multiframe / superframe bit source selection This bit has different meaning for line interfaces in S/T or Up mode: S/T interface mode: Source selection of multiframe S/Q-bits and multiframe bit M Up interface mode: Source selection of superframe S- and T-bits '0' = bits come from register A_BAC_S_TX '1' = bits come from D-channel register A_D_TX Note: Further source selection is available with register A_MS_DF.
4	0	V_SU_SYNC_NT	8 kHz synchronization pulses of the line interface path are generated even in NT mode. '0' = pulses are only generated in TE mode '1' = pulses are generated in TE and NT mode

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
5	0	V_SU_2KHZ	Select test signal frequency '0' = transmit 96 kHz (S/T interface mode) or 192 kHz (Up interface mode) test signal '1' = transmit 2 kHz test signal The test signal must be switched on with V_SU_TST_SIG = '1' in register A_SU_CTRL0.
6	0	V_SU_TRI	Line interface output buffer tristated '0' = normal operation '1' = set output buffer into tristate mode (i.e. transmitter is switched off)
7	0	V_SU_EXCHG	Exchange transmit output buffers Change of the polarity of the line interface output pins. This is equal to an external crossing of the two transmit pins.

Bits	Reset value	Name	Description
A_MS_TX [ST/Up] (w) (Reset group: H, 0, 3) 0x34			
Multiframe/superframe transmit register			
Before writing this array register the line interface must be selected by register R_SU_SEL.			
3..0	0	V_MS_TX	Multiframe/superframe bits The meaning of this bitmap depends on the selected line interface mode: S/T interface mode: S/Q bits to be transmitted in the multiframe. Bits [3..0] are Q bits [Q1,Q2,Q3,Q4] in TE mode and S bits [S1,S2,S3,S4] in NT mode. Up interface mode: T bits to be transmitted in the superframe (M-channel). Bits [3..0] are T bits [T1a,T1b,T2a,T2b].
5..4	0	(reserved)	Must be '00'.
6	0	V_UP_S_TX	S-bit of the Up superframe S-bit to be transmitted in the superframe (M-channel). The S-bit can be read from registers A_D_TX or A_BAC_S_TX alternatively. Note: This bit is only used for line interfaces in Up interface mode. It must be '0' in S/T interface mode.
7	0	(reserved)	Must be '0'.

Bits	Reset value	Name	Description
0	0	V_ST_SEL	<p>Line interface mode selection This bit selects either the S/T or Up mode of the line interface. '0' = line interface is in S/T mode and multi-register A_ST_CTRL3 is selected '1' = line interface is in Up mode and multi-register A_UP_CTRL3 is selected Note: This bit should only be changed after reset.</p>
7..1	0	V_ST_PULSE	<p>End of pulse control The shape of the pulse end can be adjusted with this bitmap. V_ST_PULSE = 0x7C should be used.</p>

A_UP_CTRL3 [ST/Up] (w) (Reset group: H, 0, 3) **0x35**

Control register of the selected line interface, register 3

This register is an array register *and* a multi-register.

Before writing this array register the line interface must be selected by register R_SU_SEL. Afterwards the multi-register selection is required with bit V_UP_SEL = '1' in this register.

Note: This register is only used in Up interface mode. See register A_ST_CTRL3 when the line interface is in S/T mode.

Bits	Reset value	Name	Description
0	0	V_UP_SEL	<p>Line interface mode selection</p> <p>This bit selects either the S/T or Up mode of the line interface.</p> <p>'0' = line interface is in S/T mode and multi-register A_ST_CTRL3 is selected</p> <p>'1' = line interface is in Up mode and multi-register A_UP_CTRL3 is selected</p> <p>Note: This bit should only be changed after reset.</p>
1	0	V_UP_VIO	<p>Up activation after superframe violation</p> <p>'0' = activation is done even without a superframe violation</p> <p>'1' = activation is only done when a superframe violation is found</p>
2	0	V_UP_DC_STR	<p>DC-balancing mode</p> <p>'0' = The DC-balancing bit is only generated when a code violation (CV) has been sent in the M bit position (normal operation)</p> <p>'1' = The DC-balancing bit is always generated as parity bit to achieve a DC-free signal</p> <p>Note: DC-balancing must be enabled with V_UP_DC_OFF = '0' to generate DC-balancing bits</p>
3	0	V_UP_DC_OFF	<p>DC-balancing bit disabled</p> <p>'0' = normal operation</p> <p>'1' = The DC-balancing bit is always '0'</p>
4	0	V_UP_RPT_PAT	<p>Allow repeated patterns of scrambled data</p> <p>'0' = prevent repeated patterns as specified in bit V_UP_SCRM_MD (normal operation)</p> <p>'1' = repeated patterns are not prevented</p>
5	0	V_UP_SCRM_MD	<p>Scrambler / descrambler mode selection</p> <p>'0' = OCTAT-P mode (compatible to Siemens / Infineon Up microchips)</p> <p>'1' = V.27 mode (compatible to ITU-T V.27 specification)</p> <p>Note: V_UP_RPT_PAT must be '0' to prevent repeated patterns.</p>

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
6	0	V_UP_SCRM_TX_OFF	Transmit scrambler disabled The transmit data is not scrambled when this bit is set to '1'. The scrambler is bypassed in this case.
7	0	V_UP_SCRM_RX_OFF	Receive descrambler disabled The receive data is not descrambled when this bit is set to '1'. The descrambler is bypassed in this case.

A_MS_DF [ST/Up]		(w)	(Reset group: H, 0, 3)	0x36
Multiframe/superframe data flow configuration register				
Before writing this array register the line interface must be selected by register R_SU_SEL.				
Bits	Reset value	Name	Description	
0	0	V_BAC_NINV	No BAC-bit inversion '0' = invert BAC-bit '1' = no inversion of BAC-bit	
1	0	V_SG_AB_INV	Invert S/G or A/B bit '0' = normal operation '1' = invert S/G or A/B bit (depending on V_SQ_T_DST)	
2	0	V_SQ_T_SRC	Source of S/Q-bits (S/T multiframe) or T-bits (Up superframe) '0' = S/Q- or T-bits come from bitmap V_MS_TX in register A_MS_TX '1' = S/Q- or T-bits come from BAC-bit of D- or E-channel (depending on V_BAC_S_SEL)	
3	0	V_M_S_SRC	Source of M-bit (S/T multiframing bit) or S-bit (Up superframe) S/T interface mode: The meaning of this bit depends on the selected line interface mode: '0' = multiframing bit M comes from multiframe / superframe controller '1' = multiframing bit M comes from D- or E-channel (depending on V_BAC_S_SEL) Up interface mode: '0' = superframe bit S comes from bit V_UP_S_TX in register A_MS_TX '1' = superframe bit S comes from D- or E-channel (depending on V_BAC_S_SEL)	
4	0	V_SQ_T_DST	Destination of S/Q-bits (S/T multiframe) or T-bits (Up superframe) '0' = received bits are stored as S/G-bit in D- and E-channel registers '1' = received bits are stored as A/B-bit in D- and E-channel registers The unused destination bit is always '1'.	
5	0	V_SU_RX_VAL	Value of unused bits in received D- and E-channel The unused bits [3..1] in registers A_D_RX and A_E_RX are set to V_SU_RX_VAL.	
7..6	0	(reserved)	Must be '00'.	

A_SU_CLK_DLY [ST/Up]		(w)	(Reset group: H, 0, 3)	0x37
Clock control register of the line interface module				
This register is not initialized after reset. It must be initialized before activating the ST/Up state machine.				
Before writing this array register the line interface must be selected by register R_SU_SEL.				
Bits	Reset value	Name	Description	
3..0	0	V_SU_CLK_DLY	<p>Line interface clock delay TE mode (S/T or Up): 4 bit delay value to adjust the 2 bit delay between receive and transmit frame. The delay of the external line interface circuitry can be compensated. The lower the value the smaller the delay between receive and transmit direction. The suitable value is 0xE (S/T interface mode) or 0xF (Up interface mode) for normal external circuitries. NT mode (for S/T interface mode only): Data sample point. The lower the value the earlier the input data is sampled. The normal operation value is 0xC. LT mode (for Up interface mode only): This bitmap is not used when the line interface operates in Up mode.</p> <p>The steps are 163 ns for line interfaces in S/T mode and 81 ns for line interfaces in Up mode.</p>	
6..4	0	V_ST_SMPL	<p>Early edge input data shaping (NT mode only) Low pass characteristic of extended bus configurations can be compensated. The lower the value the earlier input data pulses are sampled. The default value is 6 ('110') which means that no compensation is carried out. Step size is 163 ns. Note: This bit is only used for line interfaces in S/T mode. It must be '0' in Up interface mode.</p>	
7	0	(reserved)	Must be '0'.	

A_B1_TX[ST/Up]	(w)	(Reset group: -)	0x3C
Transmit register for the B1-channel data			
This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.			
Before writing this array register the line interface must be selected by register R_SU_SEL.			
Bits	Reset value	Name	Description
7..0		V_B1_TX	B1-channel data byte Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

A_B2_TX[ST/Up]	(w)	(Reset group: -)	0x3D
Transmit register for the B2-channel data			
This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.			
Before writing this array register the line interface must be selected by register R_SU_SEL.			
Bits	Reset value	Name	Description
7..0		V_B2_TX	B2-channel data byte Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

A_D_TX [ST/Up]		(w)	(Reset group: -)	0x3E
Transmit register for the D-channel data				
<p>This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.</p> <p>Before writing this array register the line interface must be selected by register R_SU_SEL. Data can be written during the non-processing phase (see V_PROC in register R_STATUS).</p>				
Bits	Reset value	Name	Description	
0		V_D_TX_S	S-bit of the Up superframe The transmitted S-bit of the Up superframe is either taken from V_D_TX_S, V_S_TX or V_MS_TX. When the line interface is in S/T mode, this bit can be used as multiframing bit M.	
4..1		(reserved)	Must be '0000' when written.	
5		V_D_TX_BAC	BAC-bit The BAC-bit can be used as S/Q-bit (S/T multiframe) or T-bit (Up superframe). Alternatively, bitmap V_MS_TX in register A_MS_TX can be used for the multiframe/ superframe transmission.	
7..6		V_D_TX	D-channel data bits	

Bits	Reset value	Name	Description
A_BAC_S_TX [ST/Up] (w) (Reset group: -) 0x3F			
BAC-bit and S-bit for multiframe / superframe transmission			
This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.			
Before writing this array register the line interface must be selected by register R_SU_SEL. Data can be written during the non-processing phase (see V_PROC in register R_STATUS).			
0		V_S_TX	S-bit of the Up superframe The transmitted S-bit of the Up superframe is either be taken from V_D_TX_S, V_S_TX or V_D_TX_S. When the line interface is in S/T mode, this bit can be used as multiframe M-bit.
4..1		(reserved)	Must be '0000' when written.
5		V_BAC_TX	BAC-bit The BAC-bit can be used as S/Q-bit (S/T multiframe) or T-bit (Up superframe). Alternatively, bitmap V_MS_TX in register A_MS_TX can be used for the multiframe / superframe transmission.
7..6		(reserved)	Must be '00' when written.

5.5.2 Read only registers

R_AF0_OVIEW	(r)	(Reset group: H, 0, 3)	0x13
Alternate frame synchronization signal overview register			
<p>This register reports the status of FSC selection of all line interfaces. FSC can either be the F0IO signal or the AF0 signal. The register value can be stored by the application software to detect any frequency slips. An interrupt can be enabled which indicates a change of this register (see V_SLIP_IRQMSK in register R_MISC_IRQMSK).</p>			
Bits	Reset value	Name	Description
0	0	V_SU0_AF0	FSC selection for line interface 0 This bit is equal to A_SU_STA with line interface 0 being selected.
1	0	V_SU1_AF0	FSC selection for line interface 1 This bit is equal to A_SU_STA with line interface 1 being selected.
7..2		(reserved)	

Bits	Reset value	Name	Description
A_SU_RD_STA [ST/Up] (r) (Reset group: H, 0, 3) 0x30			
ST/Up state machine register			
This register is used to read the current state. A new state can be set with register A_SU_WR_STA.			
Before reading this array register the line interface must be selected by register R_SU_SEL.			
3..0	0	V_SU_STA	ST/Up state Binary value of current state (NT/LT: Gx, TE: Fx)
4	0	V_SU_FR_SYNC	Frame synchronization '0' = not synchronized '1' = synchronized
5	0	V_SU_T2_EXP	Timer T2 expired '1' = timer T2 expired (NT/LT mode only)
6	0	V_SU_INFO0	INFO 0 '1' = receiving INFO 0
7	0	V_G2_G3	G2 to G3 transition allowed '0' = no operation '1' = allows transition from G2 to G3 in NT/LT mode This bit is automatically cleared after the transition and has no function in TE mode.

A_SU_DLYL [ST/Up]	(r)	(Reset group: -)	0x31
<p>NT-TE-NT/LT-TE-LT delay</p> <p>This register shows the round trip delay and is only valid in NT/LT mode. It is updated once every 250 μs.</p> <p>Line interface in S/T mode: This register reports the delay between the F-/L-bit transition of the transmit frame to the F-/L-bit transition of the receive frame of a device in NT mode. The resolution is $t_0 = 1/6.144\text{MHz} = 1/32\text{ bit length} = 162.8\text{ns}$. The minimum delay is 2 bit times because of the NT/TE frame offset.</p> <p>Line interface in Up mode: This register reports the <i>end of transmit frame</i> to <i>begin of receive frame</i> delay of a device in LT mode. The resolution is $t_0 = 1/12.288\text{MHz} = 1/32\text{ bit length} = 81.4\text{ns}$. The minimum delay is 2 bit times because of the guard time.</p> <p>In both line interface modes, the round trip delay is measured within a 10 bit range. The lower 5 bits can be read from this register, the upper 5 bits are stored in register A_SU_DLYH.</p> <p>Before reading this array register the line interface must be selected by register R_SU_SEL.</p>			
Bits	Reset value	Name	Description
4..0		V_SU_DLYL	<p>Lower part of the round trip delay The lower part of the delay time $t_{dly,l} = t_0 \cdot \text{V_SU_DLYL}$ Register A_SU_DLYH must also be read to obtain the whole delay $t_{dly} = t_{dly,l} + t_{dly,h}$.</p> <p>S/T interface mode: 0 = delay $t_{dly,l} = 0\text{ns}$ 1 = delay $t_{dly,l} = 162.8\text{ns}$ 2 = delay $t_{dly,l} = 325.5\text{ns}$ 3 = delay $t_{dly,l} = 488.3\text{ns}$... 31 = delay $t_{dly,l} = 5045.6\text{ns}$</p> <p>Up interface mode: 0 = delay $t_{dly,l} = 0\text{ns}$ 1 = delay $t_{dly,l} = 81.4\text{ns}$ 2 = delay $t_{dly,l} = 162.8\text{ns}$ 3 = delay $t_{dly,l} = 244.1\text{ns}$... 31 = delay $t_{dly,l} = 2522.8\text{ns}$</p>
7..5		(reserved)	

A_SU_DLYH [ST/Up] (r) (Reset group: –) **0x32**

NT-TE-NT / LT-TE-LT delay

This register shows the round trip delay and is only valid in NT/LT mode. It is updated once every 250 μ s.

Line interface in S/T mode: This register reports the delay between the F-/L-bit transition of the transmit frame to the F-/L-bit transition of the receive frame of a device in NT mode. The resolution is $t_0 = 1/6.144\text{MHz} = 1/32\text{bit length} = 162.8\text{ns}$. The minimum delay is 2 bit times because of the NT/TE frame offset.

Line interface in Up mode: This register reports the *end of transmit frame to begin of receive frame* delay of a device in LT mode. The resolution is $t_0 = 1/12.288\text{MHz} = 1/32\text{bit length} = 81.4\text{ns}$. The minimum delay is 2 bit times because of the guard time.

In both line interface modes, the round trip delay is measured within a 10 bit range. The lower 5 bits are stored in register A_SU_DLYL, the upper 5 bits can be read from this register. if only this register is read, the resolution is $32 \cdot t_0 = 5.208\mu\text{s}$ for a line interface in S/T mode and $32 \cdot t_0 = 2.604\mu\text{s}$ for a line interface in Up mode, which is the length of one bit in both cases.

Before reading this array register the line interface must be selected by register R_SU_SEL.

Bits	Reset value	Name	Description
4..0		V_SU_DLYH	<p>Upper part of the round trip delay The upper part of the delay time $t_{dly,h} = 32 \cdot t_0 \cdot V_SU_DLYH$. Register A_SU_DLYL can be read to obtain the fractional part of a bit time delay.</p> <p>S/T interface mode: 0 = delay $t_{dly,h} = 0\mu\text{s}$ 1 = delay $t_{dly,h} = 5.208\mu\text{s}$ 2 = delay $t_{dly,h} = 10.416\mu\text{s}$ 3 = delay $t_{dly,h} = 15.625\mu\text{s}$... 31 = delay $t_{dly,h} = 161.458\mu\text{s}$</p> <p>Up interface mode: 0 = delay $t_{dly,h} = 0\mu\text{s}$ 1 = delay $t_{dly,h} = 2.604\mu\text{s}$ 2 = delay $t_{dly,h} = 5.208\mu\text{s}$ 3 = delay $t_{dly,h} = 7.812\mu\text{s}$... 31 = delay $t_{dly,h} = 80.729\mu\text{s}$</p>
7..5		(reserved)	

A_MS_RX [ST/Up]		(r)	(Reset group: H, 0, 3)	0x34
Multiframe/superframe receive register				
Before reading this array register the line interface must be selected by register R_SU_SEL.				
Bits	Reset value	Name	Description	
3..0		V_MS_RX	Multiframe/superframe bits The meaning of this bitmap depends on the selected line interface mode: S/T interface mode: S/Q bits received in the multiframe. Bits [3..0] are S bits [S1,S2,S3,S4] in TE mode and are Q bits [Q1,Q2,Q3,Q4] in NT mode. Up interface mode: T bits received in two consecutive superframes (M-channel). Bits [3..0] are T bits [T1a,T1b,T2a,T2b].	
4	0	V_MS_RX_RDY	Received multiframe/superframe ready S/T interface mode: This bit gets '1' when a complete S or Q multiframe has been received and is ready to get read. Up interface mode: This bit gets '1' when two complete T superframes have been received and are ready to get read. Reading this register clears this bit.	
5		(reserved)		
6		V_UP_S_RX	S-bit of the Up superframe Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_D_RX and A_E_RX. Note: This bit is only used for line interfaces in Up mode. It is undefined in S/T interface mode.	
7	0	V_MS_TX_RDY	Transmitted multiframe/superframe ready S/T interface mode: This bit gets '1' when new S- or Q-bits can be written into register A_MS_TX Up interface mode: This bit gets '1' when a set of new T-bits can be written into register A_MS_TX Writing to register A_MS_TX clears this bit.	

A_SU_STA [ST/Up] (r) (Reset group: H, 0, 3) **0x35**

Status register of the line interface

Before reading this array register the line interface must be selected by register R_SU_SEL.

Note: Only bit V_SU_AF0 is used for line interfaces in either S/T or Up mode. All other bits of this register are only used in S/T interface mode.

Bits	Reset value	Name	Description
0	0	V_ST_D_HPRI09	<p>Current D-channel priority state if high priority is selected</p> <p>When high D-channel priority is selected with V_ST_D_LPRIO = '0' in register A_SU_CTRL0, this bit reports the current access priority, i.e. the number of '1's before a D-channel access is allowed.</p> <p>'0' = 8 bits '1' '1' = 9 bits '1'</p> <p>Note: This bit is only valid in S/T interface mode.</p>
1	0	V_ST_D_LPRIO11	<p>Current D-channel priority state if low priority is selected</p> <p>When low D-channel priority is selected with V_ST_D_LPRIO = '1' in register A_SU_CTRL0, this bit reports the current access priority, i.e. the number of '1's before a D-channel access is allowed.</p> <p>'0' = 10 bits '1' '1' = 11 bits '1'</p> <p>Note: This bit is only valid in S/T interface mode.</p>
2	0	V_ST_D_CONT	<p>D-channel contention</p> <p>This bit reports a difference between D- and E-channel.</p> <p>'0' = no contention occurred '1' = contention occurred in the last frame</p> <p>Note: This bit is only valid in S/T interface mode.</p>
3	0	V_ST_D_ACT	<p>D-channel active</p> <p>This bit has the value '1' when the D-channel is just transmitting data.</p> <p>Note: This bit is only valid in S/T interface mode.</p>
6..4		(reserved)	

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	V_SU_AF0	<p>FSC selection</p> <p>The frame synchronisation clock can either be the F0IO input signal or the AF0 signal which is F0IO delayed for 62.5 µs. This bit reports the current selection and can be stored by the application software to detect changes. An interrupt can be enabled which indicates a change of this bit (see V_SLIP_IRQMSK in register R_MISC_IRQMSK). '0' = F0IO used '1' = AF0 used</p> <p>This bit is also available in the overview register R_AF0_OVIEW.</p>

A_B1_RX [ST/Up]	(r)	(Reset group: –)	0x3C
<p>Receive register for the B1-channel data</p> <p>This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.</p> <p>Before reading this array register the line interface must be selected by register R_SU_SEL.</p>			
Bits	Reset value	Name	Description
7..0		V_B1_RX	<p>B1-channel data byte</p> <p>In activated state data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS). Register value is 0xFF in deactivated state.</p>

A_B2_RX [ST/Up] (r) (Reset group: –) **0x3D**

Receive register of the B2-channel data

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

Before reading this array register the line interface must be selected by register R_SU_SEL.

Bits	Reset value	Name	Description
7..0		V_B2_RX	B2-channel data byte In activated state data is valid after a FOIO or AFO pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS). Register value is 0xFF in deactivated state.

Bits	Reset value	Name	Description
<p>A_D_RX [ST/Up] (r) (Reset group: -) 0x3E</p> <p>Receive register for the D-channel data</p> <p>This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.</p> <p>Before reading this array register the line interface must be selected by register R_SU_SEL. Data is valid after a FOIO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).</p>			
0		V_D_RX_S	<p>S-bit of the Up superframe Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_E_RX and A_MS_RX. Note: This bit is only used for line interfaces in Up mode. It is undefined in S/T interface mode.</p>
3..1		(reserved)	A read access to these unused bits returns the value of V_SU_RX_VAL in register A_MS_DF.
4		V_D_RX_AB	<p>A/B-bit When V_SQ_T_DST = '1', V_D_RX_AB contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_D_RX_AB is constant '1' when V_SQ_T_DST = '0'. The A/B-bit can also be read from register A_E_RX.</p>
5		V_D_RX_SG	<p>S/G-bit When V_SQ_T_DST = '0', V_D_RX_SG contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_D_RX_SG is constant '1' when V_SQ_T_DST = '1'. The S/G-bit can also be read from register A_E_RX.</p>
7..6		V_D_RX	<p>D-channel data bits Data is valid after a FOIO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).</p>

A_E_RX[ST/Up]

(r)

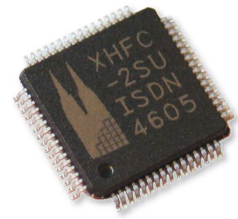
(Reset group: -)

0x3F**Receive register for the E-channel data**

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

Before reading this array register the line interface must be selected by register R_SU_SEL. Data is valid after a FOIO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).

Bits	Reset value	Name	Description
0		V_E_RX_S	S-bit of the Up superframe Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_D_RX and A_MS_RX. Note: This bit is only used for line interfaces in Up mode. It is undefined in S/T interface mode.
3..1		(reserved)	A read access to these unused bits returns the value of V_SU_RX_VAL in register A_MS_DF.
4		V_E_RX_AB	A/B-bit When V_SQ_T_DST = '1', V_E_RX_AB contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_E_RX_AB is constant '1' when V_SQ_T_DST = '0'. The A/B-bit can also be read from register A_D_RX.
5		V_E_RX_SG	S/G-bit When V_SQ_T_DST = '0', V_E_RX_SG contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_E_RX_SG is constant '1' when V_SQ_T_DST = '1'. The S/G-bit can also be read from register A_D_RX.
7..6		V_E_RX	E-channel data bits Data is valid after a FOIO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).



Chapter 6

PCM interface

Table 6.1: Overview of the XHFC-2SU PCM interface registers

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x10	R_SLOT	253	0x18	R_F0_CNTL	271
0x14	R_PCM_MD0	254	0x19	R_F0_CNTH	271
0x15	R_SL_SEL0	255	0x1D	R_SL_MAX	271
0x15	R_SL_SEL1	256	0x28	R_CI_RX	272
0x15	R_SL_SEL7	256	0x29	R_GCI_STA	273
0x15	R_MSS0	257	0x2A	R_MON_RX	273
0x15	R_PCM_MD1	259			
0x15	R_PCM_MD2	261	Read / write registers:		
0x15	R_MSS1	263	Address	Name	Page
0x15	R_SH0L	264	0xD0	A_SL_CFG	274
0x15	R_SH0H	264			
0x15	R_SH1L	264			
0x15	R_SH1H	265			
0x17	R_SU_SYNC	266			
0x28	R_CI_TX	267			
0x29	R_GCI_CFG0	268			
0x2A	R_GCI_CFG1	270			
0x2B	R_MON_TX	270			

6.1 PCM interface function

XHFC-2SU can operate in PCM master mode or PCM slave mode. This is selected with V_PCM_MD in register R_PCM_MD0.

The PCM data rate is programmable for PCM master mode as shown in Table 6.2. F0IO has always a frequency of 8 kHz. Each time slot has a width of eight bits.

Table 6.2: *PCM master mode*

V_PCM_DR in register R_PCM_MD1	C4IO clock output	Number of time slots	Data rate	Name
'00'	4.096 MHz	32	2 MBit/s	PCM30
'01'	8.192 MHz	64	4 MBit/s	PCM64
'10'	16.384 MHz	128	8 MBit/s	PCM128
'11'	1.536 MHz	12	0.75 MBit/s	

When PCM slave mode is selected, the number of PCM time slots is derived from the C4IO input frequency and V_PCM_DR is ignored. Any frequency

$$f(\text{C4IO}) = n \cdot 128 \text{ kHz} \text{ with } n = 1 \dots 128$$

is allowed and leads to n PCM time slots. F0IO must always have a frequency of 8 kHz.

XHFC-2SU has two PCM data pins STIO1 and STIO2 which can both be input or output. Data direction can be selected for every time slot independently.

6.2 PCM data flow

The PCM data flow is shown in Figure 6.1. The input and output behavior has to be programmed with bitmap `V_ROUT` in array register `A_SL_CFG[SLOT]`. Every time slots has its own configuration.

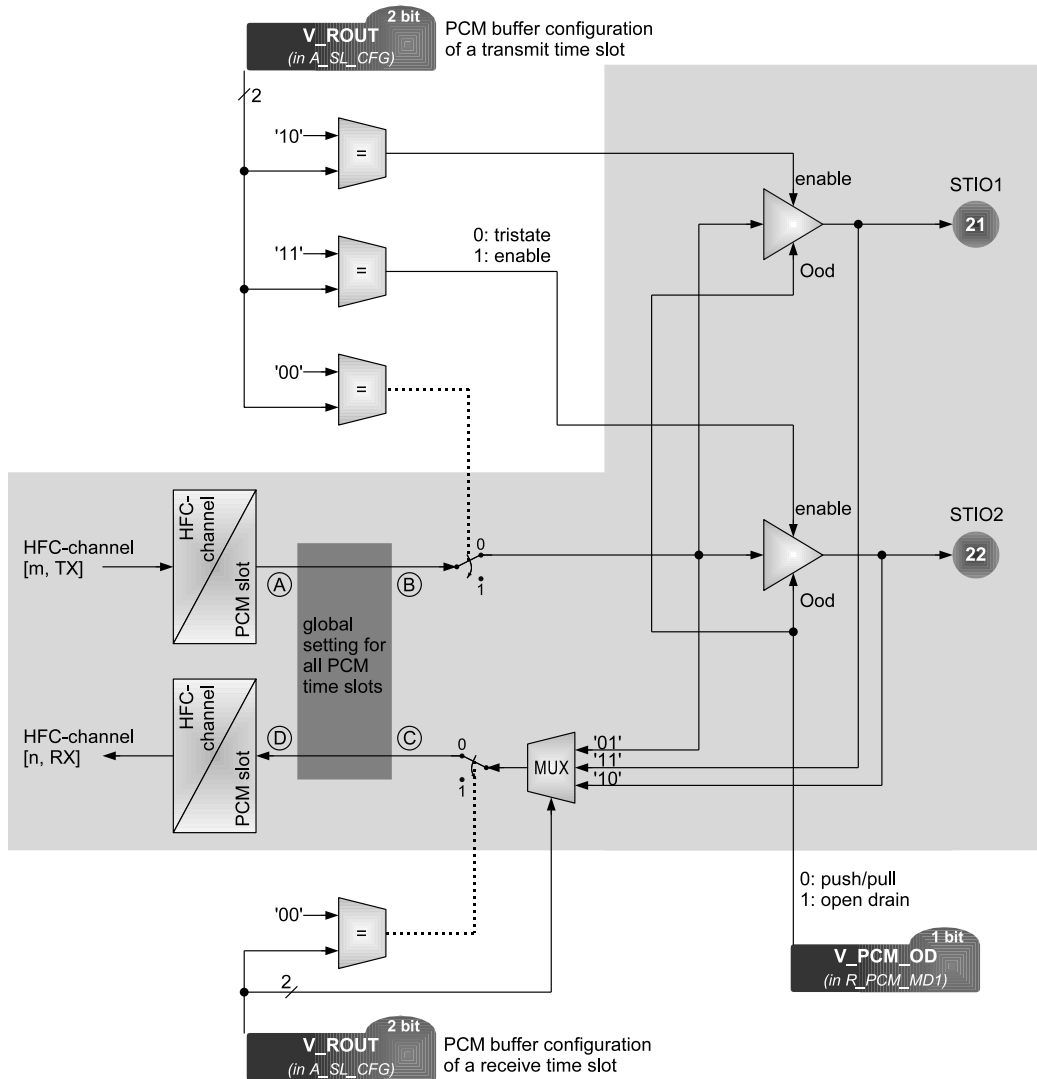


Figure 6.1: PCM data flow for transmit and receive time slots (see Figure 6.2 for additional setting of all PCM time slots between (A) .. (D))

The PCM output behavior is always setup from transmit slots. `V_ROUT = '00'` disables the PCM output, i.e. both output buffers are tristated and no data is transferred from the HFC-channel to the PCM module within this time slot. Any other value of `V_ROUT` enables the data transmission from the HFC-channel:

- `V_ROUT = '10'` enables the STIO1 output buffer.
- `V_ROUT = '11'` enables the STIO2 output buffer.
- Finally, `V_ROUT = '01'` disables both output buffers but enables the data transmission from the HFC-channel. This setting – in connection with the corresponding setting of the PCM input data path – is used to loop data internally without influencing the PCM bus.

6.3 PCM initialization

After hardware, global software or PCM reset the PCM interface starts an initialization sequence to set all A_SL_CFG registers of the PCM time slots to the reset value 0. This is even done if no valid C4IO and F0IO signals exist, which might occur in PCM slave mode. C4IO and F0IO input signals are ignored during PCM initialization.

The initialization process is indicated with V_PCM_INIT = '1' in register R_STATUS. This bit changes to '0' when the initialization sequence is finished.

6.4 PCM timing

6.4.1 Mode selection

The PCM interface of XHFC-2SU can operate either in slave mode or master mode. Slave mode is the default selection after XHFC-2SU reset.

To configure XHFC-2SU as PCM bus master, bit V_PCM_MD in register R_PCM_MD0 must be set to '1'. C4IO and F0IO signals are generated from XHFC-2SU in this case and both pins have output characteristic.

Slave mode is selected with V_PCM_MD = '0'. C4IO and F0IO are input pins in slave mode. External clocks must be connected to F0IO and also to either C4IO (when V_C2I_EN = '0' in register R_PCM_MD2) or C2IO (when V_C2I_EN = '1') in PCM slave mode because these signals are used from the ST/U_p interface and the flow controller as well (see Figure 6.5 on page 229, signal C4I is required).

6.4.2 Master mode

Figure 6.3 shows the timing diagram for PCM master mode. The timing characteristics are specified in Table 6.4. STIO1 is shown as data output and STIO2 as data input of XHFC-2SU. However, both pins can change their I/O characteristic with every PCM time slot.

The F0IO pulse is one C4IO pulse long with the default value V_F0_LEN = '0' in register R_PCM_MD0. F0IO starts one C4IO clock earlier if bit V_F0_LEN = '1'. These two pulse length are shown in Figure 6.3. The F0IO pulse can also be lengthened to indicate the start of a multiframe or superframe (see Section 6.6.4 on page 238).

6.4.3 Slave mode

Figure 6.4 shows the timing diagram for PCM slave mode. The timing characteristics are specified in Table 6.5. STIO1 is shown as data output and STIO2 as data input of XHFC-2SU. However, both pins can change their I/O characteristic with every PCM time slot.

The F0IO pulse is expected to be one C4IO pulse long with the default value V_F0_LEN = '0' in register R_PCM_MD0. F0IO is expected to start one C4IO clock earlier if bit V_F0_LEN = '1'.

If the ST/U_p interfaces are synchronized from C4IO in NT/LT mode, the frequency stability must be at least $\pm 10^{-4}$.

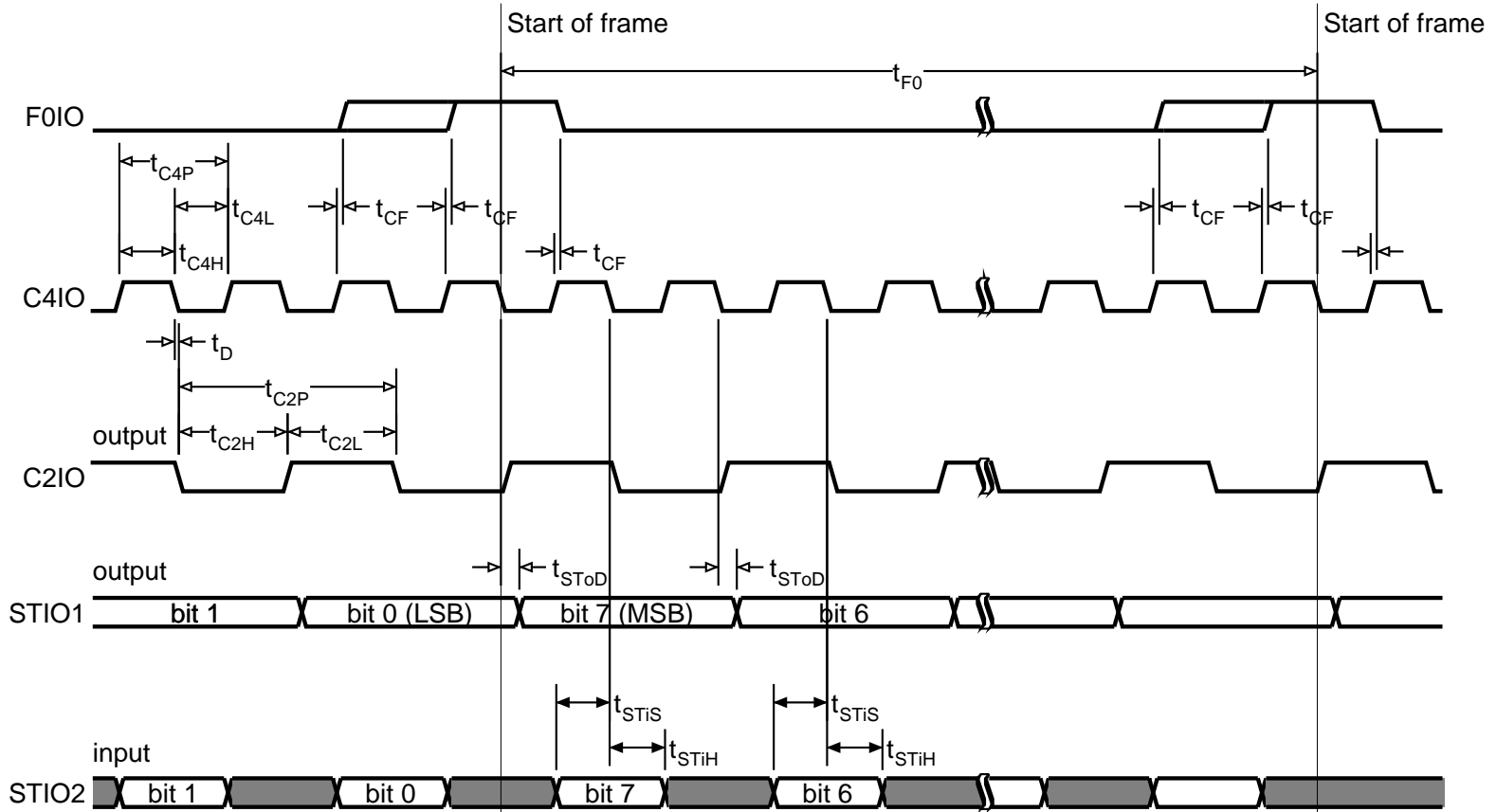


Figure 6.3: PCM timing for master mode

STIO1 is shown as data output and STIO2 as data input. However, both pins can change their I/O characteristic with every PCM time slot.

Table 6.4: Symbols of PCM timing for master mode in Figure 6.3 (All values with 50 pF load. Larger load capacitance will increase output delays.)

Symbol	min / ns	typ / ns	max / ns	Characteristic
t_C		122.070		Basic C4IO pulse width (not shown in the timing diagram)
		61.035		4.096 MHz C4IO clock for 2 MB/s
		30.518		8.192 MHz C4IO clock for 4 MB/s 16.384 MHz C4IO clock for 8 MB/s
t_{adj}		20.345		Adjust time is half a period of 24.576 MHz clock (not shown in the timing diagram)
t_{C4H}	$t_C - 6 - t_{adj}$		$t_C + 6$	C4IO high width for 2 MBit/s and 4 MBit/s
	$4/3 \cdot t_C - 6 - t_{adj}$		$4/3 \cdot t_C + 6$	C4IO high width for 8 MBit/s
t_{C4L}	$t_C - 6$		$t_C + 6 + t_{adj}$	C4IO low width for 2 MBit/s and 4 MBit/s
	$2/3 \cdot t_C - 6$		$2/3 \cdot t_C + 6 + t_{adj}$	C4IO low width for 8 MBit/s
t_{C4P}	$2 \cdot t_C - 6 - t_{adj}$		$2 \cdot t_C + 6 + t_{adj}$	C4IO clock period
t_{C2H}	$2 \cdot t_C - 6 - t_{adj}$		$2 \cdot t_C + 6 + t_{adj}$	C2IO output high width
t_{C2L}	$2 \cdot t_C - 6 - t_{adj}$		$2 \cdot t_C + 6 + t_{adj}$	C2IO output low width
t_{C2P}	$4 \cdot t_C - 6 - t_{adj}$		$4 \cdot t_C + 6 + t_{adj}$	C2IO output clock period
t_{F0}	124994	125000	125006	F0IO cycle time without adjustment
	$124994 - t_{adj}$		$125006 + t_{adj}$	1 half clock adjustment
	$124994 - 2 \cdot t_{adj}$		$125006 + 2 \cdot t_{adj}$	2 half clocks adjustment
	$124994 - 3 \cdot t_{adj}$		$125006 + 3 \cdot t_{adj}$	3 half clocks adjustment
	$124994 - 4 \cdot t_{adj}$		$125006 + 4 \cdot t_{adj}$	4 half clocks adjustment
t_D	3	5	8	C4IO \downarrow to C2IO \uparrow delay
t_{CF}	0.5		8	C4IO \uparrow to F0IO \uparrow or C4IO \uparrow to F0IO \downarrow
t_{STIS}	10			Data valid to C4IO \downarrow setup time
t_{STIH}	10			Data valid to C4IO \downarrow hold time
t_{SToD}	2		10	STIO output delay from C4IO \downarrow

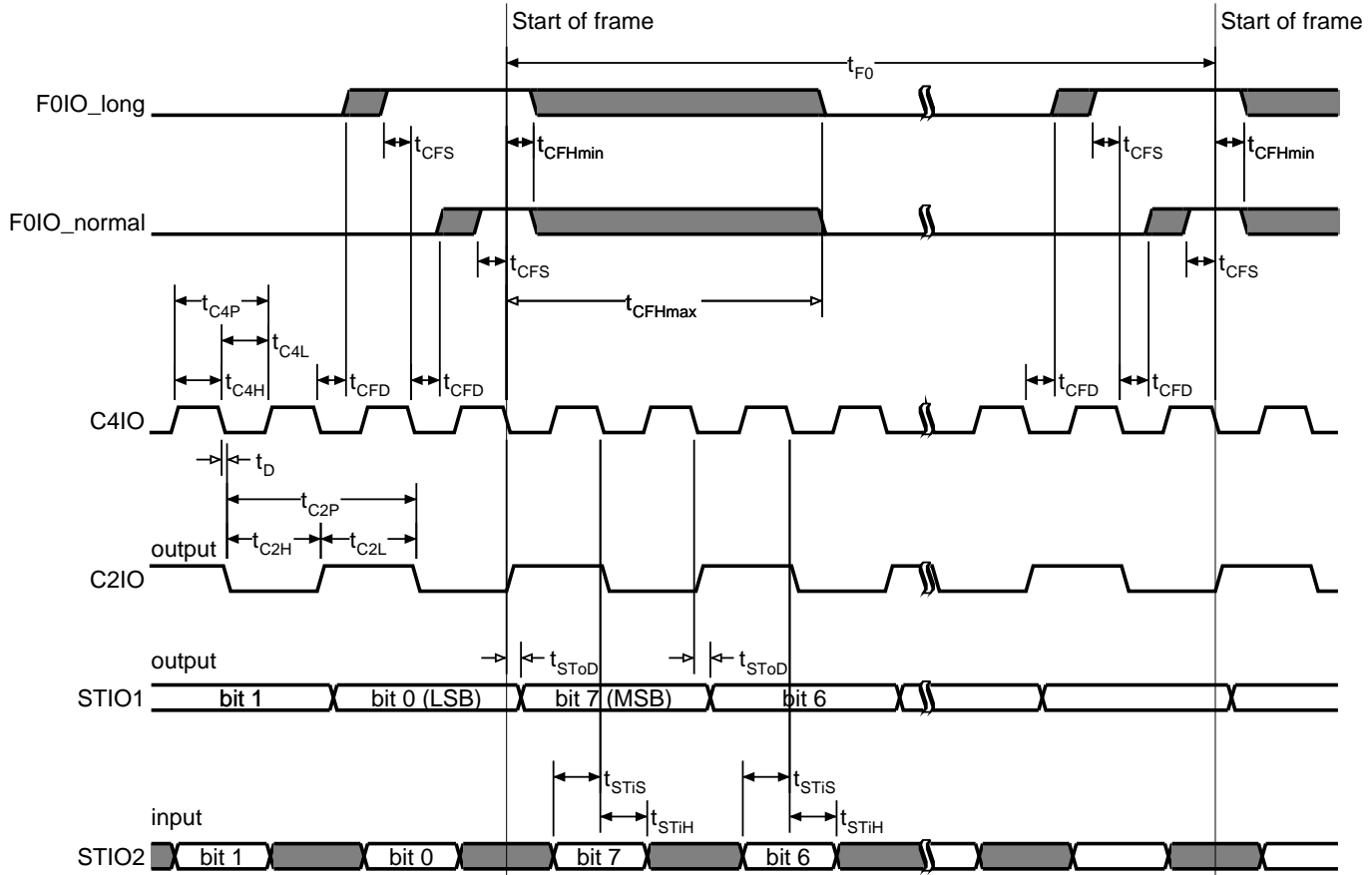


Figure 6.4: PCM timing for slave mode

STIO1 is shown as data output and STIO2 as data input. However, both pins can change their I/O characteristic with every PCM time slot.

Table 6.5: Symbols of PCM timing for slave mode in Figure 6.4 (All values with 50 pF load. Larger load capacitance will increase output delays.)

Symbol	min / ns	typ / ns	max / ns	Characteristic
t_C		122.070		Basic C4IO pulse width (not shown in the timing diagram)
		61.035		4.096 MHz C4IO clock for 2 MB/s
		30.518		8.192 MHz C4IO clock for 4 MB/s
				16.384 MHz C4IO clock for 8 MB/s
t_{C4H}	20	t_C		C4IO high width
t_{C4L}	20	t_C		C4IO low width
t_{C4P}		$2 \cdot t_C$		C4IO clock period
t_{C2H}		$2 \cdot t_C$		C2IO output high width
t_{C2L}		$2 \cdot t_C$		C2IO output low width
t_{C2P}		$4 \cdot t_C$		C2IO output clock period
t_{F0}		125000		F0IO cycle time
t_D	3	5	8	C4IO \downarrow to C2IO \uparrow delay
t_{CFS}	15	t_C		F0IO \uparrow to C4IO \downarrow setup time
t_{CFHmin}	15	t_C		F0IO \downarrow to C4IO \downarrow hold time
t_{CFHmax}	15	t_C	100000	F0IO high time after start of frame
t_{CFD}	15	t_C		C4IO \downarrow to F0IO \uparrow delay
t_{STiS}	10			Data valid to C4IO \downarrow setup time
t_{STiH}	10			Data valid to C4IO \downarrow hold time
t_{SToD}	2		10	STIO output delay from C4IO \downarrow

6.5 PCM clock synchronization

6.5.1 Overview

The PCM clock synchronization is shown in Figure 6.5. It is associated with the line interface clock synchronization which is shown in Figures 5.5 (page 167) and 5.13 (page 183) and with the MSS controller (shown in Figure 6.10 on page 236).

6.5.2 Manual or automatic synchronization source selection

Any line interface in TE mode can be chosen as synchronization source with an appropriate value in bitmap `V_SYNC_SEL` of register `R_SU_SYNC`. Alternatively, an automatic selection can be enabled with `V_MAN_SYNC = '0'` in the same register (reset default). The actual synchronization source can be read from `V_RD_SYNC_SRC` in register `R_BERT_STA`. If synchronization is lost on this TE, the next interface in TE mode with active synchronization is automatically selected.

When the automatic selection doesn't find a synchronization source, the `SYNC_I` pulse can alternatively be taken as synchronization source. This must be enabled with `V_AUTO_SYNCI = '1'` in register `R_SU_SYNC`.

6.5.3 PLL programming for F0IO generation

C4IO is adjusted from the PCM DPLL (see Figure 6.5) during the last PCM time slot to synchronize the PCM interface with the ST/U_p interface.¹ The maximum number of edge adjustments during one 125 μs cycle can be configured in the range 1..4 by the bitmap value `V_PLL_ADJ` in register `R_PCM_MD1`. This automatic adjustment is enabled with `V_PLL_MAN = '0'` in register `R_PCM_MD2`.

`V_PLL_MAN = '1'` switches into manual adjustment mode. In this case, the adjustment direction is specified in `V_PLL_ICR` of register `R_PCM_MD2`. The number of edge adjustments which is specified in `V_PLL_ADJ` is carried out within the last PCM time slot every 125 μs. This manual adjustment does not stop before `V_PLL_MAN` is reset to automatic mode. By default, the C4IO clock is adjusted four times for one half clock cycle. This can be reduced to one adjustment of a half clock cycle (see `R_PCM_MD1` register). This is useful if a non XHFC series ISDN controller is connected as slave in NT mode to the PCM bus. The synchronization source can be selected by the `R_PCM_MD2` register settings.

6.5.4 Manual PLL adjustment

In normal operation mode, the synchronization input signal is passed from the `V_SYNC_SRC` controlled multiplexer to the PCM DPLL as shown in Figure 6.5. For this `V_PLL_MAN` has to be '0' in register `R_PCM_MD2`.

The PLL output frequency can manually be adjusted if no synchronization source is available. This software controlled PLL adjustment is enabled with `V_PLL_MAN = '1'`. The `V_SYNC_SRC` controlled multiplexer must feed a 8 kHz signal in any case.

¹C4IO adjustment is only in operation when the PCM DPLL receives both 8 kHz reference clocks.

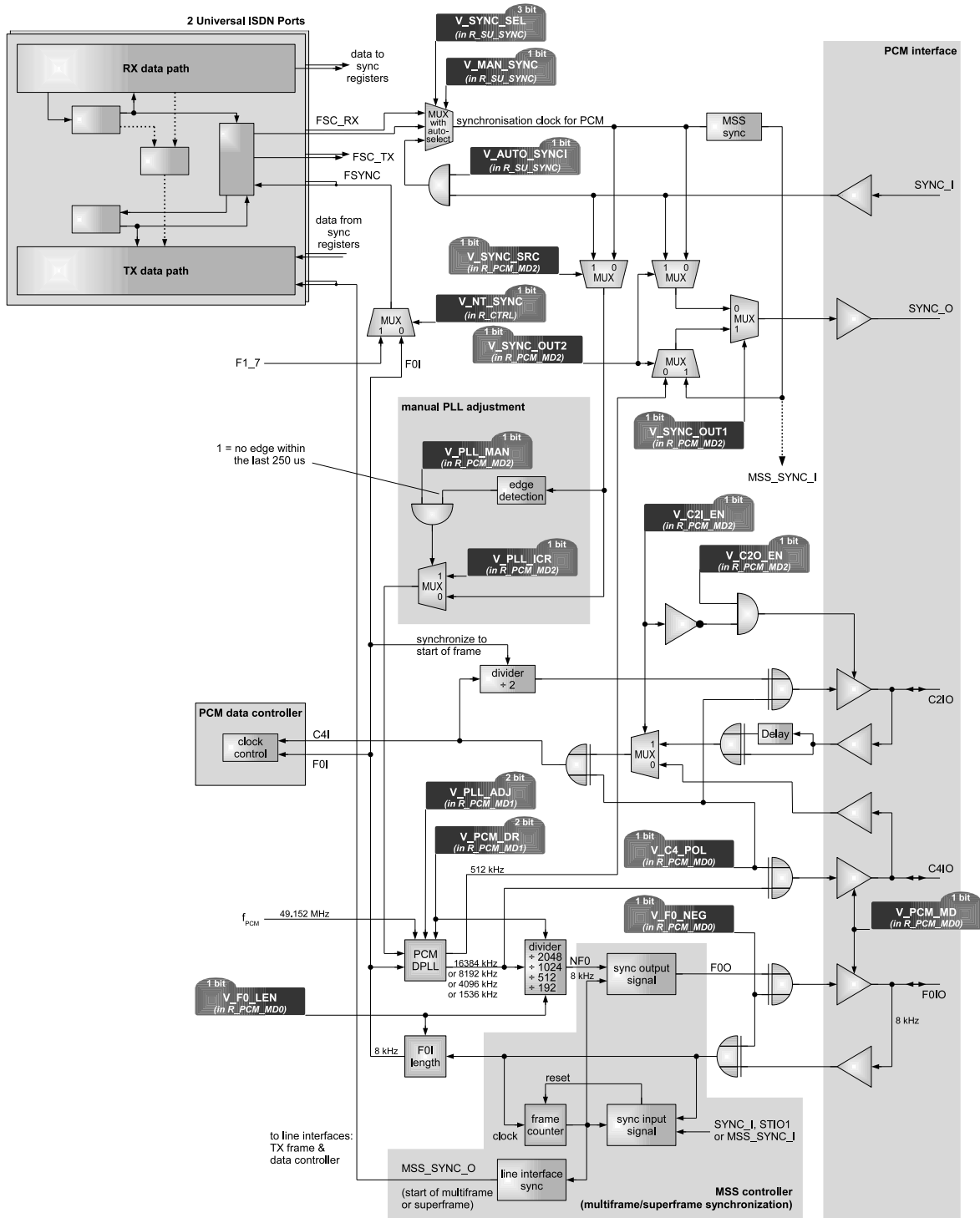


Figure 6.5: PCM clock synchronization (For details on the Universal ISDN Port see Figures 5.5 and 5.13, for details on the MSS controller see Figure 6.10)

The time of the signal edges can be increased or reduced in the last time slot of the PCM frame. $V_PLL_ICR = '0'$ results in a frequency reduction while $V_PLL_ICR = '1'$ leads to a frequency increase. The number of adjusted edges is specified in the range 1..4 with bitmap V_PLL_ADJ in register R_PCM_MD1 .

V_PLL_MAN must be set back to '0' to stop the frequency regulation of the synchronization input signal.

6.5.5 C2IO signal

The C2IO signal (pin 18) can either be configured to input or output characteristic.

C2IO is used as an output signal when $V_C2O_EN = '1'$ and $V_C2I_EN = '0'$ in register R_PCM_MD2 .

C2IO output signal is derived from C4IO by a frequency divider. In fact, there is an additional building block (not shown in Figure 6.5) which ensures a specific phase relation at the start of a frame. According to the timing diagrams in Figures 6.3 and 6.4, C2IO has its rising edge at the start of a frame. From this follows that C4IO has a falling edge with every edge of C2IO.

6.5.6 SYNC_O and FSC_RX synchronization signals

The 'frame synchronization detection' blocks shown in Figures 5.5 (page 167) and 5.13 (page 183) deliver a 8 kHz frame clock FSC_RX which can be routed to the SYNC_O synchronization output pin. FSC_RX signal is available when a frame synchronization is detected (INFO 2 or INFO 4) and it does not depend on the state machine's condition.

The normal FSC_RX pulse is high for one bit length. This is $5.208 \mu\text{s}$ in S/T interface mode or $2.604 \mu\text{s}$ in U_p interface mode. For S/T multiframe² synchronization or U_p superframe³ synchronization, the start of every multiframe or superframe is indicated with a lengthened pulse with 7.5 bit width. This is $39 \mu\text{s}$ every 40th pulse in S/T interface mode or $19.5 \mu\text{s}$ every 8th pulse in U_p interface mode.

The rising edges of FSC_RX have a distance of $125 \mu\text{s}$ and are adjusted once every $250 \mu\text{s}$ with $\pm 163 \text{ ns}$. The rising edge is stable as long as the chosen TE port receives valid INFO 2 or INFO 4 even if the S/T or U_p state machine is forced into a deactivated state F0, F2, F3, F4, F5, G0, G12, or G2. FSC_RX is low otherwise.

During XHFC-2SU reset, SYNC_O pin drives an active low signal.

6.5.7 Synchronous 512 kHz output signal

SYNC_O frequency can either be 8kHz, which is mostly used, or 512kHz. The 512kHz clock is synchronous to the frame clock as it is generated from the DPLL as shown in Figure 6.5.

When XHFC-2SU operates in PCM slave mode, which means that F0IO is used as clock input, SYNC_O should not issue 512 kHz. This would lead to a closed loop with the internal DPLL and an external PCM master PLL. A closed loop with two PLLs might be instable and can lose the nominal frequency.

²A complete S/T multiframe takes 20 S/T frames with $250 \mu\text{s}$ each. Therefore, a multiframe has a length of 5 ms in total.

³A complete U_p superframe takes 4 U_p frames with $250 \mu\text{s}$ each. Therefore, a superframe has a length of 1 ms in total.

6.5.8 Application examples for XHFC-2SU synchronization schemes

Flexible synchronization schemes can be implemented with the clock pins SYNC_I and SYNC_O of XHFC-2SU. Some important application examples are shown in this section.



Important !

As ISDN is based on a synchronous network, all devices must be synchronized to one synchronizing source. This is the central office (for S/T) or PBX (for U_p), typically.

When multiple XHFC-2SU are used within a system, all ST/U_p and PCM clocks must be synchronized to a single synchronization source. When an ST interface operates in TE mode and is connected to the central office, the synchronization source is obtained from the central office. A U_p interface in TE mode is typically synchronized to a PBX clock.

6.5.8.1 An existing system with internal PCM bus has to be expanded by ISDN interfaces

When an existing system has to be expanded by ISDN interfaces, there are two solutions as shown in Figures 6.6 and 6.7.

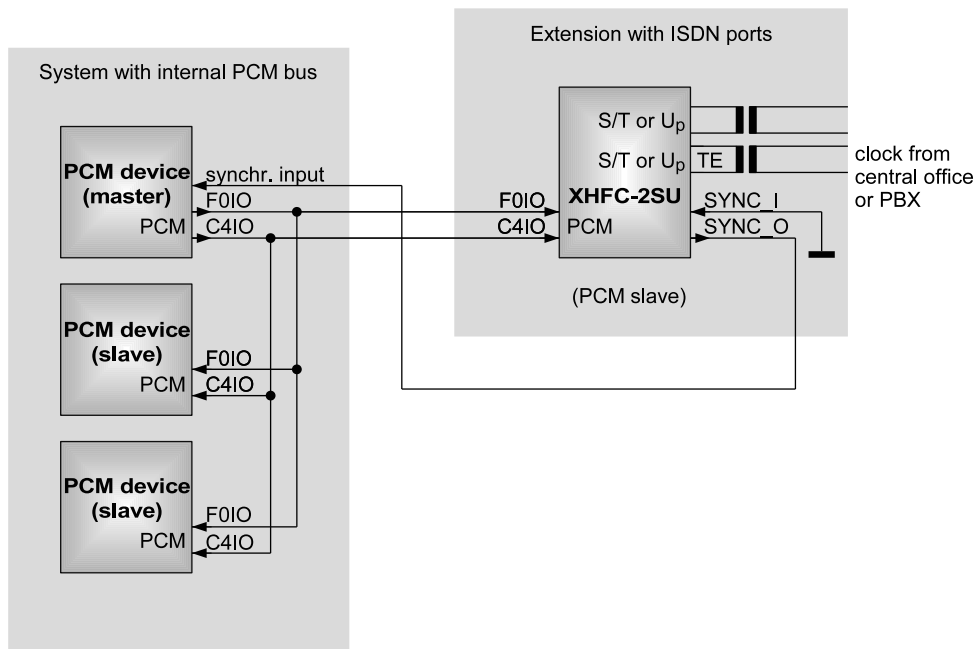


Figure 6.6: Expanding an existing system with ISDN ports (XHFC-2SU as PCM slave)

The existing system can keep the PCM master when there is a synchronization input (Figure 6.6). This must be connected to pin SYNC_O of XHFC-2SU to synchronize the whole system to the clock derived from the received ISDN signal.

SYNC_O is either derived from the central office or the DPLL of XHFC-2SU is free-running. In both cases, XHFC-2SU is the synchronization source for the whole system.

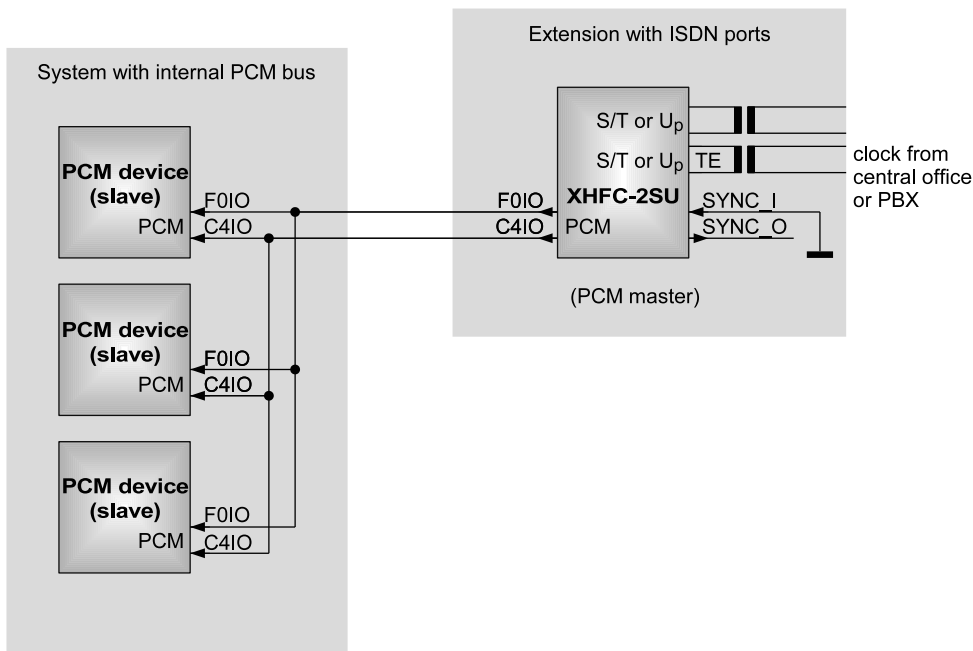


Figure 6.7: Expanding an existing system with ISDN ports (XHFC-2SU as PCM master)

When there is no synchronization input available at the existing system, XHFC-2SU must operate as PCM master and all other PCM devices in the system must be PCM slaves (Figure 6.7). Again, XHFC-2SU is the synchronization source for the whole system.

6.5.8.2 Application with multiple XHFC-2SU

Multiple XHFC-2SU can be interconnected to build up a multi ISDN port application. The SYNC_O / SYNC_I pins must be connected to a daisy chain. Two solutions are shown in Figures 6.8 and 6.9, depending on the PCM master requirements.

Figure 6.8 shows an example where any XHFC-2SU can be in TE mode. The synchronization signals are daisy chained. The last XHFC-2SU must be in PCM master mode. This assures that the F0IO/C4IO connections feed the synchronized clock to all devices.

Figure 6.9 shows an example where any XHFC-2SU can be in TE mode again. Now, the SYNC_O / SYNC_I daisy chain is looped back so that the loop is closed. For this reason, the F0IO/C4IO connections are optional due to the application needs. An arbitrary XHFC-2SU can be in PCM master mode.

Nevertheless, it is recommended to interconnect all devices with the F0IO/C4IO clocks for more flexible application capability.

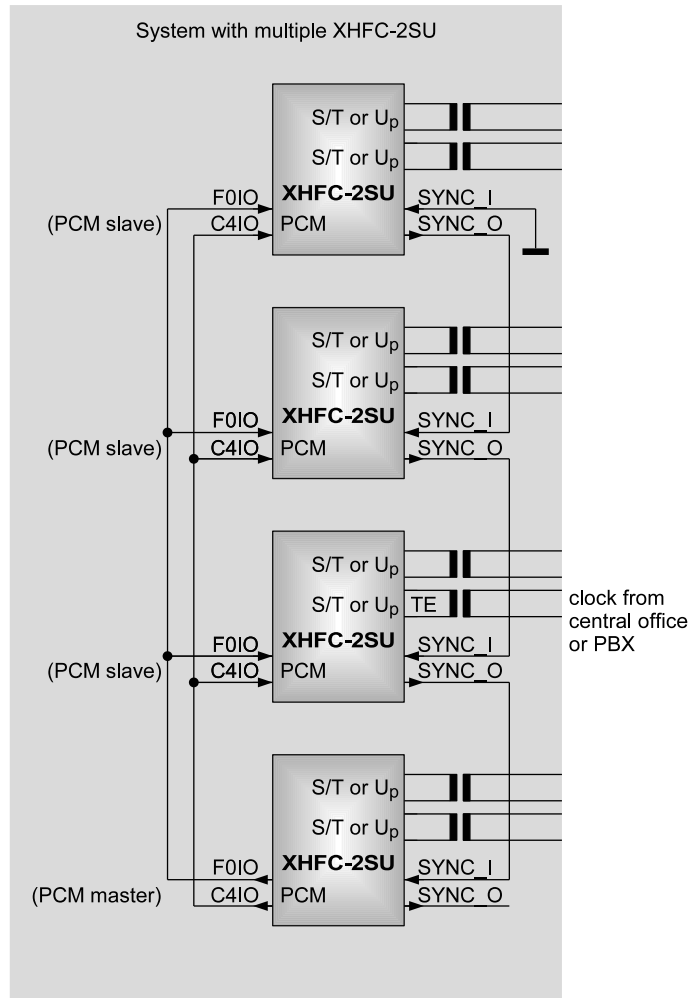


Figure 6.8: Multiple XHFC-2SU synchronized with an open loop of SYNC_O/SYNC_I

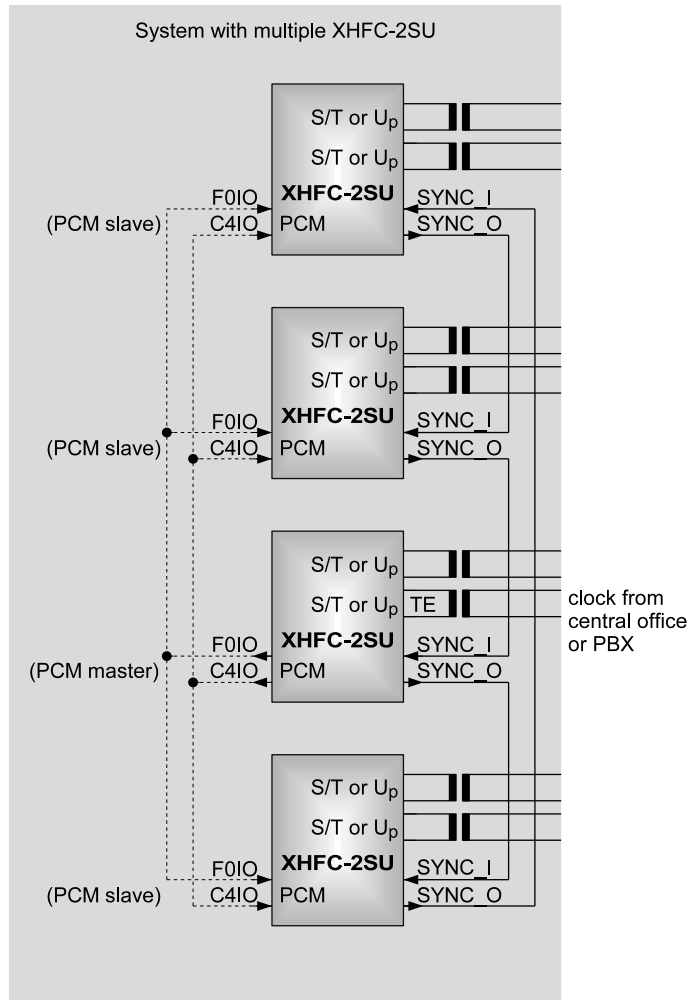


Figure 6.9: Multiple XHFC-2SU synchronized with an closed loop of SYNC_O/SYNC_I

6.6 Multiframe / superframe synchronization to the PCM interface

6.6.1 Overview

The multiframe (S/T) or superframes (U_p) of the ST/ U_p interfaces can be synchronized to the PCM interface. Furthermore, the transmit / receive ping pong is synchronized in U_p mode. Figure 6.10 shows the block diagram of the MSS controller (multiframe / superframe synchronization controller). The controller consists of four parts which are explained from Sections 6.6.2 to 6.6.5.

The basic concept of the MSS controller can be described as follows:

- An internal synchronization pulse is generated and delivered to the PCM interface for F0IO pulse width modification as well as to the line interface to force the 'start of multiframe / superframe' in transmit direction.
- The internal synchronization pulse is generated from a frame counter which can be synchronized to an external synchronization pulse.

6.6.2 Frame counter

The frame counter is the central unit of the MSS controller. The counter is incremented with every F0IO input pulse and operates as a ring counter in the range 0 . . 39.

A counter reset is initiated with a detected synchronization input signal. The frame counter is synchronized to the synchronization source with the first counter reset signal. From now on, the frame counter value is used

- to generate a modified F0IO pulse (if enabled) and
- to synchronize the multiframe / superframe transmission with the MSS_SYNC_O signal (if enabled and if the line interface operates in NT/LT mode, see Figure 5.5 on page 167 and Figure 5.13 on page 183)

to the external synchronization source.

The synchronization pulse detection can be configured with several parameters. This is explained in Section 6.6.3.

It is also possible to disable the synchronization pulse detection. There are no counter reset signals in this case. The frame counter is free-running without a synchronization input signal. However, synchronization pulses MSS_SYNC and the modified F0IO output are generated to synchronize the multiframe / superframe transmission with the PCM interface. The chip operates as a synchronization source for the 'start of multiframe / superframe' condition in this case.

6.6.3 Synchronization input signal

The MSS controller can synchronize the multiframe / superframe transmission to an external synchronization signal. This functionality must be enabled with $V_MSS_SRC_EN = '1'$ in register R_MSS0. Any detected synchronization pulse leads to a frame counter reset.

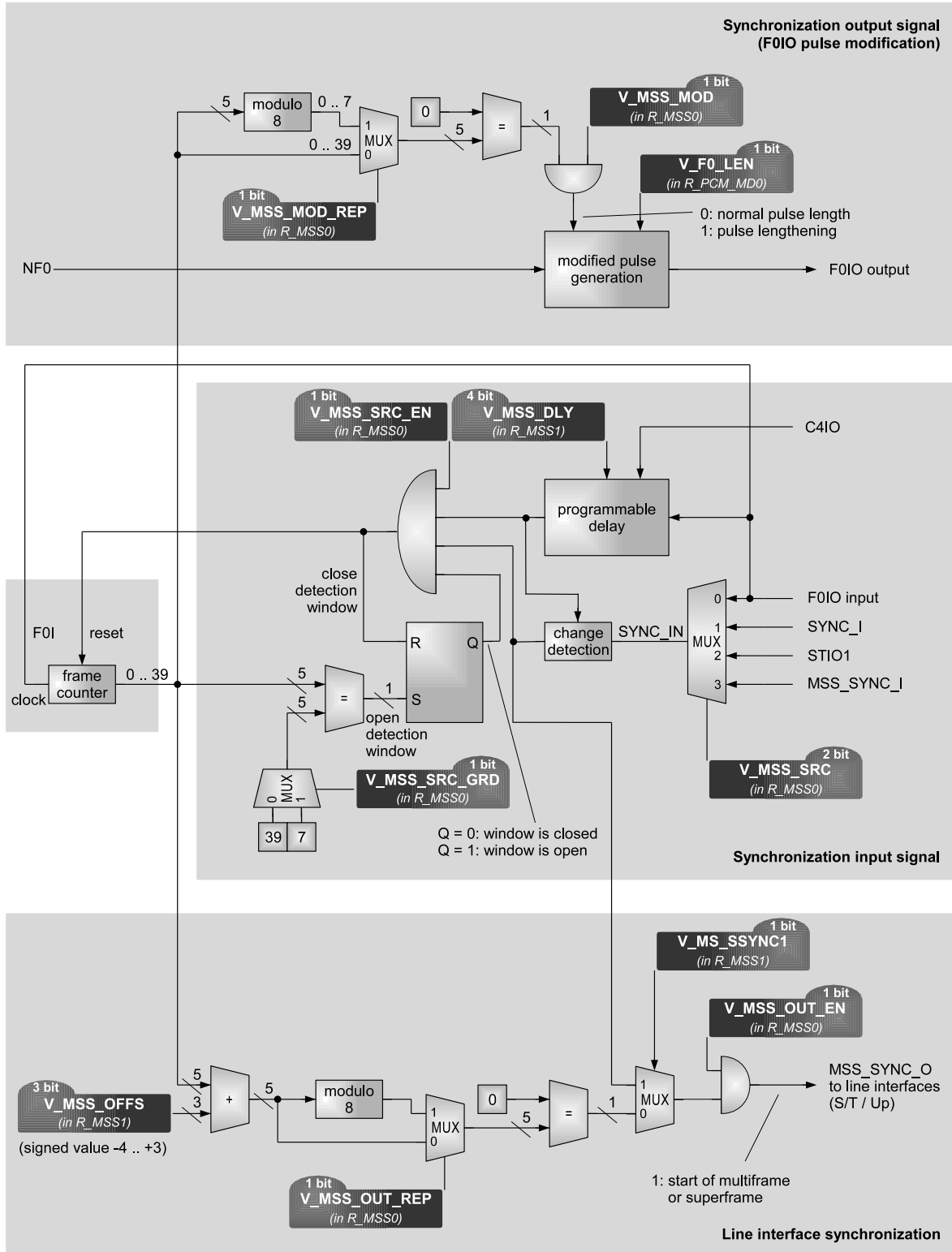


Figure 6.10: Multiframe/superframe synchronization to the PCM interface

The synchronization signal is obtained either from the F0IO, SYNC_I, STIO1 or MSS_SYNC_I signal. This can be selected with bitmap V_MSS_SRC in register R_MSS0. The source signal can be delayed by a multiple of C4IO clocks to adjust the incoming signal characteristics to the detection function. For this, V_MSS_DLY in register R_MSS1 can be set to a value in the range 0..15 C4IO clocks.

It is possible to use an active '1' or active '0' signal for synchronization. Only the change of the signal compared with the value in the previous PCM frame generates the synchronization signal.

After a synchronization pulse has reset the frame counter, the detection window is closed and the next synchronization pulse cannot be generated until the frame counter reaches the value 39 (V_MSS_SRC_GRD = '0' in register R_MSS0) or 7 (V_MSS_SRC_GRD = '1'). The detection window is opened as soon as the selected counter value has been reached. This procedure avoids an oversynchronization. V_MSS_SRC_GRD should be set to '0' when at least one line interface operates in S/T mode. The value V_MSS_SRC_GRD = '1' is recommended when both line interfaces operate in U_p mode.

The frame counter is synchronized to the synchronization input signal when the following conditions are fulfilled at the same time:

1. The synchronization detection is enabled, i.e. V_MSS_SRC_EN = '1'.
2. The detection windows is open, i.e. Q = '1'.
3. The programmable delay is passed.
4. The selected input signal has changed from the previous frame.

The delay time t_{DLY} is specified in Figure 6.11 and Table 6.6. Its maximum value is $t_{DLY,max} = 16 \cdot t_{C4P}$.

If the synchronization pulse is received on the data input line STIO1, it can be any bit within the first PCM time slot. V_MSS_DLY must have an even value in this case to ensure that the *change detection* time matches with the bit cells. V_MSS_DLY = 0 selects the first bit of the time slot 0 while V_MSS_DLY = 14 selects the last bit.

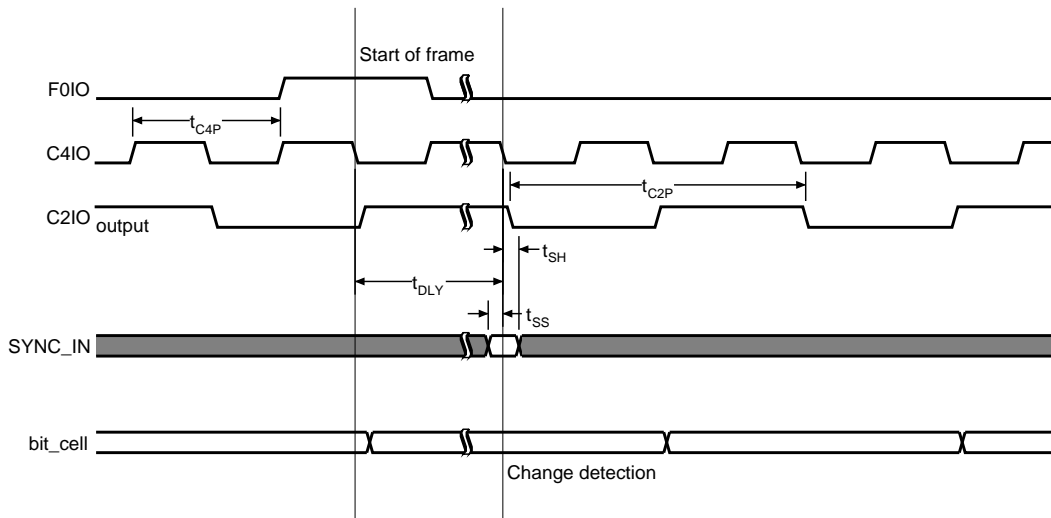


Figure 6.11: Timing specification of the SYNC_IN change detection

Table 6.6: Symbols of the SYNC_IN change detection timing specification in Figure 6.11

Symbol	min / ns	max / ns	Characteristic
t_{C4P}			C4IO clock period (depends on the selected PCM data rate)
t_{C2P}			C2IO clock period
t_{DLY}			Programmable synchronization pulse delay from start of frame $t_{DLY} = (V_MSS_DLY + 1) \cdot t_{C4P}$
t_{SS}	10		Synchronization pulse valid to F_SYNC \lceil setup time
t_{SH}	10		Synchronization pulse valid to F_SYNC \lfloor hold time

6.6.4 Synchronization output signal (F0IO pulse modification)

The start of a multiframe or superframe is indicated by a F0IO pulse which has a different length than usual.

In principle, the length of the F0IO pulse is arbitrary. The F0IO signal is used for PCM frame synchronization only on the first falling edge of C4IO. The F0IO high time has a typical length of one C4IO pulse ($V_F0_LEN = '0'$) or two C4IO pulses ($V_F0_LEN = '1'$) as described in Section 6.4.2.

The F0IO high time can be lengthen by another C4IO pulse width to indicate the multiframe/superframe synchronization signal. This must be enabled with $V_MSS_MOD = '1'$ in register R_MSS0. The repetition rate of the synchronization pulse is once every 40th PCM frame if $V_MSS_MOD_REP = '0'$ in register R_MSS0. $V_MSS_MOD_REP = '1'$ selects a repetition rate of once every 8th PCM frame.

Figure 6.12 shows all different F0IO pulses in PCM master mode. Table 6.7 summarizes the bit values which must be set to achieve these pulses.

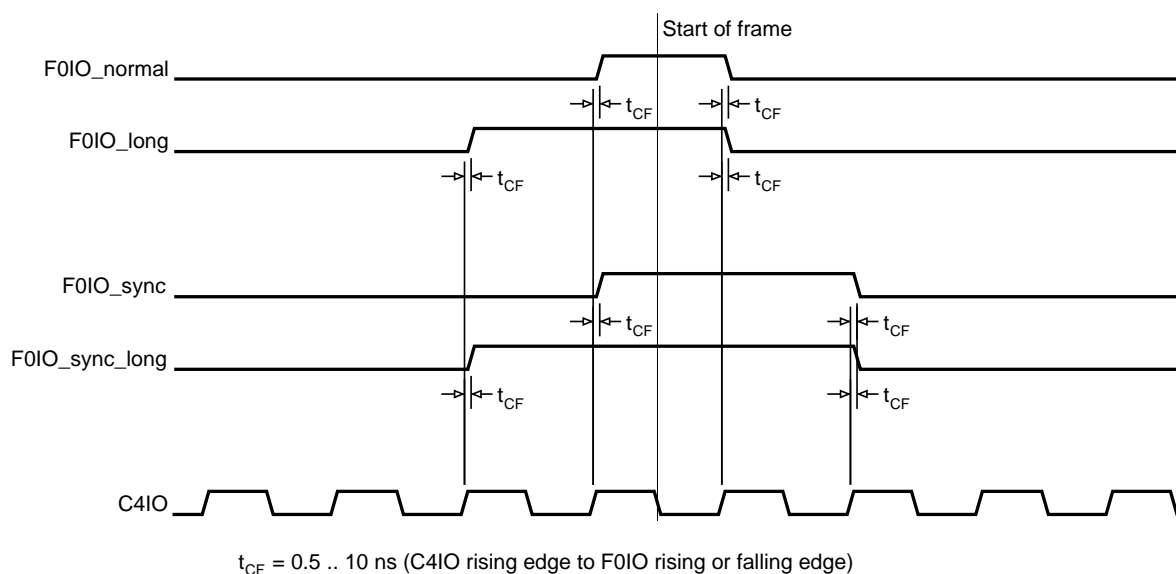

Figure 6.12: F0IO pulse modification

Table 6.7: Selection of the F0IO pulse characteristic (see also Figure 6.12)

Pulse	V_F0_LEN	V_MSS_MOD	Description
	in register R_PCM_MD0	in register R_MSS0	
F0IO_normal	'0'	'0'	normal F0IO pulse
F0IO_long	'1'	'0'	F0IO starts one C4IO period earlier (early leading edge)
F0IO_sync	'0'	'1'	F0IO pulse is lengthened by one C4IO pulse every 8th or 40th frame, F0IO_normal otherwise (delayed trailing edge)
F0IO_sync_long	'1'	'1'	F0IO starts one C4IO period earlier and is lengthened by one C4IO pulse every 8th or 40th frame, F0IO_long otherwise (combined early leading edge and delayed trailing edge)

6.6.5 Line interface synchronization

The signal MSS_SYNC_O for the multiframe/superframe synchronization is delivered to the ST/U_p interfaces. It must be enabled with V_MSS_OUT_EN = '1' in register R_MSS0. V_MSS_OUT_EN = '0' leads to an unsynchronized transmission of the multiframe/superframe structure.

The repetition rate can be specified to be either every 40th or 8th PCM frame with bit V_MSS_OUT_REP in register R_MSS0. Only line interfaces in U_p mode can be feed with a 8 PCM frame repetition rate. The repetition rate of once every 40 PCM frames must be chosen in S/T mode. This is a suitable selection for a line interface in U_p mode as well.

As there is a delay between F0IO and the line interface output, it is possible to adjust the line interface synchronization with a signed offset of -4.. +3 F0IO pulses. This can also be used to fulfill requirements which might occur with a special application where the chip is connected to another ISDN device.

6.7 External CODECs

Up to two external CODECs can be connected to the PCM interface. XHFC-2SU has two CODEC enable signals F1_0 and F1_1. An external CODEC has to be assigned to a PCM time slot via bitmaps V_SL_SEL1 and V_SL_SEL0 in registers R_SL_SEL1 and R_SL_SEL0.

The shape signals can be programmed. The last bit determines the inactive level by which non-inverted and inverted shape signals can be programmed. Every external CODEC can choose one of the two shape signals with bits V_SH_SEL1 and V_SH_SEL0 in registers R_SL_SEL1 and R_SL_SEL0.

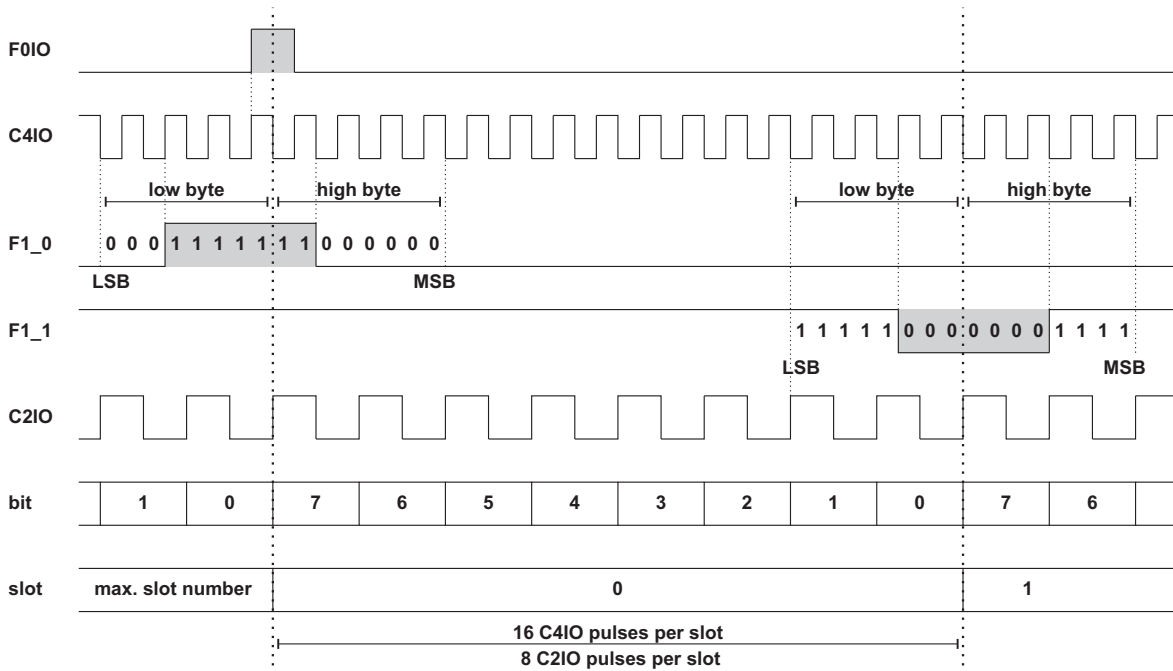


Figure 6.13: Example for two CODEC enable signal shapes

Figure 6.13 shows an example with two external CODECs. Time slot 0 starts with the F0IO pulse. In this example – assuming that PCM30 is configured – F1_0 enables the first CODEC on time slot 0 and shape bytes on R_SH0L and R_SH0H with the following register settings.

Register setup:	
R_PCM_MD0 : V_PCM_IDX = 0	(R_SL_SEL0 register accessible)
R_SL_SEL0 : V_SL_SEL0 = 0x1F	(time slot #0)
: V_SH_SEL0 = 0	(shape bytes R_SH0L and R_SH0H)

The second CODEC on time slot 1 and shape bytes on R_SH1L and R_SH1H must be configured as shown below.

Register setup:		
R_PCM_MD0	: V_PCM_IDX = 1	(R_SL_SEL1 register accessible)
R_SL_SEL1	: V_SL_SEL1 = 0	(time slot #1)
	: V_SH_SEL1 = 1	(shape bytes R_SH1L and R_SH1H)

The shown shape signals have to be programmed in reverse bit order by the following register settings.

Register setup:		
R_PCM_MD0	: V_PCM_IDX = 0xC	(R_SH0L register accessible)
R_SH0L	: V_SH0L = 0xF8	(0xF8 = '1111 1000' $\xrightarrow{\text{reverse}}$ '0001 1111')
R_PCM_MD0	: V_PCM_IDX = 0xD	(R_SH0H register accessible)
R_SH0H	: V_SH0H = 0x03	(0x03 = '0000 0011' $\xrightarrow{\text{reverse}}$ '1100 0000')
R_PCM_MD0	: V_PCM_IDX = 0xE	(R_SH1L register accessible)
R_SH1L	: V_SH1L = 0x1F	(0x1F = '0001 1111' $\xrightarrow{\text{reverse}}$ '1111 1000')
R_PCM_MD0	: V_PCM_IDX = 0xF	(R_SH1H register accessible)
R_SH1H	: V_SH1H = 0xF0	(0xF0 = '1111 0000' $\xrightarrow{\text{reverse}}$ '0000 1111')

6.8 GCI/IOM-2 mode

6.8.1 Overview

XHFC-2SU is equipped with a simple GCI controller⁴ (also known as IOMTM-2)⁵ to support interconnection to U-chips, external CODECs or DSPs.

The IOMTM-2 bus is an industrial standard for interconnecting telecommunication microchips considering the requirements of analog applications as well. It has been defined from an international manufacturers group⁶. The GCI⁷ functionality has been implemented in respect to the IOMTM-2 specification [7].

The interconnection between XHFC-2SU and a GCI device uses four wires, typically:

- C4IO: Double bit rate clock
- F0IO: 8 kHz frame signal
- STIO1: Data from XHFC-2SU to the GCI device (can be swapped with STIO2)
- STIO2: Data from the GCI device to XHFC-2SU (can be swapped with STIO1)

6.8.2 GCI frame structure

The GCI frame has a length of 4 bytes and is located at PCM time slots $S..S+3$ with $S = 4 \cdot V_GCI_SL$. GCI uses these PCM time slots in a special way, all other PCM time slots are accessible as usual.

V_GCI_SL must be in the range 0..7 for PCM30, 0..15 for PCM64 or 0..31 for PCM128.

The binary organization of the GCI frame is shown in Figure 6.14. The first two time slots are used for B1- and B2-channel data. The third time slot is occupied by the monitor channel and the last time slot contains D-channel data, command/indication bits and the handshake bits MR and MX.

Figure 6.14 shows the GCI frame from the GCI master's point of view. To distinguish between the handshake bits of the GCI master (transmitter) and the GCI slave (receiver), the handshake bits are indexed like MX_{TX} or MX_{RX} in this document partially.

The PCM slot assigner must be used to allocate any HFC-channel to the first two bytes of the CGI frame. These are B-channels, typically. When the GCI device also carries D-channel data in the GCI time slot #3, the belonging D-channel must be assigned as well. Please note, that these assignments are not automatically done by the GCI controller. Only monitor channel, C/I-channel and handshake bits are automatically assigned and handled from the GCI controller.

⁴GCI = General Circuit Interface

⁵IOMTM-2= ISDN Oriented Modular revision 2, trademark of Infineon Technologies AG

⁶Alcatel (France), Siemens (Germany), Italtel (Italy) and Plessey (UK)

⁷The term GCI is always used in this document and includes IOMTM-2 functionality as well.

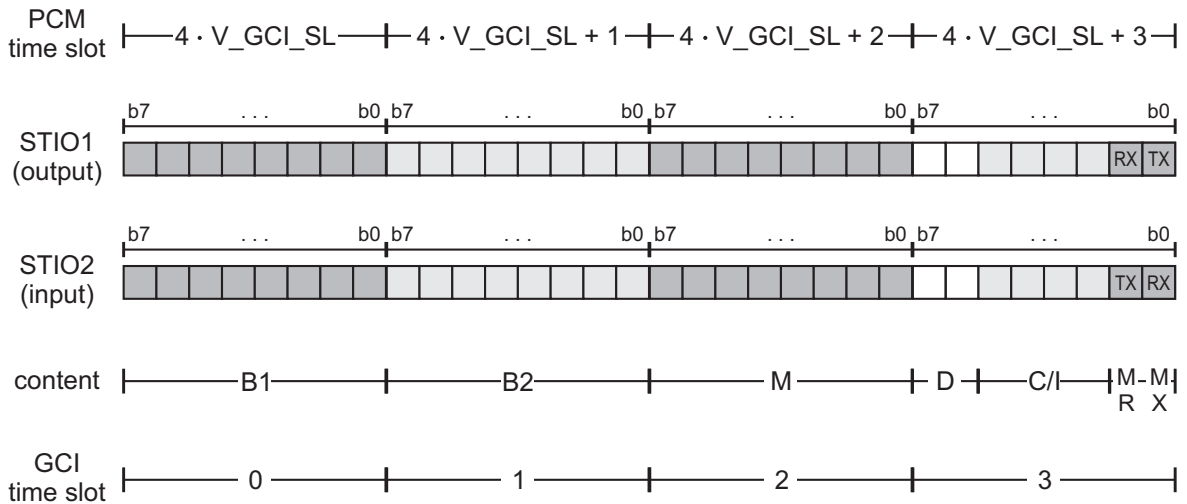


Figure 6.14: Single channel GCI format

Table 6.8: Legend of figure 6.14

Name	Description
B1	B1-channel data
B2	B2-channel data
M	Monitor channel data
D	D-channel data
C/I	Command/indication bits for controlling activation/deactivation and for additional control functions
MR	Handshake bit for the monitor channel (MR _{RX} on STIO1 output and MR _{TX} on STIO2 input)
MX	Handshake bit for the monitor channel (MX _{TX} on STIO1 output and MX _{RX} on STIO2 input)

6.8.3 GCI register programming

6.8.3.1 Enable CGI functionality

The GCI functionality is disabled after XHFC-2SU reset and all time slots of the PCM bus can be assigned to an arbitrary HFC-channel. GCI functionality must be enabled with V_GCI_EN = '1' in register R_GCI_CFG0.

When CGI functionality is not used, received data on time slots 4 · V_GCI_SL + 2 and 4 · V_GCI_SL + 3 are extracted to registers R_MON_RX and R_CI_RX nevertheless. For this reason, the interrupt mask bits V_CI_IRQMSK and V_MON_RX_IRQMSK in register R_MISC_IRQMSK should not be set to '1' to avoid senseless interrupts.

6.8.3.2 Monitor channel

Monitor data for the transmit direction must be written into register R_MON_TX. When the next monitor byte can be written, V_MON_TXR in register R_GCI_STA changes to '1'. Then the next monitor byte can be written into R_MON_TX. A write access to R_MON_TX resets V_MON_TXR to '0'. Before the last monitor byte of a command is written into R_MON_TX, the 'end of command' flag V_MON_END = '1' has to be set in register R_GCI_CFG0.

Furthermore, an interrupt occurs when V_MON_TXR changes to '1' and when the mask bit V_MON_TX_IRQMSK in register R_MISC_IRQMSK is set to '1'. The interrupt event V_MON_TX_IRQ = '1' in register R_MISC_IRQ is set even if the interrupt mask has the value '0'.

If there are no further monitor bytes to be send, the idle pattern 0xFF is transmitted.

A received monitor byte is indicated with V_MON_RXR = '1' in register R_GCI_STA and can be read from register R_MON_RX.

XHFC-2SU can accept either every received monitor byte at once, or after it has been received twice (so-called *double last look criterion*). This can be configured with bit V_MON_DLL in register R_GCI_CFG0. When *double last look criterion* is enabled, the GCI controller waits until a monitor byte has been received twice in two consecutive GCI frames.

The monitor bytes are located in GCI time slot 2 as shown in Figure 6.14. Transmitting and receiving monitor bytes is coordinated by the GCI controller of XHFC-2SU. This procedure is described in detail in Section 6.8.4.

6.8.3.3 Command / indication bits (C/I-channel)

The C/I-channel is used to interchange status information between XHFC-2SU and the connected GCI device. C/I-bits are transmitted continuously in every GCI frame until a new command / indication pattern is present.

Command / indication is used to transmit a command from the GCI master to the connected GCI slave and to receive status information (indication) in opposite direction.

XHFC-2SU transmits the command that is written into bitmap V_GCI_C of register R_CI_TX. When this value is changed, the new command is transmitted in the next GCI time slot 3.

Received indication bits can be read from V_GCI_I. Any change of the indication bits can trigger an interrupt when the interrupt mask bit V_CI_IRQMSK is set to '1' in register R_MISC_IRQMSK. The interrupt event V_CI_IRQ = '1' in register R_MISC_IRQ is set even if the interrupt mask has the value '0'.

The GCI controller does not interpret the C/I-bits. Indication bits must be processed from the host processor.

GCI devices that do not operate on D-channel data, can expand the C/I-channel from 4 bit to 6 bit. XHFC-2SU can be configured to transmit and receive 4 bit (V_MON_CI6 = '0' in register V_MON_CI6) as well as 6 bit (V_MON_CI6 = '1') C/I-channel length.

6.8.3.4 Examples for GCI frame embedding in the PCM data structure

Figure 6.15 shows a typical application where XHFC-2SU is both PCM master and GCI master (most application cases, standard configuration). This means:

- XHFC-2SU feeds PCM clocks C4IO and F0IO
- GCI frame output on STIO1
- GCI frame input on STIO2

The programming procedure for this application example is shown in Table 6.9. Please note that not mentioned bitmap values of the GCI relevant registers remain in their reset state.

GCI time slot 2 does not need to be programmed because the monitor byte is automatically assigned to this time slot when GCI is enabled. C/I-channel and handshake bits MX and MR are also automatically assigned to their time slot.

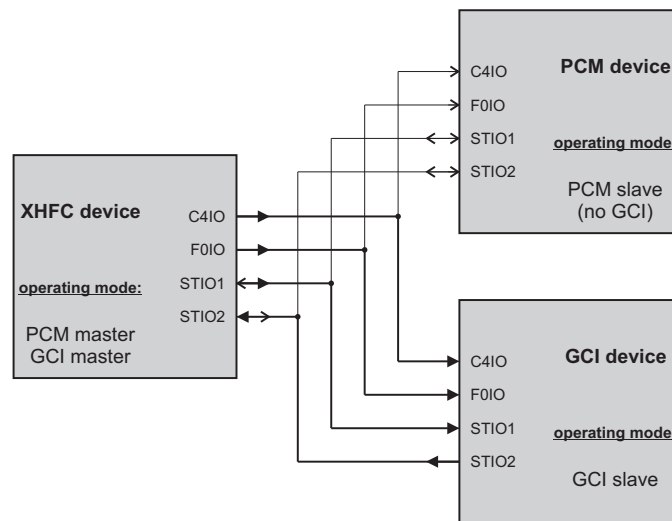


Figure 6.15: GCI application example 1 (XHFC-2SU is PCM master and GCI master)

The second application example shown in Figure 6.16 has only one difference to the first example: XHFC-2SU is configured to PCM slave mode. This means that the PCM clocks are generated from a PCM device in master mode. So the programming procedure is equal to Table 6.9 except that “set PCM to master” has to be changed to “set PCM to slave” ($V_PCM_MD = '0'$).

The last application example 3 shows XHFC-2SU operating in GCI slave mode. XHFC-2SU is PCM slave and GCI slave which means:

- GCI master feeds PCM clocks C4IO and F0IO
- GCI frame input on STIO1
- GCI frame output on STIO2

The programming procedure is shown in Figure 6.17. GCI slave mode is configured with the shown settings for $V_GCI_SWAP_TXHS$, $V_GCI_SWAP_RXHS$ and $V_GCI_SWAP_STIO$ in register R_GCI_CFG0 . Again, not mentioned bitmap values of the GCI relevant registers remain in their reset state.

Please note that register programming must be handled by the host processor connected to the XHFC-2SU microprocessor interface even if XHFC-2SU operates as GCI slave. This includes the monitor data handling as well as the C/I processing.

Table 6.9: Programming procedure for application example 1 according to Figure 6.15 (XHFC-2SU is PCM master and GCI master)

Register	Bit	Value	Function
R_GCI_CFG1	V_GCI_SL	x	Select PCM time slot group for the GCI frame (PCM time slots $4x..4x+3$)
R_PCM_MD0	V_PCM_MD	'1'	set PCM to master
R_SLOT	V_SL_NUM	$4x$	select first GCI time slot $4x$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'10'	set STIO1 to output characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_1	assign HFC-channel y_1 to the selected time slot (normally B1-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'10'	set STIO2 to input characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_1	assign HFC-channel z_1 to the selected time slot (normally $z_1 = y_1$)
R_SLOT	V_SL_NUM	$4x+1$	select second GCI time slot $4x+1$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'10'	set STIO1 to output characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_2	assign HFC-channel y_2 to the selected time slot (normally B2-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'10'	set STIO2 to input characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_2	assign HFC-channel z_2 to the selected time slot (normally $z_2 = y_2$)
R_SLOT	V_SL_NUM	$4x+3$	select fourth GCI time slot $4x+3$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'10'	set STIO1 to output characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_D	assign HFC-channel y_D to the selected time slot (normally D-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'10'	set STIO2 to input characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_D	assign HFC-channel z_D to the selected time slot (normally $z_D = y_D$)
R_GCI_CFG0	V_GCI_EN	'1'	enable GCI function

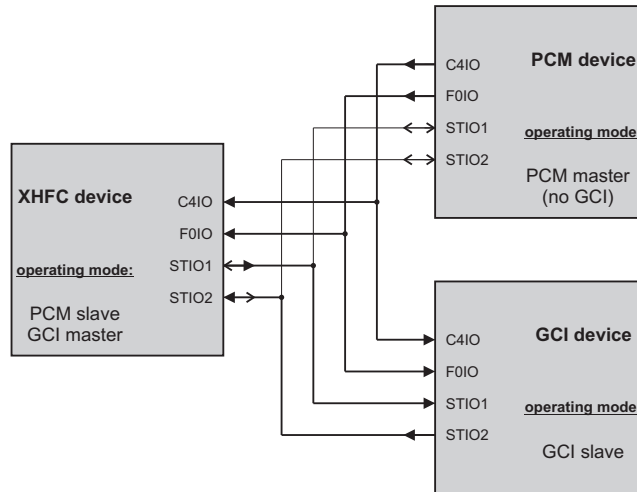


Figure 6.16: GCI application example 2 (XHFC-2SU is PCM slave and GCI master)

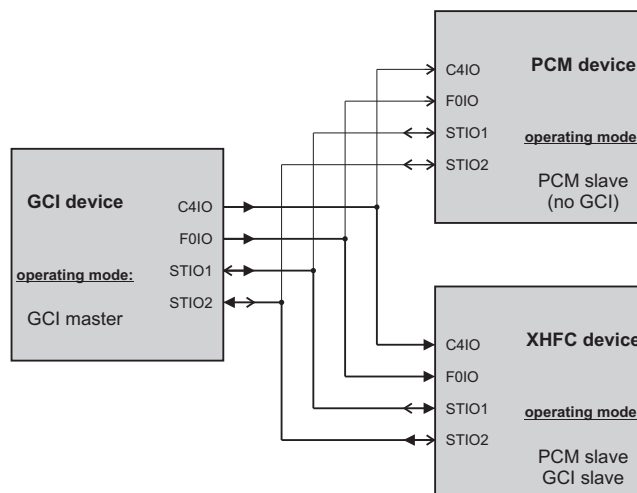


Figure 6.17: GCI application example 3 (XHFC-2SU is PCM slave and GCI slave)

Table 6.10: Programming procedure for application example 3 according to Figure 6.17 (XHFC-2SU is PCM slave and GCI slave)

Register	Bit	Value	Function
R_GCI_CFG1	V_GCI_SL	x	Select PCM time slot group for GCI frame x
R_PCM_MD0	V_PCM_MD	'0'	set PCM to slave
R_SLOT	V_SL_NUM	$4x$	select first GCI time slot $4x$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'11'	set STIO1 to input characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_1	assign HFC-channel y_1 to the selected time slot (normally B1-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'11'	set STIO2 to output characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_1	assign HFC-channel z_1 to the selected time slot (normally $z_1 = y_1$)
R_SLOT	V_SL_NUM	$4x + 1$	select second GCI time slot $4x + 1$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'11'	set STIO1 to input characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_2	assign HFC-channel y_2 to the selected time slot (normally B2-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'11'	set STIO2 to output characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_2	assign HFC-channel z_2 to the selected time slot (normally $z_2 = y_2$)
R_SLOT	V_SL_NUM	$4x + 3$	select fourth GCI time slot $4x + 3$
R_SLOT	V_SL_DIR	'0'	select transmit data direction
A_SL_CFG	V_ROUT	'11'	set STIO1 to input characteristic
A_SL_CFG	V_CH_SDIR	'0'	assign transmit HFC-channel
A_SL_CFG	V_CH_SNUM	y_D	assign HFC-channel y_D to the selected time slot (normally D-channel)
R_SLOT	V_SL_DIR	'1'	select receive data direction
A_SL_CFG	V_ROUT	'11'	set STIO2 to output characteristic
A_SL_CFG	V_CH_SDIR	'1'	assign receive HFC-channel
A_SL_CFG	V_CH_SNUM	z_D	assign HFC-channel z_D to the selected time slot (normally $z_D = y_D$)
R_GCI_CFG0	V_GCI_SWAP_TXHS	'1'	swap handshake bits MR_{TX} and MX_{TX} for GCI transmit direction
R_GCI_CFG0	V_GCI_SWAP_RXHS	'1'	swap handshake bits MR_{RX} and MX_{RX} for GCI receive direction
R_GCI_CFG0	V_GCI_SWAP_STIO	'1'	swap STIO1 and STIO2 for monitor channel and C/I
R_GCI_CFG0	V_GCI_EN	'1'	enable GCI function

6.8.4 GCI protocol

6.8.4.1 XHFC-2SU transmit procedure

Monitor bytes are transmitted and received under the control of the handshake bits MX_{TX} and MR_{TX} . The handshake bits are automatically handled by the GCI controller. Figure 6.18 shows the transmit procedure of two monitor bytes.

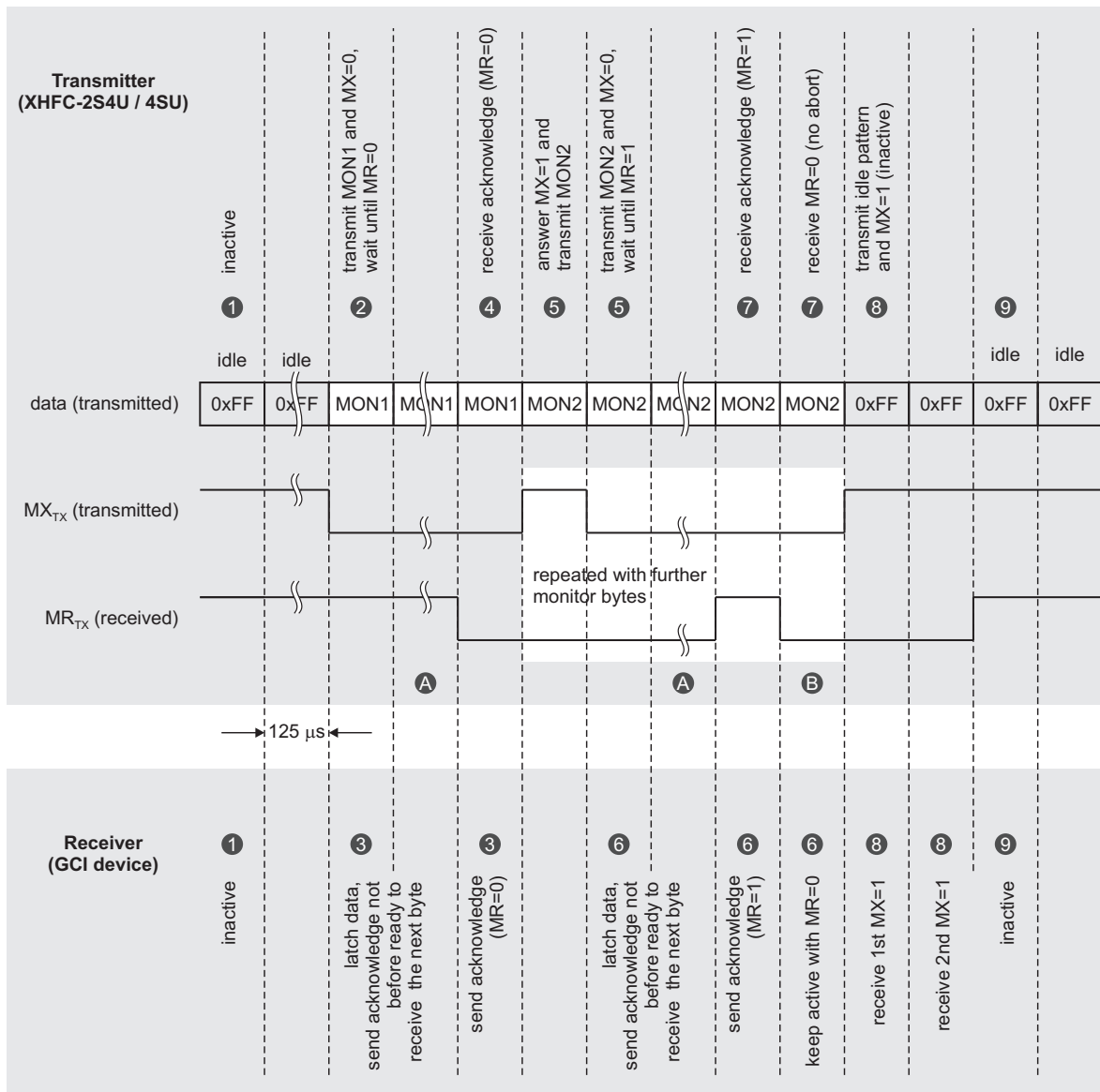


Figure 6.18: GCI protocol for monitor bytes transmission

- 1 Beginning with idle state, no monitor byte is pending, MX_{TX} and MR_{TX} are both '1'. The monitor byte has the value 0xFF in this state.
- 2 A write access to register R_MON_TX starts the transmit sequence.

The first monitor byte is transmitted within the next GCI time slot 2 and MX_{TX} is set to '0'. This monitor byte will be transmitted repeatedly in every time slot 2 until the GCI device acknowledges the byte with $MR_{TX} = '0'$.

As the GCI controller has latched the monitor byte, V_MON_TXR is set to '1'. This means that the host processor can write the next monitor byte into register R_MON_TX .

- ③ The receiver latches the first monitor byte. This can take an arbitrary number of 125 μ s cycles. An acknowledge is send when the receiver is ready for the next byte.
- ④ The transmitter gets acknowledge with $MR_{TX} = '0'$.
- ⑤ XHFC-2SU answers to the acknowledge signal with $MX_{TX} = '1'$ for one cycle when a new monitor byte has been written into register R_MON_TX . MX_{TX} is set to '0' afterwards to keep the monitor channel active. The second monitor byte is transmitted at the same time as $MX_{TX} = '1'$ and is stable until a receiver acknowledge is recognized.
- ⑥ The receiver latches the second monitor byte. This can take an arbitrary number of 125 μ s cycles. An acknowledge signal $MR_{TX} = '1'$ is send for one cycle when the receiver is ready for the next byte.
- ⑦ The transmitter receives acknowledge with $MR_{TX} = '1'$ for one cycle.
- ⑧ The procedure shown with the second monitor byte (⑤ . . ⑦) can be repeated until the whole message has been send. The last monitor byte has to be marked with $V_MON_END = '1'$. The marking must be written before the last monitor byte is stored in register R_MON_TX . Then the transmitter terminates the transmission with continuously $MX_{TX} = '1'$. Idle pattern 0xFF is send. The monitor channel of the transmitter is in inactive state.
- ⑨ When the receiver reads $MX_{TX} = '1'$ for two cycles, its monitor channel goes to inactive state with $MR_{TX} = '1'$. The monitor channel is in idle state now.

Table 6.11 summarizes the rules for the handshake signals MX_{TX} and MR_{TX} when XHFC-2SU transmits a monitor byte sequence.

Table 6.11: Rules for the handshake signals MX_{TX} and MR_{TX} when XHFC-2SU transmits a monitor byte sequence

MX_{TX}	MR_{TX}	Monitor channel state
'1'	'1'	Idle
'0'	'0'	Transmitter and receiver are both active
'0'	'1' (once)	Acknowledge of the 2 nd and following monitor bytes
'0'	'1' (repeated)	Transmitter waits for acknowledge to the 1 st monitor byte
'1' (once)	'0'	Transmitter answers to the acknowledge
'1' (repeated)	'0'	Transmitter terminates the transmission

6.8.4.2 XHFC-2SU receive procedure

The monitor channel is full duplex and operates similar in the opposite direction. XHFC-2SU recognizes idle state with the specified handshake signals and accepts every idle pattern.

Figure 6.19 shows the receive procedure of two monitor bytes with *double last look criterion* enabled.

- ❶ The monitor channel is idle when $MX_{RX} = '1'$ and $MR_{RX} = '1'$. Any received data pattern is ignored.
 - ❷ The GCI transmitter sends the first monitor byte with $MX_{RX} = '0'$.
 - ❸ The first monitor byte is received when $MX_{RX} = '0'$. When the same byte is received again (double last look enabled), it is latched and V_MON_RXR is set to '1'. After R_MON_RX has been read, acknowledge is send in the next 125 μ s cycle with $MR_{RX} = '0'$.
 - ❹ The transmitter receives acknowledge with $MR_{RX} = '0'$.
 - ❺ Answer $MX_{RX} = '1'$ for one cycle and send the next monitor byte.
 - ❻ The next monitor byte is received when $MX_{RX} = '1'$ for one cycle. When the same byte is received again (double last look enabled), it is latched and V_MON_RXR is set to '1'. After R_MON_RX has been read, acknowledge is send in the next 125 μ s cycle with $MR_{RX} = '1'$.
 - ❼ The transmitter receives acknowledge with with $MR_{RX} = '1'$ for one cycle.
 - ❽ The procedure shown with the second monitor byte (❺ . . ❽) can be repeated until the whole message has been send.
- Assumed that the GCI master finished the message transmission, MX_{RX} is set to '1' and remains in this state.
- ❾ When XHFC-2SU receives $MX_{RX} = '1'$ twice, the monitor channel returns to idle state with $MR_{RX} = '1'$. The received byte is no valid monitor byte because MX_{RX} did not return to '0' after one cycle.

The rules for the handshake signals MX_{RX} and MR_{RX} when XHFC-2SU receives a monitor byte sequence are summarized in Table 6.12.

Table 6.12: Rules for the handshake signals MX_{RX} and MR_{RX} when XHFC-2SU receives a monitor byte sequence

MX_{RX}	MR_{RX}	Monitor channel state
'1'	'1'	Idle
'0'	'0'	Transmitter and receiver are both active
'0'	'1' (once)	Receive the 1 st monitor byte, not yet acknowledged
'0'	'1' (repeated)	Receiver terminates the transmission
'1' (once)	'0'	Receiver answers to the acknowledge
'1' (repeated)	'0'	Impossible

6.8.4.3 Receiver abort

The receiver can abort the transmission if data cannot be used or is missing. This is done by $MR = '1'$ for at least 2 cycles. The monitor channel returns to idle state in this case.

When XHFC-2SU operates as GCI master, it can receive an abort message from the GCI slave. This is reported with $V_GCI_ABO = '1'$ in register R_GCI_STA . This bit is reset to '0' with a write access to register R_MON_TX . It is recommended to check V_GCI_ABO every time before writing a new monitor byte.

XHFC-2SU never aborts any message when it is receiver of the monitor channel.

6.9 Register description

6.9.1 Write only registers

R_SLOT	(w)	(Reset group: H, 0, 2)	0x10
PCM time slot selection			
This register is used to select a PCM time slot. Before a PCM slot array register can be accessed, this index register must specify the desired slot number and data direction. Depending on the V_PCM_DR value in register R_PCM_MD1, either 32, 64 or 128 time slots are available for each data direction.			
Bits	Reset value	Name	Description
0	0	V_SL_DIR	PCM time slot data direction '0' = transmit PCM data '1' = receive PCM data
7..1	0x00	V_SL_NUM	PCM time slot number

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PCM_MD0	(w)	(Reset group: H, 0, 2)	0x14
PCM mode, register 0			
Bits	Reset value	Name	Description
0	0	V_PCM_MD	PCM bus mode '0' = slave (pins C4IO and F0IO are inputs) '1' = master (pins C4IO and F0IO are outputs) If no external C4IO and F0IO signal is provided this bit must be set for operation.
1	0	V_C4_POL	Polarity of C4IO clock '0' = pin F0IO is sampled on negative clock transition of C4IO '1' = pin F0IO is sampled on positive clock transition of C4IO
2	0	V_F0_NEG	Polarity of F0IO signal '0' = positive pulse '1' = negative pulse
3	0	V_F0_LEN	Duration of F0IO signal '0' = active for one C4IO clock (244 ns at 4 MHz) '1' = active for two C4IO clocks (488 ns at 4 MHz, early leading edge) The specified signal duration is generated in PCM master mode and it is expected in PCM slave mode.
7..4	0	V_PCM_IDX	Index value to select the register at address 15 At address 15 a so-called multi-register is accessible. 0 = R_SL_SEL0 register accessible 1 = R_SL_SEL1 register accessible 2 = not used 3 = not used 4 = not used 5 = not used 6 = not used 7 = R_SL_SEL7 register accessible 8 = R_MSS0 register accessible 9 = R_PCM_MD1 register accessible 0xA = R_PCM_MD2 register accessible 0xB = R_MSS1 register accessible 0xC = R_SH0L register accessible 0xD = R_SH0H register accessible 0xE = R_SH1L register accessible 0xF = R_SH1H register accessible

R_SL_SELO	(w)	(Reset group: H, 0, 2)	0x15
<p>Slot selection register for pin F1_0</p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0 in register R_PCM_MD0.</p> <p>Note: By setting all 8 bits to '1' pin F1_0 is disabled.</p>			
Bits	Reset value	Name	Description
6..0	0x7F	V_SL_SELO	<p>PCM time slot selection</p> <p>The selected slot number is V_SL_SELO + 1 for F1_0. Slot number 0 is selected with the maximum slot number of the selected PCM speed.</p>
7	1	V_SH_SELO	<p>Shape selection</p> <p>'0' = use shape 0 set by registers R_SH0L and R_SH0H '1' = use shape 1 set by registers R_SH1L and R_SH1H</p>



Important !

For selecting slot 0, the value that has to be written into bitmaps V_SL_SELO and V_SL_SEL1 of registers R_SL_SELO and R_SL_SEL1 depends on the PCM data rate:

PCM data rate	Value
PCM30	0x1F
PCM64	0x3F
PCM128	0x7F

Please note that time slot 0 for PCM128 can only be used with V_SH_SELO = '0' and V_SH_SEL1 = '0' (SHAPE 0) in registers R_SL_SELO and R_SL_SEL1.

R_SL_SEL1	(w)	(Reset group: H, 0, 2)	0x15
Slot selection register for pin F1_1			
This multi-register is selected with bitmap V_PCM_IDX = 1 in register R_PCM_MD0.			
Note: By setting all 8 bits to '1' pin F1_1 is disabled.			
Bits	Reset value	Name	Description
6..0	0x7F	V_SL_SEL1	PCM time slot selection The selected slot number is V_SL_SEL1 +1 for F1_1. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	V_SH_SEL1	Shape selection '0' = use shape 0 set by registers R_SH0L and R_SH0H '1' = use shape 1 set by registers R_SH1L and R_SH1H

R_SL_SEL7	(w)	(Reset group: H, 0, 2)	0x15
Slot selection register for signal F1_7			
This multi-register is selected with bitmap V_PCM_IDX = 7 in register R_PCM_MD0.			
The F1_7 signal is only internally available. It can be used to shift the synchronization signal for Universal ISDN Ports in NT mode.			
Bits	Reset value	Name	Description
6..0	0x7F	V_SL_SEL7	PCM time slot selection The selected slot number is V_SL_SEL7 +1 for the F1_7 signal. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	0	(reserved)	Must be '0'.

R_MSS0	(w)	(Reset group: H, 0, 2)	0x15
PCM multiframe/superframe synchronization mode, register 0			
This multi-register is selected with bitmap V_PCM_IDX = 8 in register R_PCM_MD0.			
Bits	Reset value	Name	Description
0	0	V_MSS_MOD	F0IO pulse modification The F0IO pulse duration can be changed to generate the multiframe/superframe synchronization signal. '0' = normal operation '1' = F0IO pulse is lengthened by one C4IO clock to indicate the start of the multiframe/superframe (delayed trailing edge)
1	0	V_MSS_MOD_REP	F0IO modification repetition rate The repetition rate of the modified F0IO signal can be selected. '0' = once every 40 PCM frames '1' = once every 8 PCM frames
2	0	V_MSS_SRC_EN	Enable external multiframe/superframe synchronization signal '0' = disabled '1' = enabled
3	0	V_MSS_SRC_GRD	Synchronization guard count The multiframe/superframe synchronization signal is detected not before 7 or 39 PCM frames. '0' = not before 39 PCM frames '1' = not before 7 PCM frames
4	0	V_MSS_OUT_EN	Enable line interface synchronization signal for the multiframe/superframe '0' = the synchronization signal is disabled '1' = the synchronization signal is passed to the line interfaces
5	0	V_MSS_OUT_REP	Repetition rate of the line interface synchronization signal for the multiframe/superframe The line interface synchronization signal is generated either every 8th or 40th PCM frame. '0' = every 40th PCM frame '1' = every 8th PCM frame (can only be used in pure Up environments)

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7..6	0	V_MSS_SRC	Multiframe / superframe synchronization source One of the following signals can be chosen as synchronization source. '00' = pin F0IO '01' = SYNC_I '10' = STIO1 '11' = internal signal MSS_SYNC_I

R_PCM_MD1	(w)	(Reset group: H, 0, 2)	0x15
PCM mode, register 1			
This multi-register is selected with bitmap V_PCM_IDX = 9 in register R_PCM_MD0.			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	V_PCM_OD	Characteristic of the PCM output lines '0' = STIO1 and STIO2 have push / pull characteristic '1' = STIO1 and STIO2 have open drain characteristic (pull up resistor required)
3..2	0	V_PLL_ADJ	DPLL adjust speed '00' = C4IO clock is adjusted in the last time slot of the PCM frame 4 times by one clock cycle of the PCM clock f_{PCM} (81.4 ns all 125 μ s, 651 ppm) '01' = C4IO clock is adjusted in the last time slot of the PCM frame 3 times by one clock cycle of the PCM clock f_{PCM} (61.0 ns all 125 μ s, 489 ppm) '10' = C4IO clock is adjusted in the last time slot of the PCM frame twice by one clock cycle of the PCM clock f_{PCM} (40.7 ns all 125 μ s, 326 ppm) '11' = C4IO clock is adjusted in the last time slot of the PCM frame once by one clock cycle of the PCM clock f_{PCM} (20.3 ns all 125 μ s, 163 ppm)
5..4	0	V_PCM_DR	PCM data rate '00' = 2 MBit/s (C4IO is 4.096 MHz, 32 time slots) '01' = 4 MBit/s (C4IO is 8.192 MHz, 64 time slots) '10' = 8 MBit/s (C4IO is 16.384 MHz, 128 time slots) '11' = 0.75 MBit/s (C4IO is 1.536 MHz, 12 time slots) Every time slot exists in transmit and receive data direction.
6	0	V_PCM_LOOP	PCM test loop When this bit is set, the PCM output data is looped to the PCM input data internally for all PCM time slots. Note: When this bit is set (internal PCM loop), it is not allowed to set bit V_PCM_OD in register R_PCM_MD1 as well.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	V_PCM_SMPL	PCM receive sample point '0' = sample point at middle of PCM bit cell (normal operation) '1' = sample point at 3/4 of PCM bit cell

R_PCM_MD2	(w)	(Reset group: H, 0, 2)	0x15
PCM mode, register 2			
This multi-register is selected with bitmap V_PCM_IDX = 0xA in register R_PCM_MD0.			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	V_SYNC_OUT1	SYNC_O output signal selection V_SYNC_OUT2 is also used for synchronization selection. '0' = SYNC_O signal is either SYNC_I or the received synchronization pulse FSC_RX (see register R_SU_SYNC for synchronization source selection) '1' = SYNC_O signal is either 512 kHz from the PLL or the received multiframe / superframe synchronization pulse
2	0	V_SYNC_SRC	PCM PLL synchronization source selection '0' = line interface (see R_SU_SYNC for further synchronization configuration) '1' = SYNC_I input (8 kHz)
3	0	V_SYNC_OUT2	SYNC_O output signal selection V_SYNC_OUT1 is also used for synchronization selection. '0' = SYNC_O signal is either the received synchronization pulse FSC_RX (see register R_SU_SYNC for synchronization source selection) or 512 kHz from the PLL '1' = SYNC_O signal is either SYNC_I or the received multiframe / superframe synchronization pulse
4	0	V_C2O_EN	Enable C2IO output signal '0' = C2IO output is disabled '1' = C2IO output is enabled when also V_C2I_EN = '0'
5	0	V_C2I_EN	Single Clock on C2IO is used as PCM clock (master and slave mode) '0' = PCM data controller gets C4IO input clock '1' = PCM data controller gets C2IO input clock
6	0	V_PLL_ICR	Increase PCM frame time This bit is only valid if V_PLL_MAN is set. '0' = PCM frame time is reduced as selected by bitmap V_PLL_ADJ in register R_PCM_MD1 '1' = PCM frame time is increased as selected by bitmap V_PLL_ADJ in register R_PCM_MD1

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	V_PLL_MAN	Manual PLL adjustment '0' = PCM PLL is automatically adjusted to the SYNC_I signal or to the line interface synchronization pulse (depending on V_SYNC_OUT2 setting) '1' = PCM PLL is manually adjusted according to V_PLL_ICR. This can be used to make synchronization by software if no synchronization source is available. Note: Manual PLL adjustment is automatically disabled when a synchronization pulse is available.

R_MSS1	(w)	(Reset group: H, 0, 2)	0x15
PCM multiframe/superframe synchronization mode, register 1			
This multi-register is selected with bitmap V_PCM_IDX = 0xB in register R_PCM_MD0.			
Bits	Reset value	Name	Description
2..0	0	V_MSS_OFFS	Synchronization source offset The offset between the synchronization frame counter and the generated synchronization signal is specified in number of PCM frames. The value can be chosen in the range -4...+3. '000' = no offset '001' = 1 PCM frame '010' = 2 PCM frames '011' = 3 PCM frames '100' = -4 PCM frames '101' = -3 PCM frames '110' = -2 PCM frames '111' = -1 PCM frames
3	0	V_MS_SSYNC1	Multiframe/superframe single synchronization pulse '0' = a repetitive synchronization signal is generated every 8 or 40 PCM frames '1' = the multiframe/superframe synchronization input signal is directly used (this is normally used to generate a single MSS_SYNC_O pulse) Note: When this bit is set to '1', usually also V_MS_SSYNC2 has to be set.
7..4	0	V_MSS_DLY	Multiframe/superframe delay The position of multiframe/superframe data is always related to the regular F0IO signal. This bitmap specifies the delay in number of C4IO clock pulses. 0 = 1 C4IO pulse delay 1 = 2 C4IO pulses delay ... 15 = 16 C4IO pulses delay

R_SH0L	(w)	(Reset group: H, 0, 2)	0x15
<p>CODEC enable signal F1_0, low byte</p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0xC in register R_PCM_MD0.</p>			
Bits	Reset value	Name	Description
7..0	0x00	V_SH0L	Shape bits 7..0 Every bit is used for 1/2 C4IO clock cycle.

R_SH0H	(w)	(Reset group: H, 0, 2)	0x15
<p>CODEC enable signal F1_0, high byte</p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0xD in register R_PCM_MD0.</p>			
Bits	Reset value	Name	Description
7..0	0x00	V_SH0H	Shape bits 15..8 Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH0H defines the value for the rest of the period.

R_SH1L	(w)	(Reset group: H, 0, 2)	0x15
<p>CODEC enable signal F1_1, low byte</p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0xE in register R_PCM_MD0.</p>			
Bits	Reset value	Name	Description
7..0	0x00	V_SH1L	Shape bits 7..0 Every bit is used for 1/2 C4IO clock cycle.

R_SH1H	(w)	(Reset group: H, 0, 2)	0x15
<p>CODEC enable signal F1_1, high byte</p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0xF in register R_PCM_MD0.</p>			
Bits	Reset value	Name	Description
7..0	0x00	V_SH1H	<p>Shape bits 15..8</p> <p>Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH1H defines the value for the rest of the period.</p>

R_SU_SYNC	(w)	(Reset group: H, 0, 3)	0x17
ST/Up synchronization source			
This register selects the synchronization source for the internal or external PCM clock PLL.			
Bits	Reset value	Name	Description
2..0	0	V_SYNC_SEL	Synchronization source selection Any line interface or the SYNC_I input signal can be selected as synchronization source. A line interface can be used as synchronization source only if it is in TE mode. '000' = source is line interface 0 '001' = source is line interface 1 '010', '011' = not allowed '100' = source is SYNC_I signal '101' .. '111' = not allowed
3	0	V_MAN_SYNC	Automatically synchronization source selection '0' = automatic synchronization source selection. A TE which is synchronized to the incoming S/T signal (e.g. state F6 or F7) is chosen as synchronization source and V_SYNC_SEL is ignored. '1' = manual synchronization source selection. V_SYNC_SEL is used for synchronization source. The current synchronization source can be read from V_RD_SYNC_SRC in register R_BERT_STA.
4	0	V_AUTO_SYNCI	Enable SYNC_I as synchronization signal In addition to the line interface synchronization pulse FSC_RX, SYNC_I can be taken for synchronization. '0' = SYNC_I is not used for synchronization '1' = SYNC_I is automatically used for synchronization if no FSC_RX pulse is detected
5	0	V_D_MERGE_TX	Both D-channels are taken from one byte in TX direction.
6	0	V_E_MERGE_RX	Both E-channels are combined into one Byte in RX direction.
7	0	V_D_MERGE_RX	Both D-channels are combined into one Byte in RX direction.

R_CI_TX	(w)	(Reset group: H, 0, 2)	0x28
C/I-channel of the GCI interface			
Bits	Reset value	Name	Description
5..0	0	V_GCI_C	Command bits of the C/I-channel These bits are continuously send in the C/I-channel. C/I-channel length can either be 4 bit or 6 bit according to V_MON_CI6 setting. Bits [5,4] are ignored when the C/I-channel length is 4 bit.
7..6	0	(reserved)	Must be '00'.

R_GCI_CFG0		(w)	(Reset group: H, 0, 2)	0x29
GCI interface configuration, register 0				
Bits	Reset value	Name	Description	
0	0	V_MON_END	End of command flag for the monitor channel The transmitted monitor command ends after the next monitor byte.	
1	0	V_MON_SLOW	Transmission speed of the monitor channel '0' = the next monitor byte is sent immediately after the receive handshake bit gets high '1' = the next monitor byte is sent after a high-to-low transition of the receive handshake bit	
2	0	V_MON_DLL	Enable double last look criterion '0' = a received monitor byte is accepted at once '1' = a monitor byte is accepted only if it is received twice	
3	0	V_MON_C16	Expand C/I-channel width to 6 bits '0' = 4 bit C/I-channel '1' = 6 bit C/I-channel (no D-channel transmission)	
4	0	V_GCI_SWAP_TXHS	Swap handshake bits for transmitted monitor bytes '0' = normal operation (XHFC-2SU is GCI master) '1' = handshake bits MR _{TX} and MX _{TX} are swapped (XHFC-2SU is GCI slave) Note: This bit must be set when XHFC-2SU operates in GCI slave mode. Bits V_GCI_SWAP_RXHS and V_GCI_SWAP_STIO in this register must also be set in this case.	
5	0	V_GCI_SWAP_RXHS	Swap handshake bits for received monitor bytes '0' = normal operation (XHFC-2SU is GCI master) '1' = handshake bits MR _{RX} and MX _{RX} are swapped (XHFC-2SU is GCI slave) Note: This bit must be set when XHFC-2SU operates in GCI slave mode. Bits V_GCI_SWAP_TXHS and V_GCI_SWAP_STIO in this register must also be set in this case.	

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
6	0	V_GCI_SWAP_STIO	<p>Swap STIO pins for monitor bytes and C/I-channel</p> <p>'0' = normal operation (XHFC-2SU is GCI master) '1' = swap STIO1 and STIO2 for monitor bytes and C/I-channel (XHFC-2SU is GCI slave)</p> <p>Note: This bit must be set when XHFC-2SU operates in GCI slave mode. Bits V_GCI_SWAP_TXHS and V_GCI_SWAP_RXHS in this register must also be set in this case.</p>
7	0	V_GCI_EN	<p>Enable GCI interface function</p> <p>When this bit is set to '1', monitor bytes are transmitted and received on GCI time slot 2, C/I-channel and handshake bits are mapped to GCI time slot 3 and I/O direction of the pins STIO1 and STIO2 is configured according to bit V_GCI_SWAP_STIO. GCI time slots are chosen according to the R_GCI_CFG1 selection.</p>

R_GCI_CFG1		(w)	(Reset group: H, 0, 2)	0x2A
GCI interface configuration, register 1				
Bits	Reset value	Name	Description	
4..0	0	V_GCI_SL	<p>Slot group of the GCI interface The GCI interface allocates four PCM time slots $4 \cdot V_GCI_SL \dots 4 \cdot V_GCI_SL + 3$. Monitor bytes, C/I-channel and handshake bits are automatically mapped to their dedicated position. B1-, B2- and D-channels must be assigned according to the GCI needs with normal PCM slot assigner programming (registers R_SLOT and A_SL_CFG).</p> <p>0 = time slots 0..3 1 = time slots 4..7 ... 7 = time slots 28..31 8 = time slots 32..35 (normally not used, only for PCM64 and PCM128) ... 16 = time slots 32..35 (normally not used, only for PCM128) ... 31 = time slots 124..127 (normally not used, only for PCM128)</p> <p>Note: Normally, GCI requires PCM30 data rate which results in valid bitmap values 0..7. When PCM64 or PCM128 is used with GCI functionality, 0..15 or 0..31 can be chosen.</p>	
7..5	0	(reserved)	Must be '000'.	

R_MON_TX		(w)	(Reset group: H, 0, 2)	0x2B
Monitor data byte				
Bits	Reset value	Name	Description	
7..0	0	V_MON_TX	Monitor data byte to be transmitted	

6.9.2 Read only registers

R_F0_CNTL	(r)	(Reset group: H, 0, 1)	0x18
F0IO pulse counter, low byte			
Bits	Reset value	Name	Description
7..0	0x00	V_F0_CNTL	Low byte (bits 7..0) of the 125 μs time counter This register should be read first to latch the value of register R_F0_CNTH.

R_F0_CNTH	(r)	(Reset group: H, 0, 1)	0x19
F0IO pulse counter, high byte			
Bits	Reset value	Name	Description
7..0	0x00	V_F0_CNTH	High byte (bits 15..8) of the 125 μs time counter The low byte should be read first (see register R_F0_CNTL)

R_SL_MAX	(r)	(Reset group: -)	0x1D
Number of PCM time slots			
Bits	Reset value	Name	Description
7..0		V_SL_MAX	Number of last PCM time slot PCM time slots are numbered 0..V_SL_MAX. Master mode: Four values are possible due to V_PCM_DR setting in register R_PCM_MD1 (31, 63, 127 or 11) Slave mode: Any value in the range 0..127 is possible due to PCM master configuration

R_CI_RX		(r)	(Reset group: -)	0x28
CI-channel of the GCI interface				
Bits	Reset value	Name	Description	
5..0		V_GCI_I	Indication bits of the C/I-channel These bits are continuously received in the C/I-channel. C/I-channel length can either be 4 bit or 6 bit according to V_MON_C16 setting. Bits [5,4] are always '00' when the C/I-channel length is 4 bit.	
7..6		(reserved)		

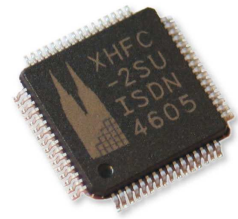
R_GCI_STA	(r)	(Reset group: H, 0, 2)	0x29
Status register of the GCI interface			
Bits	Reset value	Name	Description
0	0	V_MON_RXR	Monitor receiver ready A monitor byte has been received and can be read from register R_MON_RX.
1	1	V_MON_TXR	Monitor transmitter ready A monitor byte has been send and the next byte can be written into register R_MON_TX.
2	1	V_GCI_MX	Status of MX handshake bit This bit shows the current status of the MX _{RX} handshake bit (normally not used, for test purposes only).
3		V_GCI_MR	Status of MR handshake bit This bit shows the current status of the MR _{RX} handshake bit (normally not used, for test purposes only).
4	0	V_GCI_RX	Receiving monitor byte '0' = monitor byte transmission from GCI device to XHFC-2SU is idle '1' = monitor byte transmission from GCI device to XHFC-2SU is currently active
5	0	V_GCI_ABO	Receiver abort '0' = normal operation '1' = receiver aborted the monitor byte transmission
7..6		(reserved)	

R_MON_RX	(r)	(Reset group: -)	0x2A
Monitor data byte			
Bits	Reset value	Name	Description
7..0		V_MON_RX	Received monitor data byte

6.9.3 Read/write register

A_SL_CFG [SLOT]		(r*/w)	(Reset group: H, 0, 2)	0xD0
HFC-channel assignment for the selected PCM time slot and PCM output buffer configuration				
With this register a HFC-channel can be assigned to the selected PCM time slot. Additionally, the PCM buffers can be configured.				
Before writing this array register the PCM time slot must be selected by register R_SLOT.				
Bits	Reset value	Name	Description	
0	0	V_CH_SDIR	HFC-channel data direction '0' = HFC-channel for transmit data '1' = HFC-channel for receive data	
4..1	0x00	V_CH_SNUM	HFC-channel number (0..15)	
5		(reserved)	Must be '0' when written.	
7..6	0	V_ROUT	PCM output buffer configuration For transmit time slots: '00' = data transmission from HFC-channel disabled, output buffers disabled '01' = loop PCM data internally, output buffers disabled '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled For receive time slots: '00' = data transmission to HFC-channel disabled, input data is ignored '01' = loop PCM data internally '10' = receive data from STIO2 '11' = receive data from STIO1 Note: When this bitmap is set to '01' (internal PCM loop), it is not allowed to set bit V_PCM_OD in register R_PCM_MD1 as well.	

(See Section 2.2.3.2 on page 45 for details on Read* access.)



Chapter 7

Pulse width modulation (PWM) outputs

Table 7.1: *Overview of the XHFC-2SU PWM registers*

Address	Name	Page
0x1E	R_PWM_CFG	277
0x38	R_PWM0	277
0x39	R_PWM1	278
0x46	R_PWM_MD	278

7.1 Overview

XHFC-2SU has two PWM output lines PWM0 and PWM1 with programmable output characteristic.

The output lines can be configured as open drain, open source and push / pull by setting V_PWM0_MD respectively V_PWM1_MD in register R_PWM_MD.

7.2 Standard PWM usage

The duty cycle of the output signals can be set in registers R_PWM0 and R_PWM1. The register value defines the number of clock periods where the output signal is high during the cycle time

$$T = 256 \cdot \frac{2^3 \cdot V_PWM_FRQ}{f_{SYS}} .$$

The variable duty cycle

$$\frac{t_{high}}{t_{low}} = \frac{i}{256 - i} , i = \text{value of R_PWM0 or R_PWM1}$$

of the PWM output pins can be set from 1:255 to 255:1. The register value 0 generates an output signal which is permanently low. The duty cycle 1:1 is achieved with $i = 128$.

There are always i pulses within the period T . Each pulse-width is a multiple $1/f_{SYS}$. Due to the setup values, different pulse-width might occur. Pulses with longer or shorter width than the mostly appearing width are distributed through the whole period.

The output signal of the PWM unit can be used for analog settings by using an external RC filter which generates a voltage that can be adapted by changing the PWM register value.

7.3 Alternative PWM usage

The PWM output lines can be programmed to generate a 16 kHz signal. This signal can be used as analog metering pulse for POTS interfaces. Each PWM output line can be switched to 16 kHz signal by setting V_PWM0_16KHZ or V_PWM1_16KHZ in register R_PWM_CFG. In this case the output characteristic is also determined by the R_PWM_MD register settings.

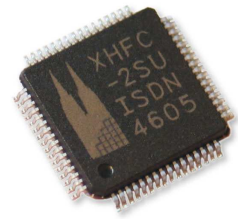
7.4 Register description

R_PWM_CFG		(w)	(Reset group: H, 0)	0x1E
Bits	Reset value	Name	Description	
3..0	0	(reserved)	Must be '0000'.	
4	0	V_PWM0_16KHZ	16 kHz signal on PWM0 '0' = normal PWM output due to the R_PWM0 register setting '1' = 16 kHz output on PWM0	
5	0	V_PWM1_16KHZ	16 kHz signal on PWM1 '0' = normal PWM output due to the R_PWM1 register setting '1' = 16 kHz output on PWM1	
7..6	0	V_PWM_FRQ	The PWM frequency is derived from the system clock f_{SYS} and can be reduced with this bitmap. '00' = PWM frequency is f_{SYS} '01' = PWM frequency is $f_{SYS} / 8$ '10' = PWM frequency is $f_{SYS} / 64$ '11' = PWM frequency is $f_{SYS} / 512$	

R_PWM0		(w)	(Reset group: H, 0)	0x38
Modulator register for pin PWM0				
Bits	Reset value	Name	Description	
7..0	0x00	V_PWM0	PWM duty cycle The value specifies the number of clock periods where the output signal of PWM0 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high	

R_PWM1	(w)	(Reset group: H, 0)	0x39
Modulator register for pin PWM1			
Bits	Reset value	Name	Description
7..0	0x00	V_PWM1	PWM duty cycle The value specifies the number of clock periods where the output signal of PWM1 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high

R_PWM_MD	(w)	(Reset group: H, 0)	0x46
PWM output mode register			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	V_WAK_EN	Enable WAKEUP pin '0' = disable WAKEUP pin '1' = enable WAKEUP pin
3..2	0	(reserved)	Must be '00'.
5..4	0	V_PWM0_MD	Output buffer configuration for pin PWM0 '00' = PWM output tristate (disable) '01' = PWM push/pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only
7..6	0	V_PWM1_MD	Output buffer configuration for pin PWM1 '00' = PWM output tristate (disable) '01' = PWM push/pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only



Chapter 8

Bit Error Rate Test (BERT)

Table 8.1: *Overview of the XHFC-2SU BERT registers*

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x1B	R_BERT_WD_MD	284	0x17	R_BERT_STA	285
			0x1A	R_BERT_ECL	285
			0x1B	R_BERT_ECH	286

8.1 BERT functionality

Bit Error Rate Test (BERT) is a very important test for communication lines. The bit error rate should be as low as possible. Increasing bit error rate is an early indication of a malfunction of components or the communication wire link itself.

XHFC-2SU includes a high performance pseudo random bit generator (PRBG) and a pseudo random bit receiver with automatic synchronization capability. The error rate can be checked by the also implemented Bit Error counter (BERT counter).



Please note !

Transparent mode must be selected for *Bit Error Rate Test*.

8.2 BERT transmitter

The PRBG can be set to a variety of different pseudo random bit patterns. Continuous '0', continuous '1' or pseudo random bit patterns with one of 6 selectable sequence length's from $2^9 - 1$ bit to $2^{23} - 1$ bit can be configured with bitmap `V_PAT_SEQ` in register `R_BERT_WD_MD`. All bit sequences are defined in the ITU-T O.150 [11] and O.151 [10] specifications.

The BERT patterns are passed through the HFC-channel assigner if `V_BERT_EN = '1'` in register `A_FIFO_CTRL[FIFO]`. For this reason, either a FIFO-to-ST/ U_p or a FIFO-to-PCM configuration must be selected. Furthermore, the allocated FIFO must be enabled to switch on the data path.

BERT patterns are generated if at least one FIFO has its bit `V_BERT_EN` set to '1'. When more than one transmit FIFO are using BERT patterns, all these patterns are generated from the same pseudo random generator. They are distributed to the FIFOs in the order of the FIFO processing sequence (see Section 3.2.3 on page 78).

Subchannel processing can be used together with the *Bit Error Rate Test*. Then the number of bits taken from the PRBG is `V_BIT_CNT`.



Please note !

To test a connection and the error detection capability of the BERT error counter, a BERT error can be generated on the receiver side of an ST/ U_p link. Setting bit `V_BERT_ERR` in register `R_BERT_WD_MD` generates one wrong BERT bit in the outgoing data stream. `V_BERT_ERR` is automatically cleared afterwards.

8.3 BERT receiver

The BERT receiver has an automatic synchronization capability. When the incoming bit stream is synchronized with the PRBG, bit `V_BERT_SYNC` in register `R_BERT_STA` is set to '1'.

A 16 bit BERT error counter is available in registers `R_BERT_ECL` and `R_BERT_ECH`. The low

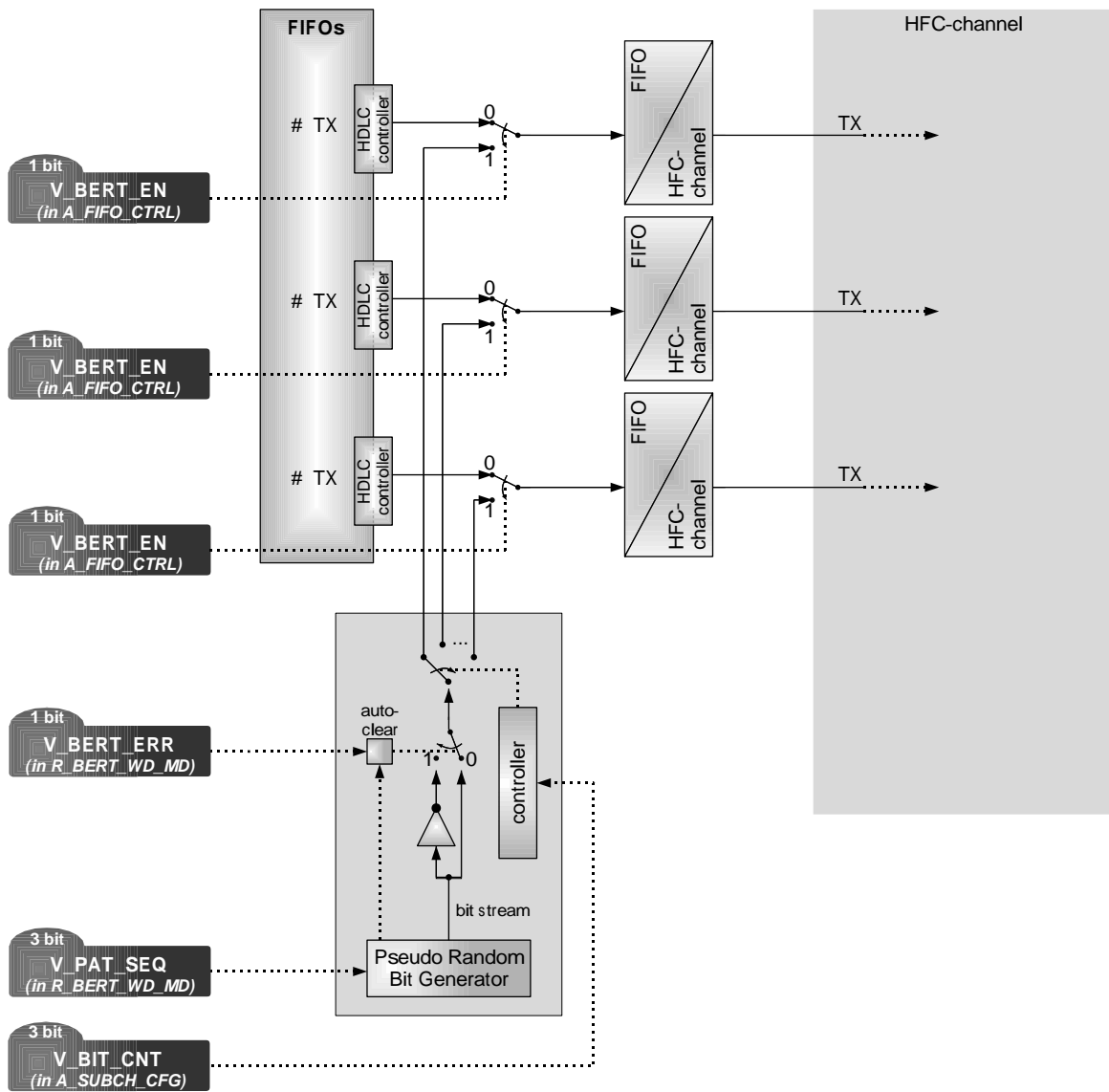


Figure 8.1: BERT transmitter block diagram

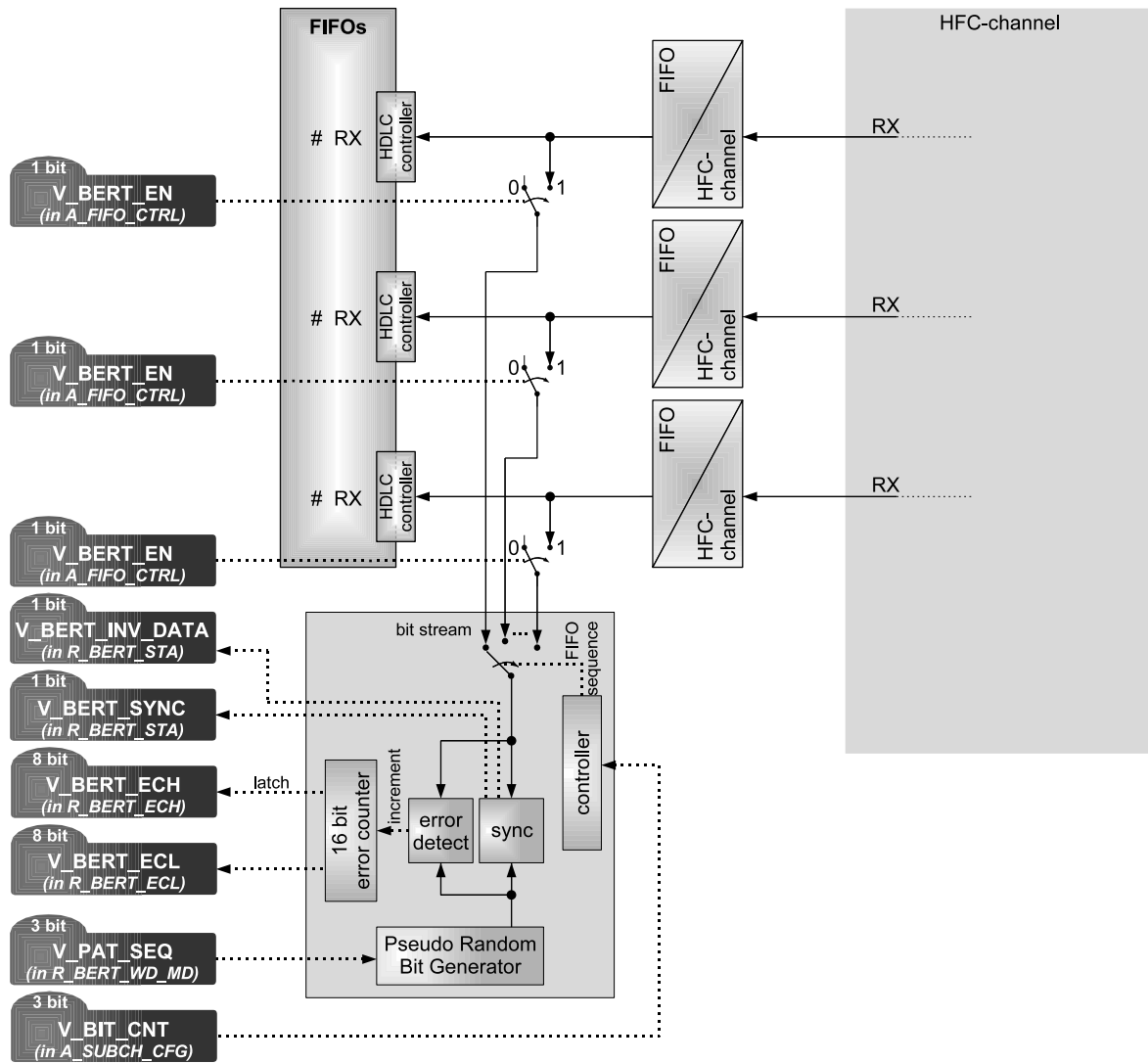


Figure 8.2: BERT receiver block diagram

byte R_BERT_ECL should be read first to latch the high byte. Then the high byte can be read from register R_BERT_ECH. A read access to the low byte R_BERT_ECL clears the 16 bit counter.

The BERT procedure should first wait for the synchronization state. After this, the BERT error counter should be cleared by reading register R_BERT_ECL.

Received BERT data is passed through the HFC-channel assigner if V_BERT_EN = '1' in register A_FIFO_CTRL[FIFO]. For this reason, either a FIFO-to-ST/U_p or a FIFO-to-PCM configuration must be selected. Furthermore, the allocated FIFO must be enabled to switch on the data path. Received BERT data is stored in the FIFO but it needs not to be read out. Received BERT data is collected from all FIFOs which have V_BERT_EN = '1' in the order of the FIFO processing sequence (see Section 3.2.3 on page 78).

Subchannel processing can be used together with the *Bit Error Rate Test*. Then V_BIT_CNT bits taken passed to the BERT receiver.

Inverted BERT data is automatically detected and can be checked with V_BERT_INV_DATA in register R_BERT_STA.

The automatic synchronization works only if the bit error rate is less than $4 \cdot 10^{-2}$. Synchronization state will not be achieved with a higher error rate. It is lost when many bit errors occur during a short time period. In this case, the re-synchronization starts automatically and a high bit error counter value indicates that a re-synchronization might has happened.

8.4 Register description

8.4.1 Write only registers

R_BERT_WD_MD	(w)	(Reset group: H, 0)	0x1B
Bit error rate test (BERT) and watchdog mode			
Bits	Reset value	Name	Description
2..0	0	V_PAT_SEQ	Continuous '0' / '1' or pseudo random pattern sequence for BERT '000' = continuous '0' pattern '001' = continuous '1' pattern '010' = sequence length $2^9 - 1$ bits '011' = sequence length $2^{10} - 1$ bits '100' = sequence length $2^{15} - 1$ bits '101' = sequence length $2^{20} - 1$ bits '110' = sequence length $2^{20} - 1$ bits, but maximal 14 bits are zero '111' = pseudo random pattern seq. $2^{23} - 1$ Note: These sequences are defined in ITU-T O.150 and O.151 specifications.
3	0	V_BERT_ERR	BERT error Generates one error bit in the BERT data stream '0' = no error generation '1' = generates one error bit This bit is automatically cleared.
4	0	(reserved)	Must be '0'.
5	0	V_AUTO_WD_RES	Automatic watchdog timer reset '0' = watchdog is only reset by V_WD_RES '1' = watchdog is reset after every access to the chip
6	0	V_WD_EN	Watchdog timer enable '0' = watchdog timer is disabled '1' = watchdog timer is enabled
7	0	V_WD_RES	Watchdog timer reset '0' = no action '1' = manual watchdog timer reset This bit is automatically cleared.

8.4.2 Read only registers

R_BERT_STA	(r)	(Reset group: H, 0, 1)	0x17
Bit error rate test status			
Bits	Reset value	Name	Description
2..0	0	V_RD_SYNC_SRC	Synchronization source selection Reports which line interface is used as synchronization source. Every line interface can be either in S/T or Up mode. The SYNC_I signal can be used as synchronization source, alternatively. '000' = line interface 0 '001' = line interface 1 '010' = not used '011' = not used '100' = SYNC_I signal '101' .. '111' = not used
3		(reserved)	
4	0	V_BERT_SYNC	BERT synchronization status '0' = BERT not synchronized to input data '1' = BERT synchronized to input data
5	0	V_BERT_INV_DATA	BERT data inversion '0' = BERT receives normal data '1' = BERT receives inverted data
7..6		(reserved)	

R_BERT_ECL	(r)	(Reset group: H, 0, 1)	0x1A
BERT error counter, low byte			
Bits	Reset value	Name	Description
7..0	0x00	V_BERT_ECL	Bits 7..0 of the BERT error counter This register should be read first to latch the value of register R_BERT_ECH. Note: The BERT counter is cleared after reading this register.

R_BERT_ECH	(r)	(Reset group: H, 0, 1)	0x1B
BERT error counter, high byte			
Bits	Reset value	Name	Description
7..0	0x00	V_BERT_ECH	Bits 15..8 of the BERT error counter Note: Low byte should be read first (see register R_BERT_ECL).



Chapter 9

Clock, PLL, reset, interrupt, timer and watchdog

Table 9.1: Overview of clock, PLL, reset, timer and watchdog registers

Write only registers:			Read only registers:			Read / write registers:		
Address	Name	Page	Address	Name	Page	Address	Name	Page
0x00	R_CIRM	305	0x10	R_IRQ_OVIEW	313	0x51	R_PLL_P	322
0x02	R_CLK_CFG	307	0x11	R_MISC_IRQ	315	0x52	R_PLL_N	322
0x11	R_MISC_IRQMSK	308	0x12	R_SU_IRQ	316	0x53	R_PLL_S	322
0x12	R_SU_IRQMSK	309	0x1C	R_STATUS	317			
0x13	R_IRQ_CTRL	310	0x20	R_FIFO_BL0_IRQ	318			
0x1A	R_TI_WD	311	0x21	R_FIFO_BL1_IRQ	319			
0x50	R_PLL_CTRL	312	0x22	R_FIFO_BL2_IRQ	320			
			0x23	R_FIFO_BL3_IRQ	321			
			0x50	R_PLL_STA	321			

9.1 Clock

9.1.1 Clock output

XHFC-2SU supplies a programmable clock output which can be used for source of clocking for any external device. Even a CPU or MCU supervising the whole system in which XHFC-2SU is used can be clocked. The left part of Figure 9.1 shows the block diagram of CLK_OUT generation.

After reset, the clock output frequency

$$f(\text{CLK_OUT}) = \frac{1}{8}f(\text{OSC_OUT})$$

is available at pin CLK_OUT.

Two clock sources are available for f_{SRC} . $V_CLK_F1 = '0'$ in register R_CLK_CFG selects the clock oscillator. Alternatively, pin F1_1 is used as clock source when $V_CLK_F1 = '1'$.

The clock output frequency depends on the programming bits as follows:

$$f(\text{CLK_OUT}) = \begin{cases} f_{\text{SRC}} & ; V_CLKO_HI = '1' \text{ and } V_CLKO_PLL = '0' \\ \frac{1}{8}f_{\text{SRC}} & ; V_CLKO_HI = '0' \text{ and } V_CLKO_PLL = '0' \\ f_{\text{out}} & ; V_CLKO_PLL = '1' \end{cases}$$

V_CLKO_OFF must be '0' to enable the tristate type driver of pin CLK_OUT.



Please note !

All setup bits shown in Figure 9.1 are implemented in a way that there are no glitches on the clocks when register bits change.

9.1.2 Clock distribution

XHFC-2SU uses several internal clock frequencies. They are generated as shown in the right part of Figure 9.1.

The system clock f_{SYS} is derived either from the OSC_OUT clock or from the internal PLL output clock. When the oscillator frequency is either 24.576 MHz or 49.152 MHz, V_CLK_PLL should be left in its reset state '0'. With any other oscillator frequency the internal PLL must be used to generate the required XHFC-2SU clock f_{SYS} and V_CLK_PLL must be set to '1' in register R_CLK_CFG.

Both the line interface clock f_{SU} and the PCM clock f_{PCM} must be set up to achieve 12.288 MHz and 49.152 MHz respectively. This is done with bit V_SU_CLK in register R_CTRL and bitmap V_PCM_CLK in register R_CLK_CFG.

The internal clocks f_{SYS} , f_{SU} and f_{PCM} can be switched off with $V_CLK_OFF = '1'$ in register R_CIRM. Any write access to the host processor interface or a high level at pin WAKEUP restarts the clock distribution.



Please note !

The timing specification of the processor interface is based on $t_{SYS} = 1/f_{SYS}$. This must be taken into account if a lower oscillator frequency is used.

Before the PLL is programmed and has reached its locked state, the host processor might wait between XHFC-2SU accesses.

9.1.3 Clock oscillator circuitry

There are different ways to provide the internal clocks of XHFC-2SU. This section describes the Pierce oscillator circuitry, gives a hint to 3rd overtone oscillator and the usage of crystal oscillator circuitries.

9.1.3.1 Frequency accuracy

ISDN applications need an exact clock frequency. By the ISDN specification a precision of ± 100 ppm is minimum requirement for passing the ISDN type approval. In respect to temperature dependence and ageing behavior a crystal with ± 50 ppm is recommended.

9.1.3.2 Pierce oscillator

A typical clock oscillator circuitry using a 24.576 MHz crystal is shown in Figure 9.2. This Pierce oscillator is very popular for clock generation and is widely known from literature.

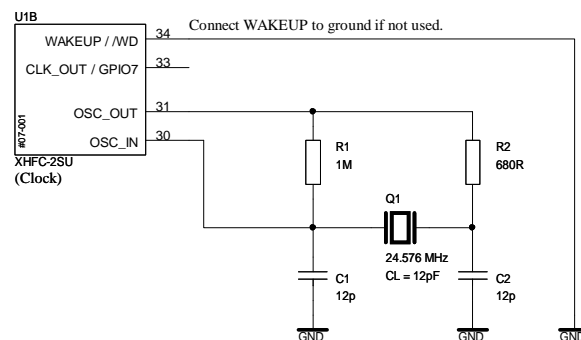


Figure 9.2: Standard XHFC-2SU quartz circuitry

The feedback resistor R1 determines the DC operation point and is typically in the range $100\text{ k}\Omega \dots 10\text{ M}\Omega$ for CMOS inverters.

The capacitive load C_L of the crystal is given in its data sheet. C1 and C2 should be chosen to fulfill

$$C_L = \frac{C_1 \cdot C_2}{C_1 + C_2} + C_s$$

where C_s is the stray capacitance. It is given by the input and output capacitances of the inverter and the shunt capacitance between the crystal terminals. Typically, C1 and C2 are chosen to be equal.

Finally, the resistor R2 is chosen to be roughly equal to the capacitive reactance of C2 at the frequency of oscillation.

$$R2 \sim \frac{1}{2\pi f_{Q1} \cdot C2}$$

The minimum value of R2 depends on the recommended power consumption of the crystal. A too small value may damage the crystal or shorten the lifetime. When R2 is too large, the oscillation might not start. As XHFC-2SU has a buffered inverter between pins OSC_IN and OSC_OUT the value of R2 can be increased. A factor of about 2..3, e.g., is well.

The circuitry shown in Figure 9.2 is based on a crystal with $C_L = 12\text{pF}$ and a stray capacitance of $C_S = 6\text{pF}$. This leads to

$$C1 = C2 = 2 \cdot (C_L - C_S) = 12\text{pF}$$

and

$$R2 = (1..3) \cdot \frac{1}{2\pi f_{Q1} \cdot C2} = (1..3) \cdot 540\Omega \sim 680\Omega .$$



Please note !

The here shown dimensioning of the oscillator circuitry is only an example and depends on the used crystal as well as on the particular board design. In general it is recommended to check oscillation build-up, power consumption of the crystal and the so-called safety factor within the particular design.

The specified drive level should not be put to the extreme to avoid early crystal ageing. Typically, within a good dimensioned circuitry, a low load capacitance C_L is a condition for a low drive level.

9.1.3.3 3rd overtone oscillator

A different oscillator frequency with double frequency 49.152 MHz can be used alternatively. For this a 3rd overtone crystal or a clock oscillator can be used.

9.1.3.4 Crystal oscillator circuitry

It is possible to feed the OSC_IN input of XHFC-2SU with a standard 3.3 V crystal oscillator. The input switching level is close to $V_{DD}/2$ (CMOS level) and XHFC-2SU can accept at least a duty cycle of 45 % high/55 % low to 55 % high/45 % low.

9.1.3.5 Several XHFC-2SU with a single oscillator circuitry

When several XHFC-2SU are used within an application, only one oscillator circuitry is required. Pin CLK_OUT can be used to distribute the clock to all XHFC-2SU as shown in Figure 9.3.

9.2 Phase locked loop (PLL)

9.2.1 Overview

Depending on the external clock source connected to pin OSC_IN, the internal PLL shown in Figure 9.1 is either required for internal clock generation or it is available for other application needs.

- When a telecommunication quartz with either 24.576 MHz or 49.152 MHz is used, the PLL is not required for f_{SYS} generation. In this case, the PLL is not occupied from XHFC-2SU and it can be used for any other application needs.
- When any other clock frequency is feed in pin OSC_IN, the PLL is required for f_{SYS} generation. f_{SU} and f_{PCM} are derived from f_{SYS} as shown in Figure 9.1.

The internal PLL is a fully digital implementation even though it is commonly seen to be an analog function. This is realized with the new DIGICC™ technology introduced by Cologne Chip ¹.

Technical features:

- Oscillator frequency range $54\text{MHz} \leq f_{\text{OSC}} \leq 108\text{MHz}$
- Programmable loop multiplication $5 \leq N \leq 255$
- Programmable predivider P and post-scaler S with range $1 \dots 256$ each
- Deterministic clock-to-clock jitter typical 120 ps
- f_{out} duty cycle 40% .. 60%
- No external loop filter or capacity needed
- Very short lock time (worst case 2000 periods of f_{ref})

9.2.2 PLL structure

The PLL consists of a predivider, the PLL circuitry and a post-scaler. This structure is shown in Figure 9.4.

The PLL loop with input frequency f_{ref} and output frequency f_{PLL} has the ratio

$$\frac{f_{\text{PLL}}}{f_{\text{ref}}} = N$$

with $N = V_PLL_N = 5 \dots 255$ (0 .. 4 are not allowed).

The overall frequency ratio

$$\frac{f_{\text{out}}}{f_{\text{in}}} = \frac{N}{P \cdot S}$$

can be adjusted with the predivider and the post-scaler. Both dividers operate in the range $1 \dots 256$.

¹Detailed information about the DIGICC™ technology and the here used PLL is documented in [1, 2]. It is *not necessary* to read these documents in order to understand this data sheet.

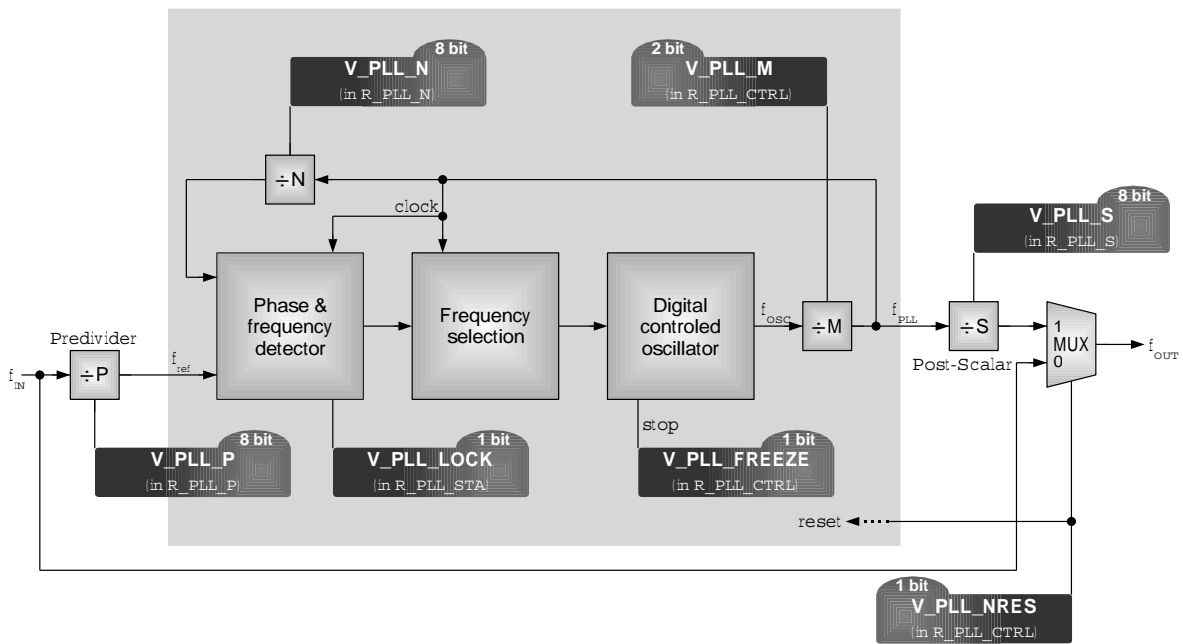


Figure 9.4: PLL block diagram

An additional divider with $M = V_PLL_M + 1 = 1..4$ has effect only inside the PLL circuitry. As the PLL output frequency f_{PLL} is used for internal PLL clock signal, the power consumption can be reduced with lower

$$f_{PLL} = \frac{f_{osc}}{M}$$

frequency.

9.2.3 PLL operation

The PLL is disabled after reset and the input frequency f_{in} is directly feed to the output, i.e. $f_{out} = f_{in}$. A '0' to '1' transition of bit V_PLL_NRES in register R_PLL_CTRL starts the PLL beginning with the lowest oscillator frequency. V_PLL_LOCK is '1' when the PLL is locked.

The PLL can be switched into standby mode without loss of current settings with $V_PLL_FREEZE = '1'$ in register R_PLL_CTRL . Then $f_{out} = 0$ until V_PLL_NRES returns to '0' again. Re-activation takes only few f_{ref} periods. V_PLL_LOCK returns '0' in standby mode.

9.2.4 Supply noise requirements for PLL locking

It is very important to consider that power supply noise of XHFC-2SU has strong influence on the PLL locking. Therefore, if the PLL's output clock needs high accuracy, power supply noise of XHFC-2SU must be very low. Typically it is recommended, that supply noise does not exceeds 10mV peak-to-peak in this case.

If the PLL output frequency is used to clock devices with low frequency accuracy requirements, power noise does not matter because the long term frequency deviation is always less than 100 ppm. This means, PLL jitter might be poor, but the PLL output frequency is accurate, of course.

Please see electrical characteristics in chapter 11 for more details.

9.2.5 PLL configuration

The PLL has four parameters P, M, N and S to be specified. Tables 9.2 and 9.3 show examples for PLL configuration settings. The parameter tuple (P, M, N, S) is given for often used frequencies. Approximate solutions have additional information about the f_{out} offset in parts per million and absolute value.

If other parameter sets are required, they can be calculated with the formulars given in section 9.2.2. Alternatively, the Cologne Chip support team will help you choosing suitable parameter sets for your application.



Please note !

The ISDN specification provides a clock precision of at least ± 100 ppm. PLL output offset must be added to the crystal precision. This must be taken into account when a PLL configuration setup is chosen from the examples in Tables 9.2 and 9.3.

Table 9.2: PLL setup examples (P, M, N, S) with ISDN related frequencies for f_{out} , (approximations have additional information about f_{out} offset)

f_{in} (MHz)	f_{out} (MHz)			
	7.68	12.288	24.576	49.152
1.8432	(1, 2, 25, 6) exact	(1, 2, 20, 3) exact	(1, 1, 40, 3) exact	(3, 2, 80, 1) exact
3.579545	(5, 1, 118, 11) -33 ppm , -249 Hz	(37, 4, 254, 2) -119 ppm , -1454 Hz	(37, 4, 254, 1) -119 ppm , -2908 Hz	(15, 2, 206, 1) +145 ppm , +7085 Hz
6	(5, 2, 32, 5) exact	(13, 1, 213, 8) +38 ppm , +462 Hz	(13, 1, 213, 4) +38 ppm , +924 Hz	(13, 1, 213, 2) +38 ppm , +1847 Hz
7.68	(1, 2, 5, 5) exact	(1, 1, 8, 5) exact	(5, 4, 16, 1) exact	(5, 2, 32, 1) exact
10.7	(41, 2, 206, 7) +19 ppm , +140 Hz	(31, 1, 178, 5) -22 ppm , -259 Hz	(16, 1, 147, 4) +23 ppm , +563 Hz	(16, 1, 147, 2) +23 ppm , +1125 Hz
12	(5, 2, 16, 5) exact	(25, 1, 128, 5) exact	(26, 1, 213, 4) +38 ppm , +924 Hz	(26, 1, 213, 2) +38 ppm , +1847 Hz
12.288	(1, 1, 5, 8) exact	(1, 1, 5, 5) exact	(1, 1, 6, 3) exact	(1, 1, 8, 2) exact
14.31818	(53, 2, 199, 7) +14 ppm , +102 Hz	(67, 4, 115, 2) -1 ppm , -10 Hz	(67, 4, 115, 1) -1 ppm , -20 Hz	(67, 2, 230, 1) -1 ppm , -39 Hz
16	(5, 2, 12, 5) exact	(25, 1, 96, 5) exact	(125, 4, 192, 1) exact	(83, 2, 255, 1) +95 ppm , +4627 Hz
24.576	(2, 1, 5, 8) exact	(2, 1, 5, 5) exact	(2, 1, 6, 3) exact	(2, 1, 8, 2) exact
25	(125, 2, 192, 5) exact	(59, 4, 58, 2) +12 ppm , +136 Hz	(59, 4, 58, 1) +12 ppm , +272 Hz	(59, 2, 116, 1) +12 ppm , +543 Hz
32.768	(8, 1, 15, 8) exact	(2, 1, 6, 8) exact	(2, 1, 6, 4) exact	(2, 1, 6, 2) exact
33	(25, 1, 64, 11) exact	(61, 1, 159, 7) +5 ppm , +57 Hz	(111, 1, 248, 3) +24 ppm , +577 Hz	(143, 2, 213, 1) +38 ppm , +1847 Hz
48	(5, 1, 8, 10) exact	(25, 1, 32, 5) exact	(125, 4, 64, 1) exact	(125, 2, 128, 1) exact
49.152	(4, 1, 5, 8) exact	(3, 1, 6, 8) exact	(3, 1, 6, 4) exact	(3, 1, 6, 2) exact
66	(25, 1, 32, 11) exact	(122, 1, 159, 7) +5 ppm , +57 Hz	(205, 1, 229, 3) -16 ppm , -391 Hz	(143, 1, 213, 2) +38 ppm , +1847 Hz

Table 9.3: PLL setup examples (P, M, N, S) with ISDN related frequencies for f_{in} , (approximations have additional information about f_{out} offset)

f_{out} (MHz)	f_{in} (MHz)			
	7.68	12.288	24.576	49.152
1.8432	(1, 2, 6, 25) exact	(1, 1, 6, 40) exact	(2, 1, 6, 40) exact	(4, 1, 6, 40) exact
3.579545	(59, 4, 110, 4) +33 ppm , +117 Hz	(10, 1, 67, 23) +1 ppm , +3 Hz	(20, 1, 67, 23) +1 ppm , +3 Hz	(40, 1, 67, 23) +1 ppm , +3 Hz
6	(2, 1, 25, 16) exact	(16, 1, 125, 16) exact	(32, 1, 125, 16) exact	(64, 1, 125, 16) exact
7.68	(1, 2, 5, 5) exact	(1, 1, 5, 8) exact	(2, 1, 5, 8) exact	(4, 1, 5, 8) exact
10.7	(89, 4, 248, 2) +22 ppm , +225 Hz	(89, 4, 155, 2) +22 ppm , +225 Hz	(89, 2, 155, 4) +22 ppm , +225 Hz	(89, 1, 155, 8) +22 ppm , +225 Hz
12	(2, 1, 25, 8) exact	(16, 1, 125, 8) exact	(32, 1, 125, 8) exact	(64, 1, 125, 8) exact
12.288	(1, 1, 8, 5) exact	(1, 1, 5, 5) exact	(2, 1, 5, 5) exact	(3, 1, 6, 8) exact
14.31818	(59, 4, 110, 1) +33 ppm , +465 Hz	(23, 1, 134, 5) +1 ppm , +12 Hz	(23, 1, 67, 5) +1 ppm , +12 Hz	(46, 1, 67, 5) +1 ppm , +12 Hz
16	(2, 1, 25, 6) exact	(16, 1, 125, 6) exact	(32, 1, 125, 6) exact	(64, 1, 125, 6) exact
24.576	(5, 4, 16, 1) exact	(1, 1, 6, 3) exact	(2, 1, 6, 3) exact	(3, 1, 6, 4) exact
25	(47, 4, 153, 1) +35 ppm , +852 Hz	(29, 4, 59, 1) -12 ppm , -276 Hz	(29, 2, 59, 2) -12 ppm , -276 Hz	(29, 1, 59, 4) -12 ppm , -276 Hz
32.768	(5, 1, 64, 3) exact	(1, 1, 8, 3) exact	(2, 1, 8, 3) exact	(3, 1, 6, 3) exact
33	(37, 3, 159, 1) +99 ppm , +3244 Hz	(89, 3, 239, 1) -58 ppm , -1888 Hz	(89, 1, 239, 2) -58 ppm , -1888 Hz	(71, 1, 143, 3) -38 ppm , -1240 Hz
48	(2, 1, 25, 2) exact	(16, 1, 125, 2) exact	(32, 1, 125, 2) exact	(64, 1, 125, 2) exact
49.152	(5, 2, 32, 1) exact	(1, 1, 8, 2) exact	(2, 1, 8, 2) exact	(3, 1, 6, 2) exact
66	(27, 1, 232, 1) -135 ppm , -8889 Hz	(35, 1, 188, 1) +63 ppm , +4115 Hz	(89, 1, 239, 1) -58 ppm , -3776 Hz	(178, 1, 239, 1) -58 ppm , -3776 Hz

9.3 Reset

XHFC-2SU has a level sensitive reset input at pin 15 with active low level. The pins MODE0 and MODE1 must already be stable during reset. The reset pulse must not be shorter than 10 ns.

After reset XHFC-2SU enters an initialization sequence. Its duration depends on the number of FIFOs and has a maximum length of 40 μ s with $f_{SYS} = 24.576$ MHz. When the initialization process is finished, bit V_BUSY in register R_STATUS changes from '1' to '0'.

PCM initialization takes 149 μ s with $f_{PCM} = 49.152$ MHz. V_PCM_INIT in register R_STATUS has the value '1' during PCM reset phase. It changes to '0' when the PCM initialization has finished.

XHFC-2SU has 4 different software resets which means that the registers are assigned to so-called reset groups.

The FIFO registers, PCM registers and ST/U_p registers belong to reset groups 1..3 and can be reset independently with the bits of register R_CIRM which are listed in Table 9.4.

A global software reset puts all registers of reset group 0 back to their default value and implies reset groups 1..3 as well. It is very similar to the hardware reset, except a few registers which have only hardware reset (see register list from page 21).

The reset bits must be set and cleared by software.

A hardware reset implies all reset groups 0..3, of course.

Table 9.4: XHFC-2SU reset groups

Reset name	Reset group	Register bit	Description
Hardware reset	H	–	Hardware reset initiated by /RES input pin. The hardware reset implies reset of all registers of reset groups 0..3 as well.
Global Software reset	0	V_SRES	The global software reset, which is similar to the hardware reset, restores the default values to the most registers. The global software reset implies reset of all registers of reset groups 1..3 as well.
HFC reset	1	V_HFC_RES	Reset for all FIFO registers of XHFC-2SU.
PCM reset	2	V_PCM_RES	Reset for all PCM registers of XHFC-2SU.
ST/U _p reset	3	V_SU_RES	Reset for all ST/U _p registers of XHFC-2SU.

Information about the allocation of the registers to the different reset groups can be found in the register list on pages 22 and 24. Some registers are allocated to more than one reset group, some have only hardware reset, and some have no default value at all.

9.4 Interrupt

9.4.1 Common features

XHFC-2SU is equipped with a maskable interrupt engine. A big variety of interrupt sources can be enabled and disabled. All interrupt events are reported on reading the interrupt status registers independently of masking the interrupts or not. Reading an interrupt status register resets the bits. Interrupt bits which are set during the read access are reported at the next read access of the interrupt status register.

Mask bits are used to enable or disable signal generation on the interrupt pin /INT. Every interrupt bit can be masked individually.

Pin 14 is the interrupt output line. After reset, all interrupts are disabled. The interrupt line must be enabled with V_GLOB_IRQ_EN set to '1' in register R_IRQ_CTRL. The polarity of the interrupt signal can be changed from *active low* to *active high* with bitmap V_IRQ_POL in the same register. Please note, that the interrupt line cannot be shared with active high polarity.

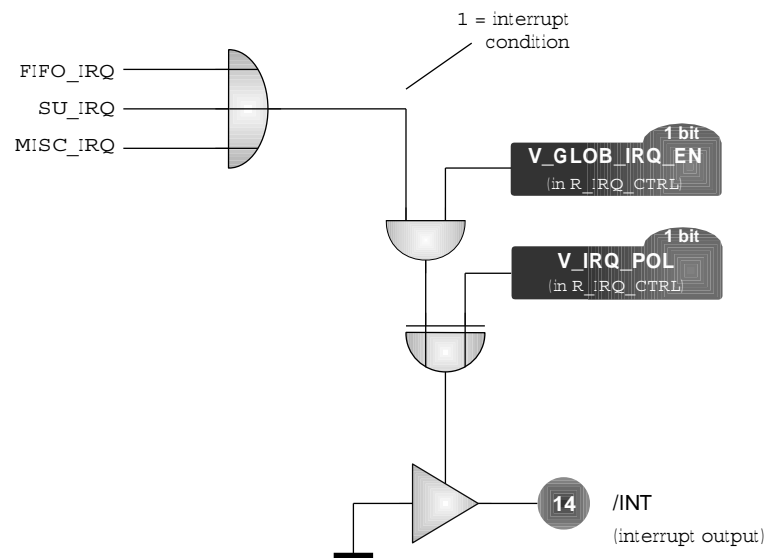


Figure 9.5: Interrupt output

9.4.2 ST/U_p interface interrupt

Every line interface can have its own interrupt capability to indicate a state change condition. The interrupt mask has to be programmed in register R_SU_IRQMSK.

When an ST/U_p interface interrupt occurred, the corresponding line interface can be determined by reading register R_SU_IRQ. This register contains the state change condition even if the interrupts are disabled.

9.4.3 FIFO interrupt

FIFO interrupts can be enabled to indicate the status for every FIFO individually.

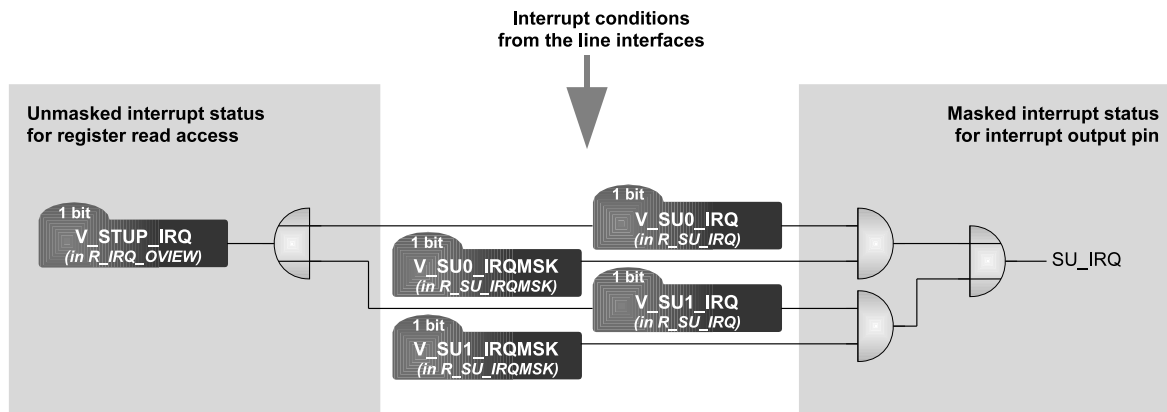


Figure 9.6: ST/Up interface interrupt

The interrupt status of all 32 FIFOs can be read from registers R_FIFO_BL0_IRQ .. R_FIFO_BL3_IRQ. All FIFOs are organized in four blocks with 8 FIFOs each. Every block has an overview bit V_FIFO_BL0_IRQ .. V_FIFO_BL3_IRQ in register R_IRQ_OVIEW.

FIFO interrupt status bits in registers R_FIFO_BL0_IRQ .. R_FIFO_BL3_IRQ are only set, when V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL as shown in Figure 9.7.

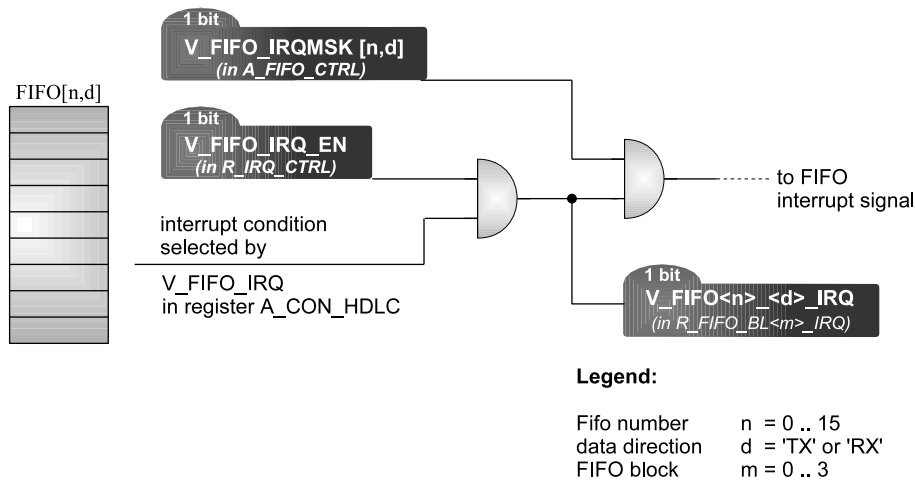


Figure 9.7: Enable FIFO interrupt condition with V_FIFO_IRQ_EN

FIFO in transparent mode (V_HDLC_TRP = '1'): An interrupt occurs due to the setting of V_FIFO_IRQ in register A_CON_HDLC.

HDLC mode without mixed interrupt mode (V_HDLC_TRP = '0' and V_MIX_IRQ = '0'): An interrupt occurs at *end of frame* condition (which leads to a frame counter increment) or after a FIFO underrun condition.

HDLC mode and mixed interrupt mode (V_HDLC_TRP = '0' and V_MIX_IRQ = '1'):

An interrupt occurs both at *end of frame* condition (which leads to a frame counter increment) or after a FIFO underrun condition and due to the setting of V_FIFO_IRQ in register A_CON_HDLC.

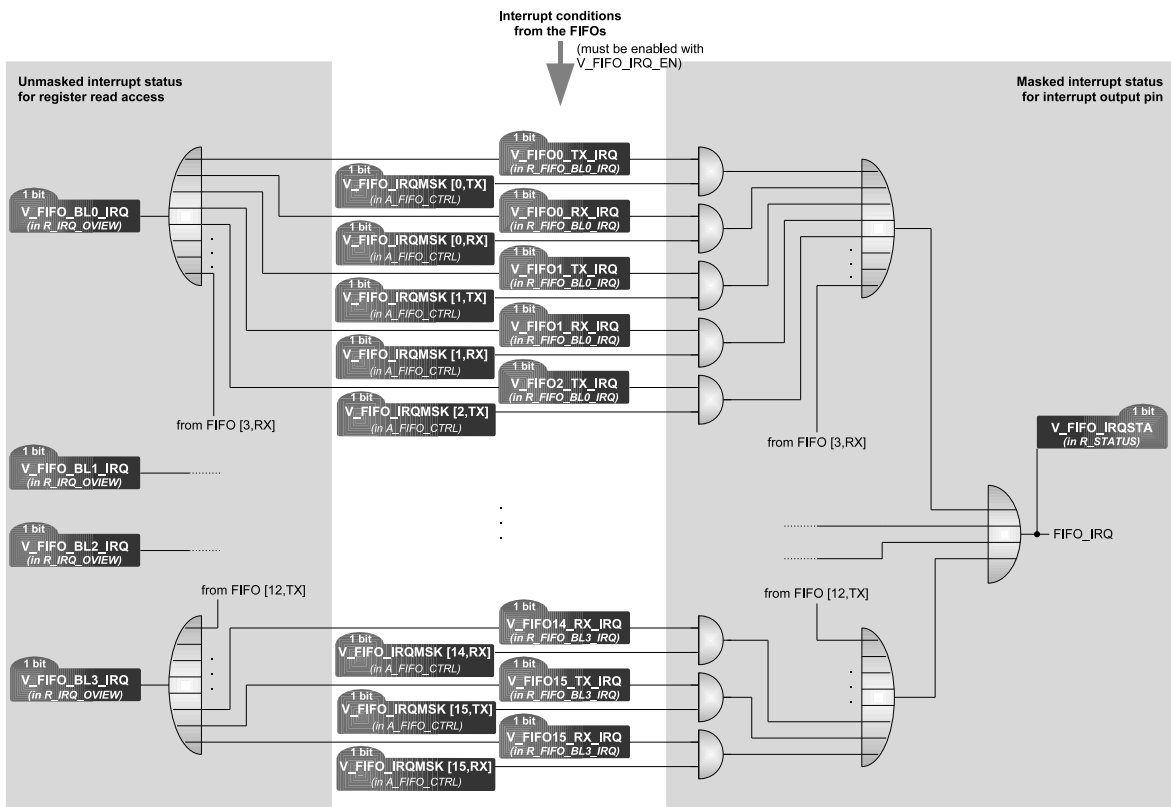


Figure 9.8: FIFO interrupt

9.4.4 Miscellaneous interrupts

Seven miscellaneous interrupts are available to report important XHFC-2SU status. Figure 9.9 shows the block diagram of these interrupt capabilities.

An overview bit V_MISC_IRQ can be read from register R_IRQ_OVIEW.

9.4.4.1 Line interface frequency slip interrupt

The frame synchronization signal can either be the F0I0 or the AF0 signal as shown in Figure 5.16 on page 188. The actual selection of a line interface can be read from bit V_SU_AF0 in register A_SU_STA or together for both line interfaces from register R_AF0_OVIEW. Any change of these selections causes an interrupt event when V_SLIP_IRQMSK in register R_MISC_IRQMSK is set to '1'.

The interrupt condition is shown in V_SLIP_IRQ of register R_MISC_IRQ even if the mask bit is not set.

It is recommended to store the R_AF0_OVIEW value by the application software to detect any frequency slips.

Data might be corrupted when a frequency slip occurs. This is explained in detail in section 5.4.2 from page 187.

An unused line interface might trigger interrupts because of the free running 8 kHz frame signal

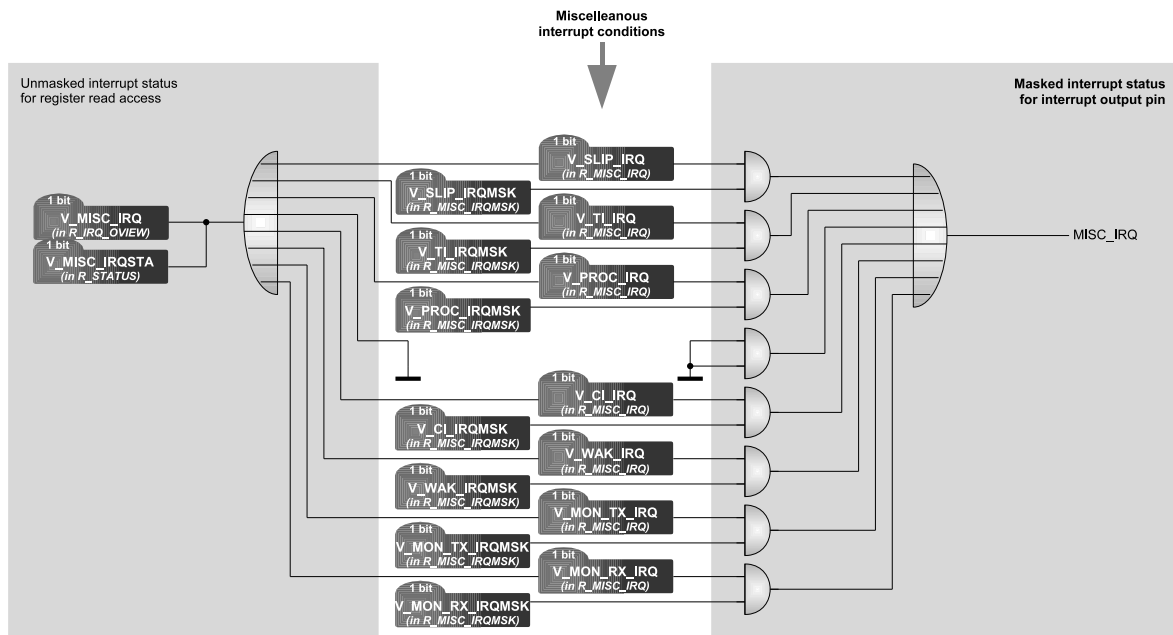


Figure 9.9: Miscellaneous interrupts

FSC_RX. These unwanted interrupt events can be avoided when the unused line interface is switched into NT mode with $V_SU_MD = '1'$ in register A_SU_CTRL0. Moreover, $V_SU_SYNC_NT$ must be in its default state '0' in register A_SU_CTRL2.

9.4.4.2 Timer interrupt

XHFC-2SU includes a timer with interrupt capability. The timer counts F0IO pulses, i.e. it is incremented every 125 μ s.

A timer event is indicated with $V_TI_IRQ = '1'$ in register R_MISC_IRQ. This event generates an interrupt if the mask bit V_TI_IRQMSK is set to '1' in register R_MISC_IRQMSK.

A timer event is generated every $2^{V_EV_TS} \cdot 250 \mu$ s where $V_EV_TS = 0..15$ in register R_TI_WD. 16 timer event frequencies are available in the range 250 μ s .. 8.192 s.

9.4.4.3 Processing / non-processing interrupt

XHFC-2SU changes every 125 μ s from non-processing into processing state. When it returns to non-processing state, this event can be reported with an interrupt.

Bit V_PROC_IRQMSK in register R_MISC_IRQMSK must be set to '1' to enable this interrupt capability. In case of an interrupt, bit V_PROC_IRQ in register R_MISC_IRQ has the value '1'.

This interrupt occurs once in every 125 μ s cycle but the distance between two consecutive interrupts changes due to the load of the internal processing machine.

9.4.4.4 Command/indication interrupt (GCI interface)

This interrupt occurs when the received indication bits of the C/I-channel have changed.

9.4.4.5 Wakeup interrupt

The wakeup pin can be enabled with $V_WAK_EN = '1'$ in register R_PWM_MD as shown in Figure 9.10. A high level is stored in V_WAK_IRQ of register R_MISC_IRQ . This bit generates an interrupt if the mask bit $V_WAK_IRQMSK = '1'$ in register R_MISC_IRQMSK .

The wakeup pin can also be used as common external interrupt input, or as general purpose input configurable with or without interrupt capability.

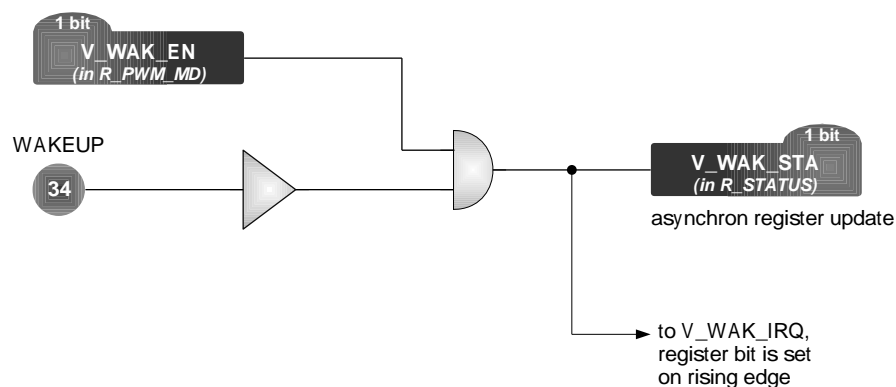


Figure 9.10: Wakeup interrupt

9.4.4.6 Interrupt for GCI monitor byte transmission

The interrupt bit $V_MON_TX_IRQ$ in register R_MISC_IRQ is set to '1' when the next monitor byte can be written into register R_MON_TX .

The interrupt status is reported in $V_MON_TX_IRQ$ of register R_MISC_IRQ even when the mask bit $V_MON_TX_IRQMSK$ in register R_MISC_IRQMSK is not set.

9.4.4.7 Interrupt after GCI monitor byte received

After a monitor byte has been received and stored in register R_MON_RX , an interrupt can be generated.

The interrupt status is reported in $V_MON_RX_IRQ$ of register R_MISC_IRQ even when the mask bit $V_MON_RX_IRQMSK$ in register R_MISC_IRQMSK is not set.

9.5 Watchdog reset

The parallel processor interface of XHFC-2SU includes a watchdog functionality.

A watchdog event generates a low signal at pin /WD. The watchdog timer can either be reset manually or automatically.

- Manual watchdog reset is selected with $V_AUTO_WD_RES = '0'$ in register $R_BERT_WD_MD$. Then, writing $V_WD_RES = '1'$ in register $R_BERT_WD_MD$ resets the watchdog timer. This bit is automatically cleared afterwards.
- $V_AUTO_WD_RES = '1'$ must be set to switch on the automatically watchdog reset. In this case every access to the chip clears the watchdog timer.

The watchdog counter is incremented every 2 ms. An event occurs after $2^{V_WD_TS} \cdot 2\text{ms}$ where $V_WD_TS = 0..15$ in register R_TI_WD . This leads to a watchdog event frequency from 2 ms to 65 536 s.

9.6 Register description

9.6.1 Write only registers

R_CIRM	(w)	(Reset group: H)	0x00
Interrupt and reset register			
All reset bits in this register must be cleared by software. It is not allowed to write any register while reset is pending.			
Bits	Reset value	Name	Description
0	0	V_CLK_OFF	Global clock enable / disable '0' = all internal clocks (f_{SYS} , f_{SU} and f_{PCM}) are enabled '1' = all internal clocks are disabled This bit is reset at every write access to XHFC-2SU or with a wake-up signal on pin WAKEUP.
1	0	V_WAIT_PROC	Additional /WAIT signal after write access The wait signal gets low with every access to a register when a preceding write access is internally not yet completed. '0' = normal /WAIT signal '1' = additional /WAIT signal This bit should be set if timing problems occur with fast processors.
2	0	V_WAIT_REG	Additional /WAIT signal during internal busy phase The wait signal gets low with every access to a register if a preceding change FIFO, increment FIFO or reset FIFO operation has activated an internal busy phase. '0' = normal /WAIT signal '1' = additional /WAIT signal during busy phase This bit can be set when busy polling (wait until not busy) is not used by the software.
3	0	V_SRES	Global software reset (reset group 0) This reset sets all registers of reset group 0 to their default value. It includes also the reset of all registers of reset groups 1..3. The selected I/O address (CIP) remains unchanged. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
4	0	V_HFC_RES	HFC reset (reset group 1) Sets all FIFO and HDLC registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
5	0	V_PCM_RES	PCM reset (reset group 2) Sets all PCM registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
6	0	V_SU_RES	Line interface reset (reset group 3) Sets all registers of both line interfaces to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
7	0	(reserved)	Must be '0'.

R_CLK_CFG	(w)	(Reset group: H)	0x02
Clock configuration register			
Bits	Reset value	Name	Description
0	0	V_CLK_PLL	Clock source selection '0' = CLK_OUT clock is derived from oscillator input OSC_IN '1' = CLK_OUT clock is derived from PLL output
1	0	V_CLKO_HI	High/low frequency selection for clock output When CLK_OUT clock is derived from oscillator input OSC_IN or F1_1 alternatively, the signal can either directly be passed to the CLK_OUT pin (high frequency) or the frequency can be divided by 8 (low frequency). '0' = low frequency, divider by 8 '1' = high frequency, no divider
2	0	V_CLKO_PLL	Source selection for clock output '0' = either OSC_IN or F1_1 signal is passed to CLK_OUT '1' = PLL output clock is passed to CLK_OUT
4..3	0	(reserved)	Must be '00'.
5	0	V_PCM_CLK	Clock of the PCM module '0' = $f_{PCM} = 2 \cdot f_{SYS}$ '1' = $f_{PCM} = f_{SYS}$ PCM clock must be set up to 49.152 MHz.
6	0	V_CLKO_OFF	Clock output enable / disable '0' = clock output pin CLK_OUT is enabled '1' = clock output pin CLK_OUT is disabled (tristate)
7	0	V_CLK_F1	Pin selection for PLL input frequency The selected pin is used for PLL input frequency as well as for CLK_OUT signal. '0' = oscillator input OSC_IN is used '1' = F1_1 input pin is used

Bits	Reset value	Name	Description
0	0	V_SLIP_IRQMSK	Interrupt mask of the line interface frequency slip
1	0	V_TI_IRQMSK	Timer interrupt mask
2	0	V_PROC_IRQMSK	Processing / non-processing transition interrupt mask (every 125 μ s)
3	0	(reserved)	Must be '0'.
4	0	V_CI_IRQMSK	Command / indication interrupt mask
5	0	V_WAK_IRQMSK	Wakeup interrupt mask
6	0	V_MON_TX_IRQMSK	Transmit monitor byte interrupt mask
7	0	V_MON_RX_IRQMSK	Receive monitor byte interrupt mask

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

Bits	Reset value	Name	Description
0	0	V_SU0_IRQMSK	Interrupt mask of line interface 0 '1' = V_SU0_IRQ in register R_SU_IRQ is used for interrupt generation
1	0	V_SU1_IRQMSK	Interrupt mask of line interface 1 '1' = V_SU1_IRQ in register R_SU_IRQ is used for interrupt generation
7..2	0	(reserved)	Must be '00 0000'.

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_IRQ_CTRL		(w)	(Reset group: H, 0)	0x13
Interrupt control register				
Bits	Reset value	Name	Description	
0	0	V_FIFO_IRQ_EN	FIFO interrupt '0' = all FIFO interrupts disabled '1' = all FIFO interrupts enabled Note: This bit enables or disables the interrupt capability of all FIFOs together. Each FIFO interrupt can be masked individually with V_FIFO_IRQMSK setting in register A_FIFO_CTRL.	
2..1	0	(reserved)	Must be '00'.	
3	0	V_GLOB_IRQ_EN	Global interrupt signal enable The interrupt line is pin 14. '0' = disable '1' = enable	
4	0	V_IRQ_POL	Polarity of interrupt signal '0' = active low signal '1' = active high signal	
7..5	0	(reserved)	Must be '000'.	

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

Bits	Reset value	Name	Description
R_TI_WD (w) (Reset group: H, 0) 0x1A			
Timer and watchdog control register			
3..0	0	V_EV_TS	Timer event after $2^n \cdot 250 \mu\text{s}$ 0 = 250 μs 1 = 500 μs 2 = 1 ms 3 = 2 ms 4 = 4 ms 5 = 8 ms 6 = 16 ms 7 = 32 ms 8 = 64 ms 9 = 128 ms 0xA = 256 ms 0xB = 512 ms 0xC = 1.024 s 0xD = 2.048 s 0xE = 4.096 s 0xF = 8.192 s
7..4	0	V_WD_TS	Watchdog event after $2^n \cdot 2 \text{ ms}$ 0 = 2 ms 1 = 4 ms 2 = 8 ms 3 = 16 ms 4 = 32 ms 5 = 64 ms 6 = 128 ms 7 = 256 ms 8 = 512 ms 9 = 1.024 s 0xA = 2.048 s 0xB = 4.096 s 0xC = 8.192 s 0xD = 16.384 s 0xE = 32.768 s 0xF = 65.536 s

R_PLL_CTRL		(w)	(Reset group: H)	0x50
PLL control register				
Bits	Reset value	Name	Description	
0	0	V_PLL_NRES	PLL reset (active low) '0' = PLL reset, PLL disabled, input clock is feed to PLL output '1' = PLL enabled, a '0' to '1' transition starts the PLL beginning with the lowest oscillator frequency	
1	0	V_PLL_TST	PLL test input '0' = test state disabled, normal operation '1' = test state enabled	
4..2	0	(reserved)	Must be '000'.	
5	0	V_PLL_FREEZE	PLL standby mode '0' = normal PLL operation '1' = PLL is in standby mode, no output clock	
7..6	0	V_PLL_M	Oscillator divider programming value The oscillator output signal is divided by $V_PLL_M + 1$.	

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

9.6.2 Read only registers

R_IRQ_OVIEW	(r)	(Reset group: H, 0, 1)	0x10
FIFO interrupt overview register			
This register gives an overview of all XHFC-2SU interrupt conditions.			
Every bit value '1' indicates that an interrupt has occurred. Interrupt conditions are only used for generating a signal on the interrupt pin 14 if the belonging mask bit is set to '1'.			
Reading this overview register does not clear any interrupt status bit.			
Bits	Reset value	Name	Description
0	0	V_FIFO_BL0_IRQ	Interrupt overview of FIFO block 0 FIFO block 0 consists of transmit FIFOs 0..3 and receive FIFOs 0..3. The exact FIFO can be determined by reading register R_FIFO_BL0_IRQ. '0' = No FIFO interrupt occurred '1' = At least one FIFO interrupt is pending.
1	0	V_FIFO_BL1_IRQ	Interrupt overview of FIFO block 1 FIFO block 1 consists of transmit FIFOs 4..7 and receive FIFOs 4..7. The exact FIFO can be determined by reading register R_FIFO_BL1_IRQ. '0' = No FIFO interrupt occurred '1' = At least one FIFO interrupt is pending.
2	0	V_FIFO_BL2_IRQ	Interrupt overview of FIFO block 2 FIFO block 2 consists of transmit FIFOs 8..11 and receive FIFOs 8..11. The exact FIFO can be determined by reading register R_FIFO_BL2_IRQ. '0' = No FIFO interrupt occurred '1' = At least one FIFO interrupt is pending.
3	0	V_FIFO_BL3_IRQ	Interrupt overview of FIFO block 3 FIFO block 3 consists of transmit FIFOs 12..15 and receive FIFOs 12..15. The exact FIFO can be determined by reading register R_FIFO_BL3_IRQ. '0' = No FIFO interrupt occurred '1' = At least one FIFO interrupt is pending.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
4	0	V_MISC_IRQ	<p>Miscellaneous interrupt overview All miscellaneous interrupts of register R_MISC_IRQ are 'ored'. '0' = No miscellaneous interrupt occurred '1' = At least one miscellaneous interrupt is pending. This bit has always the same value as V_MISC_IRQSTA in register R_STATUS. Reading register R_MISC_IRQ clears the miscellaneous interrupt bits.</p>
5	0	V_STUP_IRQ	<p>Line interface interrupt overview '0' = No line interface interrupt occurred '1' = At least one line interface interrupt is pending. Reading register R_SU_IRQ clears the line interface interrupt bits.</p>
7..6		(reserved)	

Bits	Reset value	Name	Description
Miscellaneous interrupt status register			
This register reports miscellaneous interrupt events. Reading this register clears the bits. These bits are not masked by register R_MISC_IRQMSK, i.e. they show interrupt conditions even if the interrupt is disabled.			
0	0	V_SLIP_IRQ	Interrupt status of the line interface frequency slip This interrupt occurs when the frame synchronization pulse of any line interface switches from FOIO to AF0 or reverse. The current FSC signal selection can be read from V_SU_AF0 in register A_SU_STA for the selected line interface or it can be read from the overview register R_AF0_OVIEW for both line interfaces. A change can happen from time to time if two TEs operate on slightly different frame synchronization clocks. Data might be destroyed in these cases.
1	0	V_TI_IRQ	Timer interrupt status '1' = timer elapsed
2	0	V_PROC_IRQ	Processing / non-processing transition interrupt status '1' = XHFC-2SU has changed from processing to non-processing phase (every 125 µs). Note: The current processing / non-processing status can be read from V_PROC in register R_STATUS.
3		(reserved)	
4	0	V_CI_IRQ	Command / indication interrupt status '1' = received indication bits changed
5	0	V_WAK_IRQ	Wakeup interrupt status '1' = a wakeup signal at pin WAKEUP occurred
6	0	V_MON_TX_IRQ	Transmit monitor byte interrupt status '1' = the next monitor byte can be written
7	0	V_MON_RX_IRQ	Receive monitor byte interrupt status '1' = monitor byte received

R_SU_IRQ	(r)	(Reset group: H, 0)	0x12
<p>State change interrupt status register of the line interfaces</p> <p>This register reports state changes of the line interfaces. Reading this register clears the bits. These bits are not masked by register R_SU_IRQMSK, i.e. they show state change conditions even if the interrupt is disabled.</p>			
Bits	Reset value	Name	Description
0	0	V_SU0_IRQ	Interrupt status of line interface 0 '1' = a state change occurred in line interface 0
1	0	V_SU1_IRQ	Interrupt status of line interface 1 '1' = a state change occurred in line interface 1
7..2		(reserved)	

R_STATUS	(r)	(Reset group: H, 0, 3)	0x1C
XHFC-2SU status register			
Bits	Reset value	Name	Description
0		V_BUSY	BUSY / NOBUSY status '0' = XHFC-2SU is not busy, all accesses are allowed '1' = XHFC-2SU is BUSY after initialising FIFO reset, increment <i>F</i> -counter or change FIFO
1		V_PROC	Processing / non-processing status '0' = XHFC-2SU has finished the processing phase during the 125 μ s cycle '1' = XHFC-2SU is in processing phase (once every 125 μ s cycle) Note: The processing / non-processing transition can be notified with an interrupt (see V_PROC_IRQ in register R_MISC_IRQ).
2		(reserved)	
3	0	V_LOST_STA	LOST error (frames have been lost) This means that XHFC-2SU did not process all data in 125 μ s. So data may be corrupted. Bit V_RES_LOST of register A_INC_RES_FIFO must be set to reset this bit.
4		V_PCM_INIT	PCM module initialization '0' = initialization sequence is finished '1' = initialization sequence is in progress (after hardware reset, global software reset or PCM reset) Note: The PCM clocks F0IO and C4IO are ignored during initialization.
5		V_WAK_STA	Wakeup status This bit contains the current value of pin WAKEUP when V_WAK_EN = '1' in register R_PWM_MD.
6	0	V_MISC_IRQSTA	Miscellaneous interrupt overview All miscellaneous interrupts of register R_MISC_IRQ are 'ored'. '0' = No miscellaneous interrupt occurred '1' = At least one miscellaneous interrupt is pending This bit has always the same value as V_MISC_IRQ in register R_IRQ_OVIEW. Reading register R_MISC_IRQ clears the miscellaneous interrupt bits.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7		V_FIFO_IRQSTA	Any FIFO interrupt All enabled FIFO interrupts in registers R_FIFO_BL0_IRQ..R_FIFO_BL3_IRQ are 'ored'. '0' = No FIFO interrupt is pending '1' = At least one FIFO interrupt is pending.

R_FIFO_BL0_IRQ	(r)	(Reset group: H, 0, 1)	0x20
Interrupt status register for FIFO block 0			
This register reports the interrupt status of FIFO block 0. A bit is set to '1' when an interrupt event occurs.			
Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.			
The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).			
Bits	Reset value	Name	Description
0	0	V_FIFO0_TX_IRQ	FIFO[0,TX] interrupt status
1	0	V_FIFO0_RX_IRQ	FIFO[0,RX] interrupt status
2	0	V_FIFO1_TX_IRQ	FIFO[1,TX] interrupt status
3	0	V_FIFO1_RX_IRQ	FIFO[1,RX] interrupt status
4	0	V_FIFO2_TX_IRQ	FIFO[2,TX] interrupt status
5	0	V_FIFO2_RX_IRQ	FIFO[2,RX] interrupt status
6	0	V_FIFO3_TX_IRQ	FIFO[3,TX] interrupt status
7	0	V_FIFO3_RX_IRQ	FIFO[3,RX] interrupt status

Bits	Reset value	Name	Description
0	0	V_FIFO4_TX_IRQ	FIFO[4, TX] interrupt status
1	0	V_FIFO4_RX_IRQ	FIFO[4, RX] interrupt status
2	0	V_FIFO5_TX_IRQ	FIFO[5, TX] interrupt status
3	0	V_FIFO5_RX_IRQ	FIFO[5, RX] interrupt status
4	0	V_FIFO6_TX_IRQ	FIFO[6, TX] interrupt status
5	0	V_FIFO6_RX_IRQ	FIFO[6, RX] interrupt status
6	0	V_FIFO7_TX_IRQ	FIFO[7, TX] interrupt status
7	0	V_FIFO7_RX_IRQ	FIFO[7, RX] interrupt status

R_FIFO_BL2_IRQ	(r)	(Reset group: H, 0, 1)	0x22
Interrupt status register for FIFO block 2			
<p>This register reports the interrupt status of FIFO block 2. A bit is set to '1' when an interrupt event occurs.</p> <p>Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.</p> <p>The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).</p>			
Bits	Reset value	Name	Description
0	0	V_FIFO8_TX_IRQ	FIFO[8, TX] interrupt status
1	0	V_FIFO8_RX_IRQ	FIFO[8, RX] interrupt status
2	0	V_FIFO9_TX_IRQ	FIFO[9, TX] interrupt status
3	0	V_FIFO9_RX_IRQ	FIFO[9, RX] interrupt status
4	0	V_FIFO10_TX_IRQ	FIFO[10, TX] interrupt status
5	0	V_FIFO10_RX_IRQ	FIFO[10, RX] interrupt status
6	0	V_FIFO11_TX_IRQ	FIFO[11, TX] interrupt status
7	0	V_FIFO11_RX_IRQ	FIFO[11, RX] interrupt status

R_FIFO_BL3_IRQ	(r)	(Reset group: H, 0, 1)	0x23
Interrupt status register for FIFO block 3			
<p>This register reports the interrupt status of FIFO block 3. A bit is set to '1' when an interrupt event occurs.</p> <p>Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.</p> <p>The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).</p>			
Bits	Reset value	Name	Description
0	0	V_FIFO12_TX_IRQ	FIFO[12,TX] interrupt status
1	0	V_FIFO12_RX_IRQ	FIFO[12,RX] interrupt status
2	0	V_FIFO13_TX_IRQ	FIFO[13,TX] interrupt status
3	0	V_FIFO13_RX_IRQ	FIFO[13,RX] interrupt status
4	0	V_FIFO14_TX_IRQ	FIFO[14,TX] interrupt status
5	0	V_FIFO14_RX_IRQ	FIFO[14,RX] interrupt status
6	0	V_FIFO15_TX_IRQ	FIFO[15,TX] interrupt status
7	0	V_FIFO15_RX_IRQ	FIFO[15,RX] interrupt status

R_PLL_STA	(r)	(Reset group: H)	0x50
PLL status register			
Bits	Reset value	Name	Description
6..0		(reserved)	
7	0	V_PLL_LOCK	PLL lock status '0' = PLL is unlocked or in standby mode '1' = PLL is locked

9.6.3 Read/write registers

R_PLL_P	(r/w)	(Reset group: H)	0x51
PLL predivider programming value			
Bits	Reset value	Name	Description
7..0	0	V_PLL_P	Predivider programming value The divisor of the predivider is V_PLL_P. The bitmap value 0 has the meaning of 256.

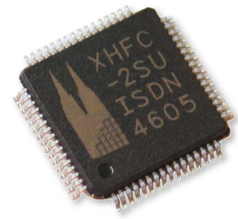
(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PLL_N	(r/w)	(Reset group: H)	0x52
PLL loop factor			
Bits	Reset value	Name	Description
7..0	0	V_PLL_N	Loop factor programming value The loop factor is V_PLL_N. Note: 0..4 are not allowed.

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PLL_S	(r/w)	(Reset group: H)	0x53
PLL post-scaler programming value			
Bits	Reset value	Name	Description
7..0	0	V_PLL_S	Post-scaler programming value The divisor of the post-scaler is V_PLL_S. The bitmap value 0 has the meaning of 256.

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)



Chapter 10

General purpose I/O pins (GPIO)

Table 10.1: Overview of the XHFC-2SU general purpose I/O registers

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x40	R_GPIO_OUT1	328	0x40	R_GPIO_IN1	335
0x42	R_GPIO_EN1	329	0x45	R_GPIO_IN2	335
0x44	R_GPIO_SEL_BL	329	0x48	R_GPIO_IN0	336
0x45	R_GPIO_OUT2	330			
0x47	R_GPIO_EN2	330			
0x48	R_GPIO_OUT0	331			
0x4A	R_GPIO_EN0	332			
0x4C	R_GPIO_SEL	333			

10.1 GPIO functionality

XHFC-2SU has up to 20 general purpose I/O (GPIO) pins. 12 pins are shared with the line interfaces and every unused ST/U_p interface makes six additional GPIO pins available. Eight further pins are shared with PCM, PWM and clock functions and are individually selectable as GPIO.

Every pin listed in Table 10.2 has three functions. A detailed GPIO block diagram is shown in Figure 10.1. For GPIO8, e.g., the following configurations are available:

- **GPIO selection bit V_GPIO_SEL0 = '0' in register R_GPIO_SEL:**

The line interface function (1st pin function) is selected for interface #0. GPIO8 output is disabled within a group of six pins (2nd pin functions GPIO8..GPIO11, GPIO16 and GPIO17 are not available).

- **GPIO selection bit V_GPIO_SEL0 = '1' in register R_GPIO_SEL and GPIO enable bit V_GPIO_EN8 = '0' in register R_GPIO_EN1:**

The line interface function (1st pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2nd pin function). GPIO8 output is disabled, i.e. the output driver is tristated.

- **GPIO selection bit V_GPIO_SEL0 = '1' in register R_GPIO_SEL and GPIO enable bit V_GPIO_EN8 = '1' in register R_GPIO_EN1:**

The line interface function (1st pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2nd pin function). GPIO8 output is enabled.

The GPIO input functionality is always enabled, i.e. registers R_GPIO_IN0 and R_GPIO_IN1 can always be read.

GPIO 0..7 are implemented in the same way with the exception that the selection bits in register R_GPIO_SEL switch only one pin from first to second function.

Unused GPIO pins should be configured as output ports. This sets the level of the input buffer – which is always active at the same pin – to a stable level and avoids floating input effect.

10.2 GPIO output voltage

As the output drivers of the ST/U_p interfaces are supplied from an external source, the GPIO output voltage of these pins is influenced by this external source as well.

Yet it is recommended to connect the VDD_SU0 and VDD_SU1 pins of unused ST/U_p interfaces to VDD.

For a few applications a GPIO output voltage different from VDD might be useful. In this case the external voltage must not exceed 3.6 V. Table 10.2 shows the allocation of power supply pins to the GPIO output drivers.

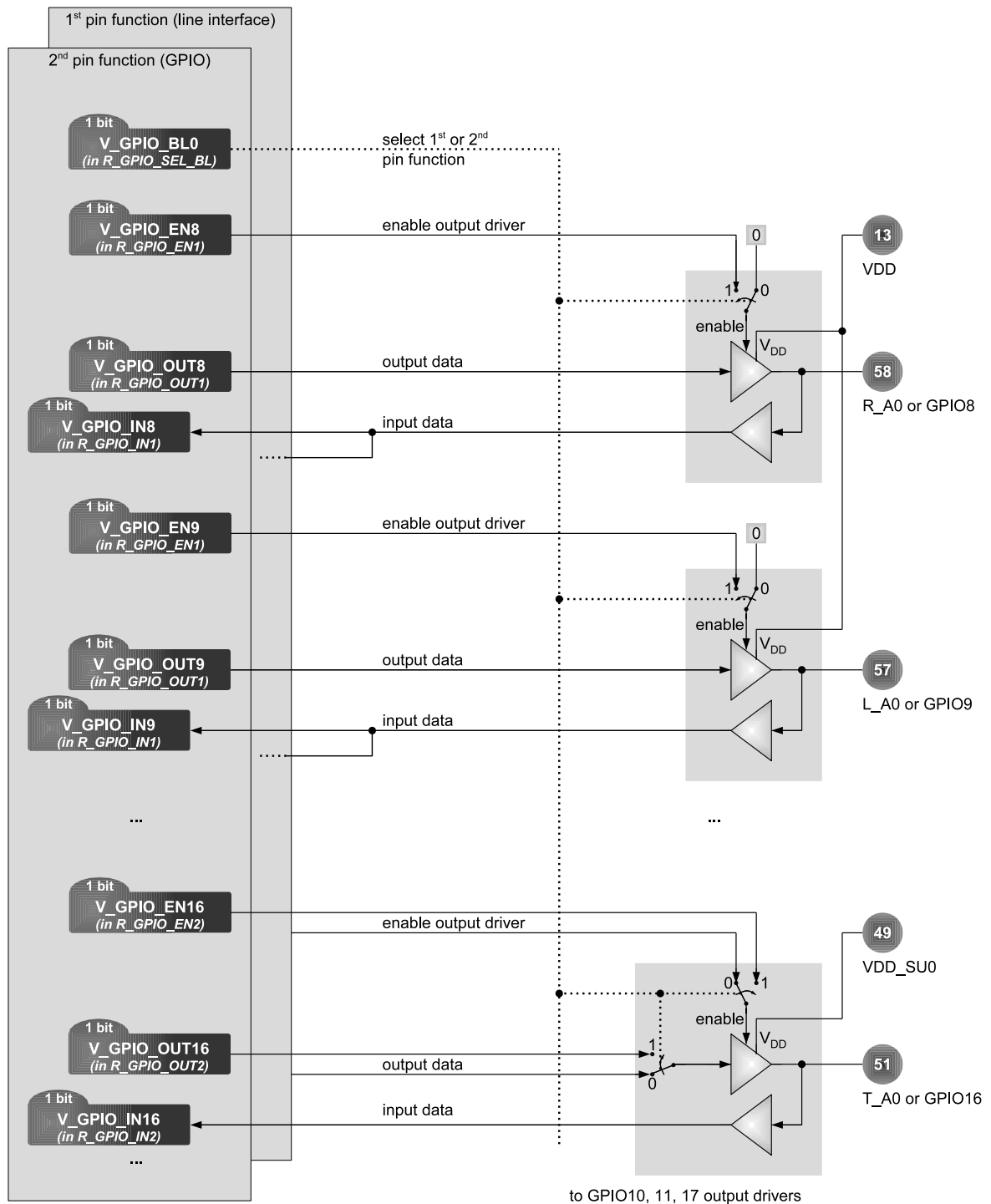


Figure 10.1: GPIO block diagram (GPIO8, GPIO9 and GPIO16 exemplarily)

Table 10.2: GPIO pins

Pin number	GPIO pin	GPIO byte	Shared with (1 st function)	Output driver power supply
36	GPIO0	0	PWM0	VDD
35	GPIO1	0	PWM1	VDD
28	GPIO2	0	F1_0	VDD
27	GPIO3	0	F1_1	VDD
21	GPIO4	0	STIO1	VDD
22	GPIO5	0	STIO2	VDD
16	GPIO6	0	SYNC_O	VDD
33	GPIO7	0	CLK_OUT	VDD
58	GPIO8	1	R_A0	VDD
57	GPIO9	1	L_A0	VDD
56	GPIO10	1	L_B0	VDD
55	GPIO11	1	R_B0	VDD
42	GPIO12	1	R_B1	VDD
41	GPIO13	1	L_B1	VDD
40	GPIO14	1	L_A1	VDD
39	GPIO15	1	R_A1	VDD
51	GPIO16	2	T_A0	VDD_SU0
50	GPIO17	2	T_B0	VDD_SU0
47	GPIO18	2	T_B1	VDD_SU1
46	GPIO19	2	T_A1	VDD_SU1

10.3 Activation state F7/G3 signalling

Some GPIO outputs can alternatively be used to report the activation states F7 or G3. This function is typically used to drive LEDs directly from XHFC-2SU pins. Figure 10.2 shows exemplarily for GPIO0, how the activation state is linked up to the GPIO signal path.

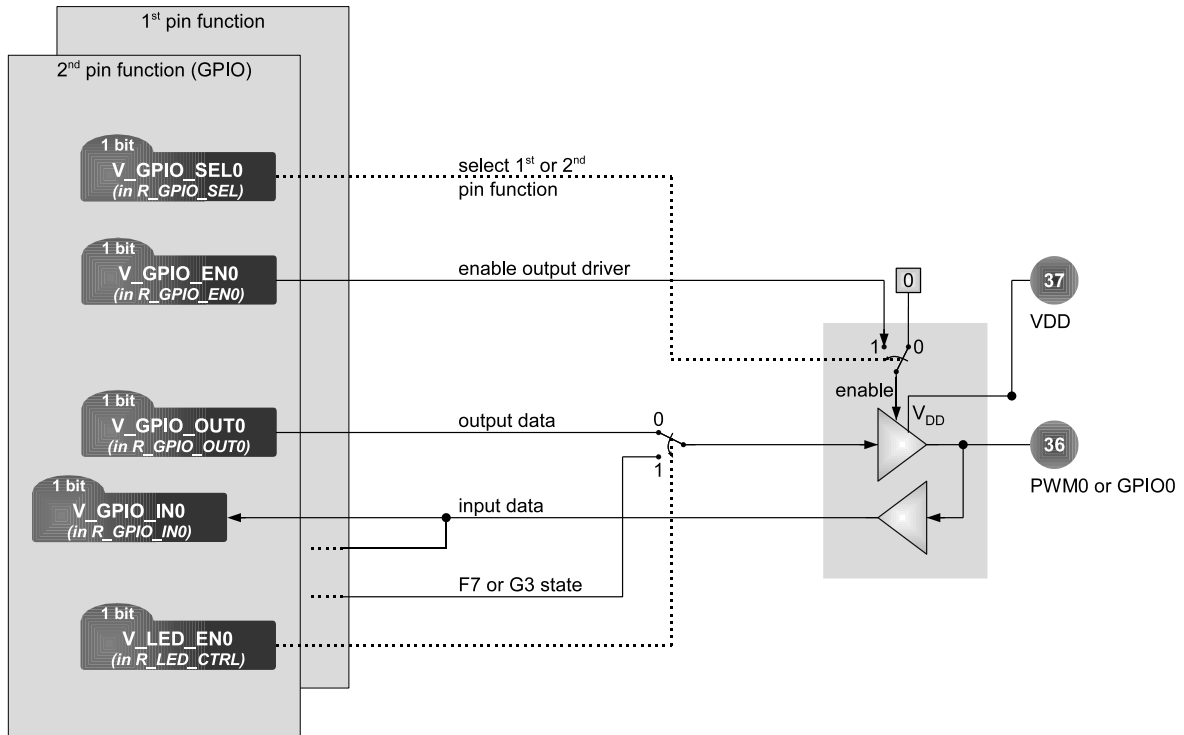


Figure 10.2: Activation state signalling exemplarily shown with GPIO0 (similar for GPIO1 .. GPIO5)

Table 10.3: State F7/G3 reporting

Line interface	V_LED_ROT in register R_SU_LED_CTRL			
	'00'	'01'	'10'	'11'
0	GPIO0 and GPIO4	GPIO1 and GPIO5	GPIO2	GPIO3
1	GPIO1 and GPIO5	GPIO2	GPIO3	GPIO0 and GPIO4

The line activation state of each line interface number 0 or 1 can be assigned to pins GPIO0 .. GPIO5 by a signal rotator / shifter.

$$\text{GPIO port number} = (\text{line interface number} + \text{V_LED_ROT}) \bmod 4$$

Additionally, GPIO4 can be used to report the activation state in parallel to GPIO0 and GPIO5 can be used to report the activation state in parallel to GPIO1.

Table 10.3 shows the assignment of GPIO port numbers 0 .. 5 and line interface numbers 0 .. 1.

10.4 Register description

10.4.1 Write only registers

R_GPIO_OUT1		(w)	(Reset group: H, 0)	0x40
GPIO output data bits 15..8				
Bits	Reset value	Name	Description	
0	0	V_GPIO_OUT8	Output data bit for pin GPIO8	
1	0	V_GPIO_OUT9	Output data bit for pin GPIO9	
2	0	V_GPIO_OUT10	Output data bit for pin GPIO10	
3	0	V_GPIO_OUT11	Output data bit for pin GPIO11	
4	0	V_GPIO_OUT12	Output data bit for pin GPIO12	
5	0	V_GPIO_OUT13	Output data bit for pin GPIO13	
6	0	V_GPIO_OUT14	Output data bit for pin GPIO14	
7	0	V_GPIO_OUT15	Output data bit for pin GPIO15	

R_GPIO_EN1	(w)	(Reset group: H, 0)	0x42
GPIO output enable bits 15..8			
Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.			
Bits	Reset value	Name	Description
0	0	V_GPIO_EN8	Output enable for pin GPIO8
1	0	V_GPIO_EN9	Output enable for pin GPIO9
2	0	V_GPIO_EN10	Output enable for pin GPIO10
3	0	V_GPIO_EN11	Output enable for pin GPIO11
4	0	V_GPIO_EN12	Output enable for pin GPIO12
5	0	V_GPIO_EN13	Output enable for pin GPIO13
6	0	V_GPIO_EN14	Output enable for pin GPIO14
7	0	V_GPIO_EN15	Output enable for pin GPIO15

R_GPIO_SEL_BL	(w)	(Reset group: H, 0)	0x44
Selection register for GPIO block			
12 GPIO pins are grouped in blocks with 6 GPIO functions each. These blocks use the line interfaces as first pin function alternatively.			
This register controls only the output driver, whereas the input functionality needs no programming.			
Bits	Reset value	Name	Description
0	0	V_GPIO_BL0	GPIO function on line interface no. 0 '0' = line interface no. 0 used (first pin function) '1' = GPIO8..GPIO11, GPIO16 and GPIO17 used (second pin function)
1	0	V_GPIO_BL1	GPIO function on line interface no. 1 '0' = line interface no. 1 used (first pin function) '1' = GPIO12..GPIO15, GPIO18 and GPIO19 used (second pin function)
7..2	0	(reserved)	Must be 0.

R_GPIO_OUT2	(w)	(Reset group: H, 0)	0x45
GPIO output data bits 19..16			
Bits	Reset value	Name	Description
0	0	V_GPIO_OUT16	Output data bit for pin GPIO16
1	0	V_GPIO_OUT17	Output data bit for pin GPIO17
2	0	V_GPIO_OUT18	Output data bit for pin GPIO18
3	0	V_GPIO_OUT19	Output data bit for pin GPIO19
7..4	0	(reserved)	Must be '0000'.

R_GPIO_EN2	(w)	(Reset group: H, 0)	0x47
GPIO output enable bits 19..16			
Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.			
Bits	Reset value	Name	Description
0	0	V_GPIO_EN16	Output enable for pin GPIO16
1	0	V_GPIO_EN17	Output enable for pin GPIO17
2	0	V_GPIO_EN18	Output enable for pin GPIO18
3	0	V_GPIO_EN19	Output enable for pin GPIO19
7..4	0	(reserved)	Must be '0000'.

R_GPIO_OUT0		(w)	(Reset group: H, 0)	0x48
GPIO output data bits 7..0				
Bits	Reset value	Name	Description	
0	0	V_GPIO_OUT0	Output data bit for pin GPIO0	
1	0	V_GPIO_OUT1	Output data bit for pin GPIO1	
2	0	V_GPIO_OUT2	Output data bit for pin GPIO2	
3	0	V_GPIO_OUT3	Output data bit for pin GPIO3	
4	0	V_GPIO_OUT4	Output data bit for pin GPIO4	
5	0	V_GPIO_OUT5	Output data bit for pin GPIO5	
6	0	V_GPIO_OUT6	Output data bit for pin GPIO6	
7	0	V_GPIO_OUT7	Output data bit for pin GPIO7	

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_GPIO_EN0	(w)	(Reset group: H, 0)	0x4A
GPIO output enable bits 7..0			
Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.			
Bits	Reset value	Name	Description
0	0	V_GPIO_EN0	Output enable for pin GPIO0
1	0	V_GPIO_EN1	Output enable for pin GPIO1
2	0	V_GPIO_EN2	Output enable for pin GPIO2
3	0	V_GPIO_EN3	Output enable for pin GPIO3
4	0	V_GPIO_EN4	Output enable for pin GPIO4
5	0	V_GPIO_EN5	Output enable for pin GPIO5
6	0	V_GPIO_EN6	Output enable for pin GPIO6
7	0	V_GPIO_EN7	Output enable for pin GPIO7

R_GPIO_SEL	(w)	(Reset group: H, 0)	0x4C
GPIO selection register			
This register controls only the output driver, whereas the input functionality needs no programming.			
Bits	Reset value	Name	Description
0	0	V_GPIO_SEL0	Selection of first or second pin function '0' = PWM0 (first pin function) '1' = GPIO0 (second pin function)
1	0	V_GPIO_SEL1	Selection of first or second pin function '0' = PWM1 (first pin function) '1' = GPIO1 (second pin function)
2	0	V_GPIO_SEL2	Selection of first or second pin function '0' = F1_0 (first pin function) '1' = GPIO2 (second pin function)
3	0	V_GPIO_SEL3	Selection of first or second pin function '0' = F1_1 (first pin function) '1' = GPIO3 (second pin function)
4	0	V_GPIO_SEL4	Selection of first or second pin function '0' = STIO1 (first pin function) '1' = GPIO4 (second pin function)
5	0	V_GPIO_SEL5	Selection of first or second pin function '0' = STIO2 (first pin function) '1' = GPIO5 (second pin function)
6	0	V_GPIO_SEL6	Selection of first or second pin function '0' = SYNC_O (first pin function) '1' = GPIO6 (second pin function)
7	0	V_GPIO_SEL7	Selection of first or second pin function '0' = CLK_OUT (first pin function) '1' = GPIO7 (second pin function)

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_SU_LED_CTRL	(w)	(Reset group: H, 0)	0x4D
LED control register			
This register can be used to indicate activated states F7 or G3 on two pins of GPIO0 .. GPIO5. Activated state is indicated by high level. LEDs with series resistor can directly be connected to XHFC-2SU.			
Bits	Reset value	Name	Description
0	0	V_LED_EN0	Enable LED output signal on pin GPIO0 '0' = normal GPIO function on pin GPIO0 '1' = activated state is indicated on pin GPIO0
1	0	V_LED_EN1	Enable LED output signal on pin GPIO1 '0' = normal GPIO function on pin GPIO1 '1' = activated state is indicated on pin GPIO1
2	0	V_LED_EN2	Enable LED output signal on pin GPIO2 '0' = normal GPIO function on pin GPIO2 '1' = activated state is indicated on pin GPIO2
3	0	V_LED_EN3	Enable LED output signal on pin GPIO3 '0' = normal GPIO function on pin GPIO3 '1' = activated state is indicated on pin GPIO3
4	0	V_LED_EN4	Enable LED output signal on pin GPIO4 '0' = normal GPIO function on pin GPIO4 '1' = activated state is indicated on pin GPIO4
5	0	V_LED_EN5	Enable LED output signal on pin GPIO5 '0' = normal GPIO function on pin GPIO5 '1' = activated state is indicated on pin GPIO5
7..6	0	V_LED_ROT	LED output rotator/shifter Activated state of the line interfaces can be assigned to pins GPIO0 .. GPIO5 via a signal rotator/shifter. When I_n is the line interface with number n : '00' = I_0 is assigned to GPIO0 and GPIO4, I_1 is assigned to GPIO1 and GPIO5 '01' = I_0 is assigned to GPIO1 and GPIO5, I_1 is assigned to GPIO2 '10' = I_0 is assigned to GPIO2, I_1 is assigned to GPIO3 '11' = I_0 is assigned to GPIO3, I_1 is assigned to GPIO0 and GPIO4

(See Section 13 on page 343 for a fault description and workaround of an address decoding problem which concerns this register among others.)

10.4.2 Read only registers

R_GPIO_IN1	(r)	(Reset group: -)	0x40
GPIO input data bits 15..8			
Note: Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0		V_GPIO_IN8	Input data bit from pin GPIO8
1		V_GPIO_IN9	Input data bit from pin GPIO9
2		V_GPIO_IN10	Input data bit from pin GPIO10
3		V_GPIO_IN11	Input data bit from pin GPIO11
4		V_GPIO_IN12	Input data bit from pin GPIO12
5		V_GPIO_IN13	Input data bit from pin GPIO13
6		V_GPIO_IN14	Input data bit from pin GPIO14
7		V_GPIO_IN15	Input data bit from pin GPIO15

R_GPIO_IN2	(r)	(Reset group: -)	0x45
GPI input data bits 19..16			
Note: Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0		V_GPIO_IN16	Input data bit from pin GPIO16
1		V_GPIO_IN17	Input data bit from pin GPIO17
2		V_GPIO_IN18	Input data bit from pin GPIO18
3		V_GPIO_IN19	Input data bit from pin GPIO19
7..4		(reserved)	

R_GPIO_IN0	(r)	(Reset group: –)	0x48
GPIO input data bits 7..0			
Note: Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0		V_GPIO_IN0	Input data bit from pin GPIO0
1		V_GPIO_IN1	Input data bit from pin GPIO1
2		V_GPIO_IN2	Input data bit from pin GPIO2
3		V_GPIO_IN3	Input data bit from pin GPIO3
4		V_GPIO_IN4	Input data bit from pin GPIO4
5		V_GPIO_IN5	Input data bit from pin GPIO5
6		V_GPIO_IN6	Input data bit from pin GPIO6
7		V_GPIO_IN7	Input data bit from pin GPIO7



Chapter 11

Electrical characteristics

Absolute maximum ratings ^{*1}

Parameter	Symbol	Min.	Max.	Conditions
Power supply	V_{DD}	-0.3 V	+4.6 V	
Input voltage	V_I	-0.3 V	$V_{DD} + 0.3 \text{ V}$ ($\leq 4.6 \text{ V}$)	3.3 V pins
	V_I	-0.3 V	6.0 V	5 V tolerant pins
Output voltage	V_O	-0.3 V	$V_{DD} + 0.3 \text{ V}$	
Storage temperature	T_{stg}	-55 °C	+125 °C	

Recommended operating conditions

Parameter	Symbol	Min.	Typ.	Max	Conditions
Power supply	V_{DD}	3.0 V	3.3 V	3.6 V	
Operating temperature	T_{opr}	-30 °C		+85 °C	
Peak-to-peak power supply noise	$V_{noise, pp}$			50 mV	PLL not used or no locking requirement
	$V_{noise, pp}$			10 mV	PLL used, stable locking required

Electrical characteristics for 3.3 V power supply

Parameter	Symbol	Min.	Typ.	Max	Conditions
Low input voltage ^{*2}	V_{IL}	-0.3 V		0.8 V	
High input voltage ^{*3}	V_{IH}	2.0 V		$V_{DD} + 0.3 \text{ V}$	
High input voltage ^{*4}	V_{IH}	2.0 V		5.5 V	
Low output voltage ^{*2}	V_{OL}			0.4 V	
High output voltage ^{*2}	V_{OH}	2.4 V			
Internal pull-up resistor to V_{DD}	$R_{pu,int}$	55 k Ω	110 k Ω	330 k Ω	

(pin type IOpu # only, see pin list from page 36 and legend on page 40)

The pin capacitance depends on the pin type. For details see legend of the pin list on page 40.

^{*1}: Stresses beyond those listed under 'Absolute maximum ratings' may cause permanent damage to the device. These are stress ratings only, and operation of the device at these or at any other conditions above those given in this data sheet is not implied. Exposure to limiting values for extended periods may affect device reliability.

^{*2}: All pins except oscillator and S/T/ U_p type.

^{*3}: All pins except oscillator, S/T/ U_p type and 5 V tolerant pins.

^{*4}: Only 5 V tolerant pins.

Power consumption for 3.3 V power supply

Parameter	Symbol	Min.	Typ.	Max	Conditions
Oscillator running with 24.576 MHz, internal clock disabled	I_{opr}		4 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running	I_{opr}		19 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running, both S/T interfaces sending 96 kHz test signal on 50 Ω load	I_{opr}		39 mA		21 °C
Oscillator stopped	I_{opr}		1 mA		21 °C
Power consumption of PLL only with $f_{OSC} = 74$ MHz and $M = 1$	I_{opr}		7 mA		21 °C
Power consumption of PLL only with $f_{OSC} = 74$ MHz and $M = 3$	I_{opr}		5 mA		21 °C

Thermal package characteristics

Parameter	Symbol	Min.	Typ.	Max	Conditions
Junction to air	Θ_{JA}		+52 °C/W		Multi-Layer PCB
Junction to case	Θ_{JC}		+18 °C/W		Multi-Layer PCB



Chapter 12

XHFC-2SU package dimensions

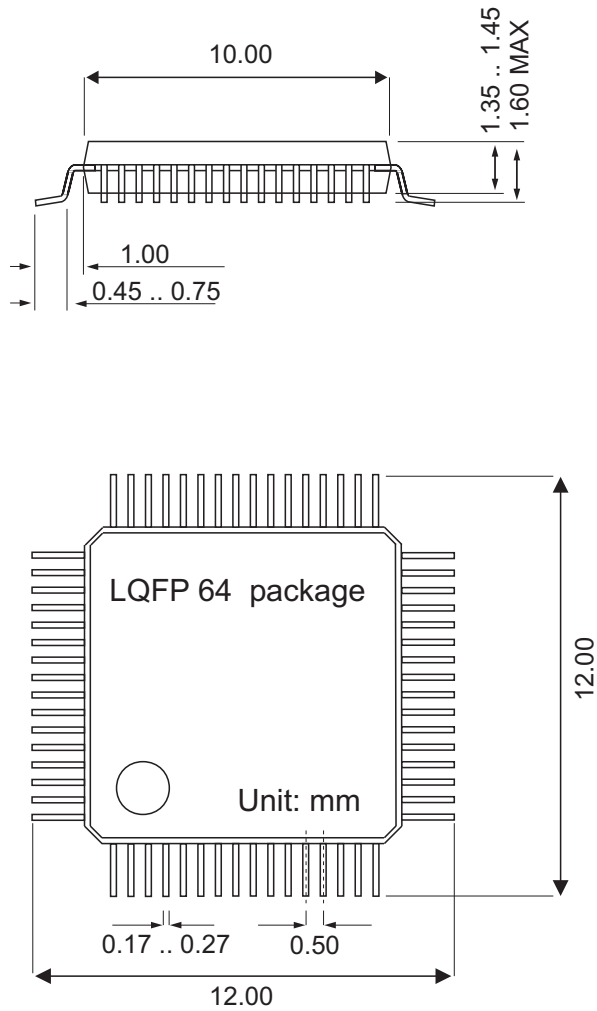
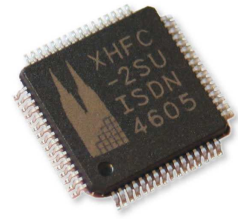


Figure 12.1: XHFC-2SU package dimensions



Chapter 13

XHFC-2SU address decoding erratum

13.1 Fault description

XHFC-2SU has an address decoding problem with write access to some registers. This erratum chapter describes the problem as well as work-arounds to avoid unwanted chip behaviour.

The address decoding fault refers to register addresses 0x48 .. 0x5F. Any write access to these registers writes also to their counterregisters at in the address range 0x08 .. 0x1F.

Due to the fact, that some addresses are not used by the chips, there are only few registers involved as shown in Table 13.1.

Table 13.1: *Involved registers of the address decoding fault*

Write access to		Writes also to	
0x48	R_GPIO_OUT0	0x08	R_RAM_ADDR
0x4C	R_GPIO_SEL	0x0C	R_FIFO_THRES
0x4D	R_SU_LED_CTRL	0x0D	R_FIFO_MD
0x50	R_PLL_CTRL	0x10	R_SLOT
0x51	R_PLL_P	0x11	R_MISC_IRQMSK
0x52	R_PLL_N	0x12	R_SU_IRQMSK
0x53	R_PLL_S	0x13	R_IRQ_CTRL

Read-only registers are not concerned in this fault.

13.2 Work-Around

The general work-around is to write registers in the address range 0x48 .. 0x5F only once during chip initialization. The counterregisters 0x08 .. 0x1F should be written afterwards. Exceptionally, register R_GPIO_OUT0 (0x48) can be written at any time, because it's counterregister R_RAM_ADDR (0x08) is normally not used.

Table 13.2 shows a complete list of the involved register addresses. Only gray marked lines are concerned in the decoding fault. In these cases, a detailed work-around is described to ensure a faultless chip behaviour.

Table 13.2: Detailed register list and work-around description

Write access to		Writes also to		Remark
0x48	R_GPIO_OUT0	0x08	R_RAM_ADDR	R_RAM_ADDR must be written again after R_GPIO_OUT0 write access. Typically, there is no reason for using register R_RAM_ADDR, so that any write access to register R_GPIO_OUT0 causes no problem.
0x49	–	0x09	R_RAM_CTRL	Address 0x49 is not used for write access.
0x4A	R_GPIO_EN0	0x0A	–	Counterregister 0x0A is not available.
0x4B	–	0x0B	R_FIRST_FIFO	Address 0x4B is not used for write access.
0x4C	R_GPIO_SEL	0x0C	R_FIFO_THRES	Register R_GPIO_SEL should be written during chip initialization. Typically, there is no need to write R_GPIO_SEL later once more. If necessary, register R_FIFO_THRES must be rewritten afterwards and unwanted FIFO fill levels might be reported in registers R_FILL_BL0..R_FILL_BL3. Software should always check the actual FIFO fill level before writing or reading FIFO bytes and should not rely on the expected FIFO fill level threshold.

(continued on next page)

Table 13.2: Detailed register list and work-around description

(continued from previous page)

Write access to		Writes also to		Remark
0x4D	R_SU_LED_CTRL	0x0D	R_FIFO_MD	<p>Register R_SU_LED_CTRL should be written during chip initialization. Typically, there is no need to write R_SU_LED_CTRL later once more. If necessary in particular applications, write access to R_SU_LED_CTRL is only allowed when no FIFO is in use because this destroys the overall FIFO setup. Write access to this pair of registers must fulfill strong timing constraints as follows:</p> <ol style="list-style-type: none"> 1. Configure FIFO mode by writing R_FIFO_MD. 2. Initiate a global software reset to accept this setting. 3. Configure R_SU_LED_CTRL within a not-busy phase ($V_BUSY = '0'$ in register R_STATUS). This write access is not allowed while $V_BUSY = '1'$. 4. Re-write R_FIFO_MD within the same not-busy phase. As the FIFO setup is evaluated only during busy phases, it is important to restore the value immediately after R_SU_LED_CTRL write access.
0x4E	–	0x0E	A_INC_RES_FIFO	Addresses 0x4E and 0x4F are not used for write access.
0x4F	–	0x0F	–	
0x50	R_PLL_CTRL	0x10	R_SLOT	<p>Typically, register R_PLL_CTRL is written during initialization. R_SLOT should be written afterwards.</p> <p>Applications that use the PLL's enable/disable function should ensure that PCM time slot selection with register R_SLOT and following access to PCM slot array registers is not interrupted by R_PLL_CTRL write access.</p>
0x51	R_PLL_P	0x11	R_MISC_IRQMSK	It is strongly recommended to write registers R_PLL_P, R_PLL_N and R_PLL_S during initialization only. The counterpart registers R_MISC_IRQMSK, R_SU_IRQMSK and R_IRQ_CTRL should be initialized afterwards to avoid unwanted interrupt generation, interrupt suppression or unwanted interrupt controller behaviour.
0x52	R_PLL_N	0x12	R_SU_IRQMSK	
0x53	R_PLL_S	0x13	R_IRQ_CTRL	

(continued on next page)

Table 13.2: Detailed register list and work-around description

(continued from previous page)

Write access to	Writes also to	Remark
0x54 –	0x14 R_PCM_MD0	Addresses 0x54..0x5F are not used for write access.
0x55 –	0x15 (11 multi-registers)	
0x56 –	0x16 R_SU_SEL	
0x57 –	0x17 R_SU_SYNC	
0x58 –	0x18 –	
0x59 –	0x19 –	
0x5A –	0x1A R_TI_WD	
0x5B –	0x1B R_BERT_WD_MD	
0x5C –	0x1C –	
0x5D –	0x1D –	
0x5E –	0x1E R_PWM_CFG	
0x5F –	0x1F –	

References

- [1] Cologne Chip AG. *DIGICCTM PLL Technology*, November 2004.
- [2] Cologne Chip AG. *C3-PLL-2– Phase-locked Loop (PLL) Frequency Synthesizer IP Core*, June 2005.
- [3] Electronic Manufacturing Industry Association of Germany, Information & Communication Technology Group (ZVEI Zentralverband Elektrotechnik, Fachverband Informations- und Kommunikationstechnik (I+K Forum)). *ZVEI Documentation DKZ-N; Interfaces and signaling protocols for ISDN telecommunication installations. (only available in German under title: ZVEI-Dokumentation DKZ-N; Schnittstellen und Signalisierungsprotokolle für Telekommunikationsanlagen im ISDN)*, Mai 1989. Volume IV: DKZ-N part 1.2, DKZ-N2 part 2.2.
- [4] European Telecommunications Standards Institute. *Technical Basis for Regulation (TBR 3): Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment to connect to an ISDN using ISDN basic access*.
- [5] European Telecommunications Standards Institute. *Technical Basis for Regulation (TBR 4): Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment to connect to an ISDN using ISDN primary rate access*.
- [6] Infineon Technologies. *PEB 2096 Data Sheet; ICs for Communications: Octal Transceiver for Upn Interfaces (OCTAT-P)*, 2.1 edition, April 1999.
- [7] Siemens AG. *ICs for Communication. IOM[®]-2 Interface Reference Guide*, 3 1991.
- [8] Telecommunication Standardization Sector of International Telecommunication Union (ITU). *ITU-T V.27: Data communication over the telephone network; 4800 bits per second MODEM with manual equalizer standardized for use on leased telephone-type circuits*, 1988, 1993.
- [9] Telecommunication Standardization Sector of International Telecommunication Union (ITU). *ITU-T I.430: Integrated services digital network (ISDN); ISDN user-network interfaces. Basic user-network interface – Layer 1 specification*, November 1995.
- [10] Telecommunication Standardization Union (ITU). O.151: Error performance measuring equipment operating at the primary rate and above. In *Specification of measurement equipment*. ITU-T (Telecommunication Standardization Sector of ITU), 1992.
- [11] Telecommunication Standardization Union (ITU). O.150: Equipment for the measurement of digital and analogue/digital parameters. In *General requirements for instrumentation for performance measurements on digital transmission equipment*. ITU-T (Telecommunication Standardization Sector of ITU), 1996.

List of register and bitmap abbreviations

This list shows all abbreviations which are used to define the register and bitmap names. Appended digits are not shown here except they have a particular meaning.

16KHZ	16 kHz	CHANNEL	HFC-channel	E	E-channel
2KHZ	2 kHz	CHIP	microchip	ECH	error counter, high byte
AB	A/B-bit	CI	Command/Indication (C/I channel of the GCI interface)	ECL	error counter, low byte
ABO	abort	CI6	6 bit C/I-channel length	EN	enable
ACT	active, activation	CIRM	configuration, interrupt and reset	END	end
ADDR	address	CLK	clock	ERR	error
ADJ	adjust	CLKO	clock output	EV	event
AF0	alternative frame synchronization signal	CNT	counter	EXCHG	exchange
AUTO	automatic	CNTH	counter, high byte	EXP	expired
B1	B1-channel	CNTL	counter, low byte	F	<i>F</i> -counter
B12	B1- and B2-channels	CON	connection settings	F0	frame synchronization signal
B2	B2-channel	CONT	contention	F1	F1_1 pin
BAC	BAC-bit	CRC	cyclic redundancy check	F1	<i>F1</i> -counter
BERT	bit error rate test	CTRL	control	F2	<i>F2</i> -counter
BIT	bit	D	D-channel	FDIR	direction (FIFO-related)
BL	block	DATA	data	FIFO	FIFO
BUSY	busy	DC	DC-balancing bit	FILL	fill level
C	command bits of the C/I-channel	DF	data flow	FIRST	first
C2I	C2 clock input (PCM bit clock)	DIR	direction	FLOW	flow
C2O	C2 clock output (PCM bit clock)	DLL	double last look criterion	FNUM	number (FIFO-related)
C4	C4IO clock (PCM double bit clock)	DLY	delay	FR	frame
CFG	configuration	DLYH	delay, high byte	FREEZE	freeze
CH	HFC-channel	DLYL	delay, low byte	FRQ	frequency
		DONE	done	FSM	FIFO sequence mode
		DR	data rate		
		DST	destination	G2	G2 state

G3	G3 state	MD	mode	RDY	ready
GCI	GCI interface	MERGE	merge	REG	register
GLOB	global	MISC	miscellaneous	REP	repetition
GPIO	general purpose input / output	MIX	mixed	RES	reset
GRD	guard	MOD	modification	REV	reverse
HDLC	high-level data link control	MON	monitor channel of the GCI interface	ROT	Rotator, rotation
HFC	HDLC FIFO controller	MR	handshake bit MR	ROUT	routing
HI	high	MS	multiframe / superframe	RPT	repeat
HPRIO9	high priority, 9 bits	MSK	mask	RV	revision
I	indication bits of the C/I-channel	MSS	multiframe / superframe synchronization	RX	receive
ICR	increase	MULT	multiple	RXHS	receiver handshake
ID	identifier	MX	handshake bit MX	RXR	receiver ready
IDX	index	N	PLL loop factor	S	S-bit
IFF	inter frame fill	NEG	negative	S	PLL post-scaler
IGNO	ignore	NEXT	next	SCRM	scrambler
IN	input	NINV	no inversion	SDIR	direction (slot-related)
INC	increment	NO	no	SEL	select, selection
INFO0	INFO 0 line condition (no signal)	NOINC	no increment	SEQ	sequence
INIT	initialization	NRES	active low reset	SET	setup
INT	internal	NT	NT mode	SG	S/G-bit
INV	invert, inversion	NUM	number	SH	shape
IRQ	interrupt	OD	open drain output	SH0H	shape 0, high byte
IRQMSK	interrupt mask	OFF	off	SH0L	shape 0, low byte
IRQSTA	interrupt status	OFFS	offset	SH1H	shape 1, high byte
LD	load	OSC	oscillator	SH1L	shape 1, low byte
LED	Light emitting diode	OUT	output	SIG	signal
LEN	length	OVIEW	overview	SL	time slot
LO	low	P	PLL predivider	SLIP	frequency slip
LOCK	locked	PAT	pattern	SLOT	time slot
LOOP	loop	PCM	pulse code modulation	SLOW	slow
LOST	frame data lost	PLL	phase locked loop	SMPL	sample
LPRIO	low priority	POL	polarity	SNUM	number (slot-related)
LPRIO11	low priority, 11 bits	PROC	processing	SQ	S/Q-bits
M	M-bit	PU	pulse	SRAM	SRAM
M	divider value M	PULSE	pulse	SRC	source
MAN	manual	PWM	pulse width modulation	SRES	software reset
MAX	maximum	RAM	RAM	SSYNC	single synchronization pulse
		RD	read	ST	S/T interface
				STA	state, status
				START	start
				STATUS	status
				STIO	STIO pins
				STOP	stop
				STR	strict

STUP	S/T / U _p interface)	THRES	threshold	USAGE	usage
SU	Universal ISDN Port (combined S/T and U _p interface)	TI	timer	USE	use, usage
SUBCH	subchannel	TRI	tristate	VAL	value
SWAP	swap	TRP	transparent	VIO	code violation
SYNC	synchronize, synchronization	TS	time step	WAIT	wait
SYNCI	synchronization input signal	TST	test	WAK	wakeup
T	T-bits	TX	transmit	WD	watchdog timer
T2	S/T timer T2	TXHS	transmitter handshake	WR	write
		TXR	transmitter ready	Z1	Z1-counter
		UNIDIR	unidirectional	Z2	Z2-counter
		UP	U _p interface		

Index of register and bitmap names

Index entries are sorted by name. Pages of the register tables are printed in bold type.

A_B1_RX	186, 204, 213, 217 , 218	Ch. 4: 126, 127, 129, 131, 136
Ch. 5: 159, 162, 177, 178, 186, 215	A_D_TX	Ch. 9: 318
A_B1_TX	Ch. 5: 159, 177, 186, 198, 200, 207	A_MS_DF
Ch. 5: 159, 177, 186, 206	A_E_RX	Ch. 5: 198, 204 , 217, 218
A_B2_RX	Ch. 5: 159, 161, 162, 177, 178, 186, 204, 213, 217, 218	A_MS_RX
Ch. 5: 159, 162, 177, 178, 186, 216	A_F1	Ch. 5: 157, 161, 167, 176, 177, 183, 213 , 217, 218
A_B2_TX	Ch. 3: 86	A_MS_TX
Ch. 5: 159, 177, 186, 206	Ch. 4: 138	Ch. 5: 157, 161, 162, 167, 176–178, 183, 200 , 204, 207, 208, 213
A_BAC_S_TX	A_F2	A_SL_CFG
Ch. 5: 159, 177, 186, 198, 200, 208	Ch. 3: 86	Ch. 2: 45
A_CHANNEL	Ch. 4: 139	Ch. 3: 82–84, 89–92, 96, 97, 105, 116, 117, 122, 123
Ch. 2: 45	A_FIFO_CTRL	Ch. 6: 221–223, 246, 248, 270, 274
Ch. 3: 83, 84, 86, 93, 95–97, 99, 101, 103, 104, 106, 107, 119, 120, 122, 123	Ch. 2: 45	A_ST_CTRL3
Ch. 4: 150	Ch. 3: 86, 101	Ch. 5: 156, 196, 201 , 202
A_CH_MSK	Ch. 4: 140, 147, 152	A_SUBCH_CFG
Ch. 2: 45	Ch. 8: 280, 283	Ch. 2: 45
Ch. 3: 86, 87, 94, 101, 102, 108, 109, 113–116, 118, 119, 121, 122	Ch. 9: 311, 319–322	Ch. 3: 86, 87, 93, 94, 101, 102, 108, 109, 113, 114, 116, 117, 119–123
Ch. 4: 136, 146	A_FIFO_DATA	Ch. 4: 132, 146, 149
A_CON_HDLC	Ch. 3: 86	A_SU_CLK_DLY
Ch. 2: 45	Ch. 4: 128, 145	Ch. 5: 158, 163, 167, 179, 183, 205
Ch. 3: 77, 79, 86, 87, 89–91, 94–97, 101, 102, 104–106, 114, 116, 117, 119, 120, 122, 123	A_FIFO_DATA_NOINC	A_SU_CTRL0
Ch. 4: 126, 128, 131, 147 , 152	Ch. 3: 86	Ch. 5: 159, 167, 176, 183, 195 , 199, 214
Ch. 9: 300	Ch. 4: 128, 131, 145	Ch. 9: 302
A_D_RX	A_FIFO_SEQ	A_SU_CTRL1
Ch. 5: 159, 162, 177, 178,	Ch. 2: 45	Ch. 4: 127
	Ch. 3: 86, 101, 103, 104, 106, 107	Ch. 5: 163, 164, 167, 180,
	Ch. 4: 151	
	A_FIFO_STA	
	Ch. 4: 136, 140 , 153	
	A_INC_RES_FIFO	
	Ch. 13: 348	
	Ch. 3: 86	

182, 183, 197	Ch. 9: 288, 298, 306	Ch. 3: 99–103
A_SU_CTRL2	R_CI_RX	Ch. 4: 127, 133
Ch. 5: 159, 167, 169, 176, 183, 195, 198	Ch. 6: 243, 272	R_FSM_IDX
Ch. 9: 302	R_CI_TX	Ch. 3: 86, 99, 101, 103, 104, 106, 107
A_SU_DLYH	Ch. 6: 244, 267	Ch. 4: 137
Ch. 5: 167, 183, 211, 212	R_CLK_CFG	R_GCI_CFG0
A_SU_DLYL	Ch. 9: 288, 292, 308	Ch. 6: 243–246, 248, 268
Ch. 5: 167, 183, 211 , 212	R_CTRL	R_GCI_CFG1
A_SU_RD_STA	Ch. 2: 70	Ch. 6: 246, 248, 269, 270
Ch. 5: 163, 167, 179, 182, 183, 194, 210	Ch. 9: 288	R_GCI_STA
A_SU_STA	R_F0_CNTH	Ch. 6: 244, 252, 273
Ch. 5: 209, 214	Ch. 6: 271	R_GPIO_EN0
Ch. 9: 302, 316	R_F0_CNTL	Ch. 10: 334
A_SU_WR_STA	Ch. 6: 271	Ch. 13: 347
Ch. 5: 163, 164, 167, 179, 180, 182, 183, 194 , 197, 210	R_FIFO	R_GPIO_EN1
A_UP_CTRL3	Ch. 3: 79, 84–86, 89–91, 93, 95–97, 99, 101, 102, 104–106, 109, 113–123	Ch. 10: 326, 331
Ch. 5: 156, 182, 201, 202	Ch. 4: 126, 130, 131, 136, 137 , 138–141, 145–152	R_GPIO_EN2
A_USAGE	R_FIFO_BL0_IRQ	Ch. 10: 332
Ch. 4: 141	Ch. 4: 152	R_GPIO_IN0
A_Z1	Ch. 9: 300, 314, 319	Ch. 10: 326, 337
Ch. 3: 86	R_FIFO_BL1_IRQ	R_GPIO_IN2
Ch. 4: 138	Ch. 9: 314, 320	Ch. 10: 337
A_Z2	R_FIFO_BL2_IRQ	R_GPIO_OUT0
Ch. 3: 86	Ch. 9: 314, 321	Ch. 10: 333
Ch. 4: 138	R_FIFO_BL3_IRQ	Ch. 13: 346, 347
R_AF0_OVIEW	Ch. 4: 152	R_GPIO_OUT1
Ch. 5: 209 , 215	Ch. 9: 300, 314, 319, 322	Ch. 10: 330
Ch. 9: 302, 316	R_FIFO_MD	R_GPIO_OUT2
R_BERT_ECH	Ch. 13: 346, 348	Ch. 10: 332
Ch. 8: 280, 283, 285, 286	Ch. 3: 85, 93, 99	R_GPIO_SEL
R_BERT_ECL	Ch. 4: 128, 133, 135 , 137	Ch. 10: 326, 335
Ch. 8: 280, 283, 285 , 286	R_FIFO_THRES	Ch. 13: 346, 347
R_BERT_STA	Ch. 13: 346, 347	R_GPIO_SEL_BL
Ch. 6: 228, 266	Ch. 4: 134 , 141–144	Ch. 10: 331
Ch. 8: 280, 283, 285	R_FILL_BL0	R_INT_DATA
R_BERT_WD_MD	Ch. 13: 347	Ch. 2: 45, 73
Ch. 13: 349	Ch. 4: 141	R_IRQ_CTRL
Ch. 8: 280, 284	R_FILL_BL1	Ch. 13: 346, 348
Ch. 9: 305	Ch. 4: 142	Ch. 4: 152
R_CHIP_ID	R_FILL_BL2	Ch. 9: 299, 300, 311 , 319–322
Ch. 2: 72	Ch. 4: 143	R_IRQ_OVIEW
R_CHIP_RV	R_FILL_BL3	Ch. 9: 300, 302, 314 , 318
Ch. 2: 72	Ch. 13: 347	R_MISC_IRQ
R_CIRM	Ch. 4: 144	Ch. 6: 244
Ch. 4: 126	R_FIRST_FIFO	Ch. 9: 302–304, 309, 315,
	Ch. 13: 347	

	316, 318			Ch. 9: 298, 315, 316, 318
R_MISC_IRQMSK	Ch. 13: 346, 348	R_PWM_CFG	Ch. 13: 349	R_SU_IRQ
Ch. 5: 209, 215		Ch. 7: 276, 277		Ch. 5: 167, 183, 186
Ch. 6: 243, 244		R_PWM_MD		Ch. 9: 299, 310, 315, 317
Ch. 9: 302–304, 309 , 316		Ch. 7: 276, 278		R_SU_IRQMSK
R_MON_RX		Ch. 9: 303, 318		Ch. 13: 346, 348
Ch. 6: 243, 244, 252, 273		R_RAM_ADDR		Ch. 5: 186
Ch. 9: 304		Ch. 13: 346, 347		Ch. 9: 299, 310 , 317
R_MON_TX		Ch. 2: 71 , 74		R_SU_LED_CTRL
Ch. 6: 244, 249, 250, 252,		R_RAM_CTRL		Ch. 10: 329, 336
270 , 273		Ch. 13: 347		Ch. 13: 346, 348
Ch. 9: 304		Ch. 2: 71 , 74		R_SU_SEL
R_MSS0		R_RAM_DATA		Ch. 13: 349
Ch. 6: 235, 237–239, 254,		Ch. 2: 45, 71, 74		Ch. 5: 156, 193 , 194, 195,
257		R_RAM_USE		197, 198, 200–202,
R_MSS1		Ch. 2: 72		204–208, 210–218
Ch. 5: 198		R_SH0H		R_SU_SYNC
Ch. 6: 237, 254, 263		Ch. 6: 240, 241, 254–256,		Ch. 13: 349
R_PCM_MD0		264		Ch. 6: 228, 261, 266
Ch. 13: 349		R_SH0L		R_TI_WD
Ch. 6: 220, 223, 239–241,		Ch. 6: 240, 241, 254–256,		Ch. 13: 349
246, 248, 254 , 255–257,		264		Ch. 9: 303, 305, 312
259, 261, 263–265		R_SH1H		V_ABO_DONE
R_PCM_MD1		Ch. 6: 241, 254–256, 265		Ch. 4: 136, 140
Ch. 6: 220, 222, 228, 230,		R_SH1L		V_ADDR_INC
253, 254, 259 , 261, 271,		Ch. 6: 241, 254–256, 264		Ch. 2: 71
274		R_SLOT		V_ADDR_RES
R_PCM_MD2		Ch. 13: 346, 348		Ch. 2: 71
Ch. 6: 223, 228, 230, 254,		Ch. 3: 82–84, 89–92, 96, 97,		V_AUTO_SYNCI
261		105, 116, 117, 122, 123		Ch. 6: 228, 229, 266
R_PLL_CTRL		Ch. 6: 246, 248, 253 , 270,		V_AUTO_WD_RES
Ch. 13: 346, 348		274		Ch. 8: 284
Ch. 9: 294, 313		R_SL_MAX		Ch. 9: 305
R_PLL_N		Ch. 6: 271		V_B12_SWAP
Ch. 13: 346, 348		R_SL_SEL0		Ch. 5: 161, 162, 177, 178,
Ch. 9: 323		Ch. 6: 240, 254, 255		197
R_PLL_P		R_SL_SEL1		V_B1_RX
Ch. 13: 346, 348		Ch. 6: 240, 241, 254, 255,		Ch. 5: 215
Ch. 9: 323		256		V_B1_RX_EN
R_PLL_S		R_SL_SEL7		Ch. 5: 159, 162, 176, 178,
Ch. 13: 346, 348		Ch. 2: 70		198
Ch. 9: 323		Ch. 5: 166, 182		V_B1_TX
R_PLL_STA		Ch. 6: 254, 256		Ch. 5: 206
Ch. 9: 322		R_STATUS		V_B1_TX_EN
R_PWM0		Ch. 13: 348		Ch. 5: 159, 161, 176, 177,
Ch. 7: 276, 277		Ch. 4: 127		195
R_PWM1		Ch. 5: 186, 206–208,		V_B2_RX
Ch. 7: 276, 277, 278		215–218		Ch. 5: 216
		Ch. 6: 223		V_B2_RX_EN

Ch. 5: 159, 162, 176, 178, 198	104, 106, 107, 119, 120, 122, 123	V_D_MERGE_TX Ch. 6: 266
V_B2_TX Ch. 5: 206	Ch. 4: 150	V_D_RES Ch. 4: 127 Ch. 5: 161, 177, 197
V_B2_TX_EN Ch. 5: 159, 161, 176, 177, 195	V_CH_FNUM Ch. 3: 84, 93, 95–97, 103, 104, 106, 107, 119, 120, 122, 123	V_D_RX Ch. 5: 217
V_BAC_D Ch. 5: 161, 197	Ch. 4: 150	V_D_RX_AB Ch. 5: 217
V_BAC_NINV Ch. 5: 161, 177, 204	V_CH_MSK Ch. 3: 112–114, 116, 119, 122	V_D_RX_S Ch. 5: 217
V_BAC_S_SEL Ch. 5: 161, 177, 198 , 204	Ch. 4: 146	V_D_RX_SG Ch. 5: 217
V_BAC_TX Ch. 5: 208	V_CH_SDIR Ch. 3: 82, 84, 89–92, 96, 97, 105, 116, 117, 122, 123	V_D_TX Ch. 5: 207
V_BERT_ECH Ch. 8: 282, 286	Ch. 6: 246, 248, 274	V_D_TX_BAC Ch. 5: 207
V_BERT_ECL Ch. 8: 282, 285	V_CH_SNUM Ch. 3: 82, 84, 89–92, 96, 97, 105, 116, 117, 122, 123	V_D_TX_S Ch. 5: 207 , 208
V_BERT_EN Ch. 4: 152 Ch. 8: 280–283	Ch. 6: 246, 248, 274	V_EV_TS Ch. 9: 303, 312
V_BERT_ERR Ch. 8: 280, 281, 284	V_CI_IRQ Ch. 6: 244 Ch. 9: 302, 316	V_E_MERGE_RX Ch. 6: 266
V_BERT_INV_DATA Ch. 8: 282, 283, 285	V_CI_IRQMSK Ch. 6: 243, 244 Ch. 9: 302, 309	V_E_RX Ch. 5: 218
V_BERT_SYNC Ch. 8: 280, 282, 285	V_CLKO_HI Ch. 9: 288, 289, 292, 308	V_E_RX_AB Ch. 5: 218
V_BIT_CNT Ch. 3: 108–111, 114, 116, 117, 119, 120, 122, 123 Ch. 4: 146, 149 Ch. 8: 280–283	V_CLKO_OFF Ch. 9: 288, 289, 292, 308	V_E_RX_SG Ch. 5: 218
V_BUSY Ch. 13: 348 Ch. 4: 127 Ch. 9: 298, 318	V_CLKO_PLL Ch. 9: 288, 289, 292, 308	V_F0_CNTH Ch. 6: 271
V_C2I_EN Ch. 6: 223, 229, 230, 261	V_CLK_F1 Ch. 9: 288, 289, 292, 308	V_F0_CNTL Ch. 6: 271
V_C2O_EN Ch. 6: 229, 230, 261	V_CLK_OFF Ch. 9: 288, 289, 306	V_F0_LEN Ch. 6: 223, 229, 236, 238, 239, 254
V_C4_POL Ch. 6: 229, 254	V_CLK_PLL Ch. 9: 288, 289, 308	V_F0_NEG Ch. 6: 229, 254
V_CHIP_ID Ch. 2: 72	V_DATA_FLOW Ch. 3: 77, 79–81, 89–91, 95–97, 104–106, 114, 116, 117, 119, 120, 122, 123	V_F1 Ch. 4: 138
V_CHIP_RV Ch. 2: 72	Ch. 4: 148	V_F2 Ch. 4: 139
V_CH_FDIR Ch. 3: 84, 93, 95–97, 103,	V_DF_MD Ch. 3: 85, 93, 99 Ch. 4: 135 , 137	V_FIFO0_RX_IRQ Ch. 9: 301, 319
	V_D_MERGE_RX Ch. 6: 266	V_FIFO0_TX_IRQ Ch. 9: 301, 319
		V_FIFO10_RX_IRQ

Ch. 9: 321	V_FIFO7_TX_IRQ	V_FIFO_NUM
V_FIFO10_TX_IRQ	Ch. 9: 320	Ch. 3: 84, 89–91, 95–97, 104–106, 114, 116, 117, 119, 120, 122, 123
Ch. 9: 321	V_FIFO8_RX_IRQ	Ch. 4: 137
V_FIFO11_RX_IRQ	Ch. 9: 321	V_FILL_FIFO0_RX
Ch. 9: 321	V_FIFO8_TX_IRQ	Ch. 4: 141
V_FIFO11_TX_IRQ	Ch. 9: 321	V_FILL_FIFO0_TX
Ch. 9: 321	V_FIFO9_RX_IRQ	Ch. 4: 141
V_FIFO12_RX_IRQ	Ch. 9: 321	V_FILL_FIFO10_RX
Ch. 9: 322	V_FIFO9_TX_IRQ	Ch. 4: 143
V_FIFO12_TX_IRQ	Ch. 9: 321	V_FILL_FIFO10_TX
Ch. 9: 322	V_FIFO_BL0_IRQ	Ch. 4: 143
V_FIFO13_RX_IRQ	Ch. 9: 300, 301, 314	V_FILL_FIFO11_RX
Ch. 9: 322	V_FIFO_BL1_IRQ	Ch. 4: 143
V_FIFO13_TX_IRQ	Ch. 9: 301, 314	V_FILL_FIFO11_TX
Ch. 9: 322	V_FIFO_BL2_IRQ	Ch. 4: 143
V_FIFO14_RX_IRQ	Ch. 9: 301, 314	V_FILL_FIFO12_RX
Ch. 9: 301, 322	V_FIFO_BL3_IRQ	Ch. 4: 144
V_FIFO14_TX_IRQ	Ch. 9: 300, 301, 314	V_FILL_FIFO12_TX
Ch. 9: 322	V_FIFO_DATA	Ch. 4: 144
V_FIFO15_RX_IRQ	Ch. 4: 145	V_FILL_FIFO13_RX
Ch. 9: 301, 322	V_FIFO_DATA_NOINC	Ch. 4: 144
V_FIFO15_TX_IRQ	Ch. 4: 145	V_FILL_FIFO13_TX
Ch. 9: 301, 322	V_FIFO_DIR	Ch. 4: 144
V_FIFO1_RX_IRQ	Ch. 3: 79, 81, 84, 89–91, 93, 95–97, 104–106, 114, 116, 117, 119, 120, 122, 123	V_FILL_FIFO14_RX
Ch. 9: 301, 319	Ch. 4: 137 , 148	Ch. 4: 144
V_FIFO1_TX_IRQ	V_FIFO_ERR	V_FILL_FIFO14_TX
Ch. 9: 301, 319	Ch. 4: 136, 140 , 153	Ch. 4: 144
V_FIFO2_RX_IRQ	V_FIFO_IRQ	V_FILL_FIFO15_RX
Ch. 9: 319	Ch. 3: 87, 89–91, 94–97, 102–106, 114, 116, 117, 119, 120, 122, 123	Ch. 4: 144
V_FIFO2_TX_IRQ	Ch. 4: 126, 147 , 148, 152	V_FILL_FIFO15_TX
Ch. 9: 301, 319	Ch. 9: 300, 319–322	Ch. 4: 144
V_FIFO3_RX_IRQ	V_FIFO_IRQMSK	V_FILL_FIFO1_RX
Ch. 9: 319	Ch. 4: 152	Ch. 4: 141
V_FIFO3_TX_IRQ	Ch. 9: 311, 319–322	V_FILL_FIFO1_TX
Ch. 9: 319	V_FIFO_IRQSTA	Ch. 4: 141
V_FIFO4_RX_IRQ	Ch. 9: 319	V_FILL_FIFO2_RX
Ch. 9: 320	V_FIFO_IRQ_EN	Ch. 4: 141
V_FIFO4_TX_IRQ	Ch. 4: 152	V_FILL_FIFO3_RX
Ch. 9: 320	Ch. 9: 300, 301, 311 , 319–322	Ch. 4: 141
V_FIFO5_RX_IRQ	V_FIFO_LPRIO	V_FILL_FIFO3_TX
Ch. 9: 320	Ch. 2: 70	Ch. 4: 141
V_FIFO5_TX_IRQ	V_FIFO_MD	V_FILL_FIFO4_RX
Ch. 9: 320	Ch. 4: 128, 135	Ch. 4: 142
V_FIFO6_RX_IRQ		V_FILL_FIFO4_TX
Ch. 9: 320		Ch. 4: 142
V_FIFO6_TX_IRQ		
Ch. 9: 320		
V_FIFO7_RX_IRQ		
Ch. 9: 320		

V_FILL_FIFO5_RX	270	V_GPIO_EN8
Ch. 4: 142	V_GCI_SWAP_RXHS	Ch. 10: 326, 327, 331
V_FILL_FIFO5_TX	Ch. 6: 245, 248, 268 , 269	V_GPIO_EN9
Ch. 4: 142	V_GCI_SWAP_STIO	Ch. 10: 327, 331
V_FILL_FIFO6_RX	Ch. 6: 245, 248, 268, 269	V_GPIO_IN0
Ch. 4: 142	V_GCI_SWAP_TXHS	Ch. 10: 329, 338
V_FILL_FIFO6_TX	Ch. 6: 245, 248, 268 , 269	V_GPIO_IN1
Ch. 4: 142	V_GLOB_IRQ_EN	Ch. 10: 338
V_FILL_FIFO7_RX	Ch. 9: 299, 311	V_GPIO_IN10
Ch. 4: 142	V_GPIO_BL0	Ch. 10: 337
V_FILL_FIFO7_TX	Ch. 10: 327, 331	V_GPIO_IN11
Ch. 4: 142	V_GPIO_BL1	Ch. 10: 337
V_FILL_FIFO8_RX	Ch. 10: 331	V_GPIO_IN12
Ch. 4: 143	V_GPIO_EN0	Ch. 10: 337
V_FILL_FIFO8_TX	Ch. 10: 329, 334	V_GPIO_IN13
Ch. 4: 143	V_GPIO_EN1	Ch. 10: 337
V_FILL_FIFO9_RX	Ch. 10: 334	V_GPIO_IN14
Ch. 4: 143	V_GPIO_EN10	Ch. 10: 337
V_FILL_FIFO9_TX	Ch. 10: 331	V_GPIO_IN15
Ch. 4: 143	V_GPIO_EN11	Ch. 10: 337
V_FIRST_FIFO_DIR	Ch. 10: 331	V_GPIO_IN16
Ch. 3: 101, 103	V_GPIO_EN12	Ch. 10: 327, 337
Ch. 4: 127, 133	Ch. 10: 331	V_GPIO_IN17
V_FIRST_FIFO_NUM	V_GPIO_EN13	Ch. 10: 337
Ch. 3: 101, 103	Ch. 10: 331	V_GPIO_IN18
Ch. 4: 127, 133	V_GPIO_EN14	Ch. 10: 337
V_FR_ABO	Ch. 10: 331	V_GPIO_IN19
Ch. 4: 140, 153	V_GPIO_EN15	Ch. 10: 337
V_G2_G3	Ch. 10: 331	V_GPIO_IN2
Ch. 5: 163, 182, 210	V_GPIO_EN16	Ch. 10: 338
V_G2_G3_EN	Ch. 10: 327, 332	V_GPIO_IN3
Ch. 5: 163, 164, 180, 182,	V_GPIO_EN17	Ch. 10: 338
197	Ch. 10: 332	V_GPIO_IN4
V_GCI_ABO	V_GPIO_EN18	Ch. 10: 338
Ch. 6: 252, 273	Ch. 10: 332	V_GPIO_IN5
V_GCI_C	V_GPIO_EN19	Ch. 10: 338
Ch. 6: 244, 267	Ch. 10: 332	V_GPIO_IN6
V_GCI_EN	V_GPIO_EN2	Ch. 10: 338
Ch. 6: 243, 246, 248, 269	Ch. 10: 334	V_GPIO_IN7
V_GCI_I	V_GPIO_EN3	Ch. 10: 338
Ch. 6: 244, 272	Ch. 10: 334	V_GPIO_IN8
V_GCI_MR	V_GPIO_EN4	Ch. 10: 327, 337
Ch. 6: 273	Ch. 10: 334	V_GPIO_IN9
V_GCI_MX	V_GPIO_EN5	Ch. 10: 327, 337
Ch. 6: 273	Ch. 10: 334	V_GPIO_OUT0
V_GCI_RX	V_GPIO_EN6	Ch. 10: 329, 333
Ch. 6: 273	Ch. 10: 334	V_GPIO_OUT1
V_GCI_SL	V_GPIO_EN7	Ch. 10: 333
Ch. 6: 242, 243, 246, 248,	Ch. 10: 334	V_GPIO_OUT10

Ch. 10: 330	V_GPIO_SEL7	Ch. 9: 318
V_GPIO_OUT11	Ch. 10: 335	V_MAN_SYNC
Ch. 10: 330	V_HDLC_TRP	Ch. 5: 187
V_GPIO_OUT12	Ch. 3: 87, 89–91, 95–97, 103–106, 114, 116, 117, 119, 120, 122, 123	Ch. 6: 228, 229, 266
Ch. 10: 330	Ch. 4: 131, 147	V_MISC_IRQ
V_GPIO_OUT13	Ch. 9: 300	Ch. 9: 302, 315 , 318
Ch. 10: 330	V_HFC_RES	V_MISC_IRQSTA
V_GPIO_OUT14	Ch. 4: 126	Ch. 9: 315, 318
Ch. 10: 330	Ch. 9: 298, 307	V_MIX_IRQ
V_GPIO_OUT15	V_IDX	Ch. 4: 147, 152
Ch. 10: 330	Ch. 3: 101, 103, 104, 106, 107	Ch. 9: 300, 319–322
V_GPIO_OUT16	Ch. 4: 137	V_MON_CI6
Ch. 10: 327, 332	V_IFF	Ch. 6: 244, 267, 268 , 272
V_GPIO_OUT17	Ch. 3: 89–91, 95–97, 104–106, 114, 116, 117, 119, 120, 122, 123	V_MON_DLL
Ch. 10: 332	Ch. 4: 128, 147	Ch. 6: 244, 268
V_GPIO_OUT18	V_INC_F	V_MON_END
Ch. 10: 332	Ch. 4: 129, 131, 136	Ch. 6: 244, 250, 268
V_GPIO_OUT19	V_INT_DATA	V_MON_RX
Ch. 10: 332	Ch. 2: 73	Ch. 6: 273
V_GPIO_OUT2	V_INV_DATA	V_MON_RXR
Ch. 10: 333	Ch. 3: 114, 116, 117, 119, 120, 122, 123	Ch. 6: 244, 252, 273
V_GPIO_OUT3	Ch. 4: 149	V_MON_RX_IRQ
Ch. 10: 333	V_IRQ_POL	Ch. 9: 302, 304, 316
V_GPIO_OUT4	Ch. 9: 299, 311	V_MON_RX_IRQMSK
Ch. 10: 333	V_LED_EN0	Ch. 6: 243
V_GPIO_OUT5	Ch. 10: 329, 336	Ch. 9: 302, 304, 309
Ch. 10: 333	V_LED_EN1	V_MON_SLOW
V_GPIO_OUT6	Ch. 10: 336	Ch. 6: 268
Ch. 10: 333	V_LED_EN2	V_MON_TX
V_GPIO_OUT7	Ch. 10: 336	Ch. 6: 270
Ch. 10: 333	V_LED_EN3	V_MON_TXR
V_GPIO_OUT8	Ch. 10: 336	Ch. 6: 244, 250, 273
Ch. 10: 327, 330	V_LED_EN4	V_MON_TX_IRQ
V_GPIO_OUT9	Ch. 10: 336	Ch. 6: 244
Ch. 10: 327, 330	V_LED_EN5	Ch. 9: 302, 304, 316
V_GPIO_SEL0	Ch. 10: 336	V_MON_TX_IRQMSK
Ch. 10: 326, 329, 335	V_LED_ROT	Ch. 6: 244
V_GPIO_SEL1	Ch. 10: 329, 336	Ch. 9: 302, 304, 309
Ch. 10: 335	V_LOOP_FIFO	V_MSS_DLY
V_GPIO_SEL2	Ch. 3: 114, 116, 117, 119, 120, 122, 123	Ch. 6: 236–238, 263
Ch. 10: 335	Ch. 4: 149	V_MSS_MOD
V_GPIO_SEL3	V_LOST_STA	Ch. 6: 236, 238, 239, 257
Ch. 10: 335		V_MSS_MOD_REP
V_GPIO_SEL4		Ch. 6: 236, 238, 257
Ch. 10: 335		V_MSS_OFFS
V_GPIO_SEL5		Ch. 6: 236, 263
Ch. 10: 335		V_MSS_OUT_EN
V_GPIO_SEL6		Ch. 6: 236, 239, 257
Ch. 10: 335		V_MSS_OUT_REP

Ch. 6: 236, 239, 257	V_PCM_IDX	V_PWM0
V_MSS_SRC	Ch. 6: 240, 241, 254 ,	Ch. 7: 277
Ch. 6: 236, 237, 258	255–257, 259, 261,	V_PWM0_16KHZ
V_MSS_SRC_EN	263–265	Ch. 7: 276, 277
Ch. 6: 235–237, 257	V_PCM_INIT	V_PWM0_MD
V_MSS_SRC_GRD	Ch. 6: 223	Ch. 7: 276, 278
Ch. 6: 236, 237, 257	Ch. 9: 298, 318	V_PWM1
V_MS_RX	V_PCM_LOOP	Ch. 7: 278
Ch. 5: 176, 213	Ch. 6: 222, 259	V_PWM1_16KHZ
V_MS_RX_RDY	V_PCM_MD	Ch. 7: 276, 277
Ch. 5: 176, 213	Ch. 4: 128	V_PWM1_MD
V_MS_SSYNC1	Ch. 6: 220, 223, 229, 245,	Ch. 7: 276, 278
Ch. 5: 198	246, 248, 254	V_PWM_FRQ
Ch. 6: 236, 263	V_PCM_OD	Ch. 7: 276, 277
V_MS_SSYNC2	Ch. 6: 221, 222, 259 , 274	V_RAM_ADDR0
Ch. 5: 161, 177, 198	V_PCM_RES	Ch. 2: 71
Ch. 6: 263	Ch. 9: 298, 307	V_RAM_ADDR1
V_MS_TX	V_PCM_SMPL	Ch. 2: 71
Ch. 5: 176, 200 , 204, 207,	Ch. 6: 222, 260	V_RAM_DATA
208	V_PLL_ADJ	Ch. 2: 74
V_MS_TX_RDY	Ch. 6: 228–230, 259 , 261	V_RD_SYNC_SRC
Ch. 5: 176, 213	V_PLL_FREEZE	Ch. 6: 228, 266
V_MULT_SU	Ch. 9: 294, 313	Ch. 8: 285
Ch. 5: 193	V_PLL_ICR	V_RES_FIFO
V_M_S_SRC	Ch. 6: 228–230, 261 , 262	Ch. 4: 126, 127, 136
Ch. 5: 161, 177, 204	V_PLL_LOCK	V_RES_FIFO_ERR
V_NEXT_FIFO_DIR	Ch. 9: 294, 322	Ch. 4: 127, 136 , 140
Ch. 3: 103, 104, 106, 107	V_PLL_M	V_RES_LOST
Ch. 4: 151	Ch. 9: 294, 313	Ch. 4: 136
V_NEXT_FIFO_NUM	V_PLL_MAN	Ch. 9: 318
Ch. 3: 103, 104, 106, 107	Ch. 6: 228–230, 261, 262	V_REV
Ch. 4: 151	V_PLL_N	Ch. 3: 87, 89–91, 94–97,
V_NO_CRC	Ch. 9: 293, 294, 323	103–106, 114, 116, 117,
Ch. 4: 153	V_PLL_NRES	119, 120, 122, 123
V_NO_REP	Ch. 9: 294, 313	Ch. 4: 131, 137
Ch. 4: 153	V_PLL_P	V_ROUT
V_NT_SYNC	Ch. 9: 294, 323	Ch. 3: 89–92, 96, 97, 105,
Ch. 2: 70	V_PLL_S	116, 117, 122, 123
Ch. 6: 229	Ch. 9: 294, 323	Ch. 6: 221, 222, 246, 248,
V_OSC_OFF	V_PLL_TST	274
Ch. 2: 70	Ch. 9: 313	V_SEQ_END
Ch. 9: 289	V_PROC	Ch. 3: 101, 103, 104, 106,
V_PAT_SEQ	Ch. 5: 186, 206–208,	107
Ch. 8: 280–282, 284	215–218	Ch. 4: 151
V_PCM_CLK	Ch. 9: 316, 318	V_SG_AB_INV
Ch. 9: 288, 289, 308	V_PROC_IRQ	Ch. 5: 162, 178, 204
V_PCM_DR	Ch. 9: 302, 303, 316 , 318	V_SH0H
Ch. 6: 220, 229, 253, 259 ,	V_PROC_IRQMSK	Ch. 6: 241, 264
271	Ch. 9: 302, 303, 309	V_SH0L

Ch. 6: 241, 264	Ch. 5: 214	V_SU_DLYH
V_SH1H	V_ST_D_HPRIO9	Ch. 5: 212
Ch. 6: 241, 265	Ch. 5: 214	V_SU_DLYL
V_SH1L	V_ST_D_LPRIO	Ch. 5: 211
Ch. 6: 241, 264	Ch. 5: 195, 214	V_SU_EXCHG
V_SH_SEL0	V_ST_D_LPRIO11	Ch. 5: 161, 167, 169, 177, 183, 199
Ch. 6: 240, 255	Ch. 5: 214	V_SU_FR_SYNC
V_SH_SEL1	V_ST_E_IGNO	Ch. 5: 210
Ch. 6: 240, 241, 255, 256	Ch. 5: 161, 197	V_SU_INFO0
V_SLIP_IRQ	V_ST_E_LO	Ch. 5: 163, 210
Ch. 9: 302, 316	Ch. 5: 161, 197	V_SU_LD_STA
V_SLIP_IRQMSK	V_ST_PULSE	Ch. 5: 163, 179, 182, 194
Ch. 5: 209, 215	Ch. 5: 196, 201	V_SU_MD
Ch. 9: 302, 309	V_ST_PU_CTRL	Ch. 5: 167, 183, 195
V_SL_DIR	Ch. 5: 161, 196	Ch. 9: 302
Ch. 3: 84, 89–92, 96, 97, 105, 116, 117, 122, 123	V_ST_SEL	V_SU_RES
Ch. 6: 246, 248, 253	Ch. 5: 156, 201	Ch. 9: 298, 307
V_SL_MAX	V_ST_Smpl	V_SU_RX_VAL
Ch. 6: 271	Ch. 5: 205	Ch. 5: 162, 178, 204 , 217, 218
V_SL_NUM	V_ST_SQ_EN	V_SU_SEL
Ch. 3: 84, 89–92, 96, 97, 105, 116, 117, 122, 123	Ch. 5: 159, 161, 195	Ch. 5: 156, 193
Ch. 6: 246, 248, 253	V_SU0_AF0	V_SU_SET_G2_G3
V_SL_SEL0	Ch. 5: 209	Ch. 5: 164, 180, 194 , 197
Ch. 6: 240, 255	V_SU0_IRQ	V_SU_SET_STA
V_SL_SEL1	Ch. 5: 186	Ch. 5: 194
Ch. 6: 240, 241, 255, 256	Ch. 9: 300, 310, 317	V_SU_STA
V_SL_SEL7	V_SU0_IRQMSK	Ch. 5: 210
Ch. 6: 256	Ch. 5: 186	V_SU_STOP
V_SQ_T_DST	Ch. 9: 300, 310	Ch. 5: 162, 167, 178, 183, 196
Ch. 5: 162, 178, 204 , 217, 218	V_SU1_AF0	V_SU_SYNC_NT
V_SQ_T_SRC	Ch. 5: 209	Ch. 5: 167, 183, 198
Ch. 5: 161, 177, 204	V_SU1_IRQ	Ch. 9: 302
V_SRAM_USE	Ch. 5: 186	V_SU_T2_EXP
Ch. 2: 72	Ch. 9: 300, 310	Ch. 5: 210
V_SRES	V_SU1_IRQMSK	V_SU_TRI
Ch. 4: 126	Ch. 5: 186	Ch. 5: 161, 167, 177, 183, 199
Ch. 9: 298, 306	Ch. 9: 300, 310, 317	V_SU_TST_SIG
V_START_BIT	V_SU1_IRQMSK	Ch. 5: 161, 177, 195 , 199
Ch. 3: 108–111, 114–117, 119, 120, 122, 123	Ch. 5: 186	V_SYNC_OUT1
Ch. 4: 149	Ch. 9: 300, 310	Ch. 6: 229, 261
V_STUP_IRQ	V_SU_2KHZ	V_SYNC_OUT2
Ch. 9: 300, 315	Ch. 5: 161, 177, 195, 199	Ch. 6: 229, 261 , 262
V_ST_D_ACT	V_SU_ACT	V_SYNC_SEL
Ch. 5: 214	Ch. 5: 163, 179, 194	Ch. 5: 187
V_ST_D_CONT	V_SU_AF0	Ch. 6: 228, 229, 266
	Ch. 5: 188, 214, 215	
	Ch. 9: 302, 316	
	V_SU_CLK	
	Ch. 2: 70	
	Ch. 9: 288, 289	
	V_SU_CLK_DLY	
	Ch. 5: 158, 205	

V_SYNC_SRC	Ch. 5: 177, 183, 202	Ch. 9: 306
Ch. 6: 228, 229, 261	V_UP_SCRM_MD	V_WAK_EN
V_S_TX	Ch. 5: 177, 178, 182, 183, 202	Ch. 7: 278
Ch. 5: 207, 208	V_UP_SCRM_RX_OFF	Ch. 9: 303, 318
V_THRES_RX	Ch. 5: 178, 182, 183, 203	V_WAK_IRQ
Ch. 4: 134 , 141–144	V_UP_SCRM_TX_OFF	Ch. 9: 302, 303, 316
V_THRES_TX	Ch. 5: 177, 182, 183, 203	V_WAK_IRQMSK
Ch. 4: 134 , 141–144	V_UP_SEL	Ch. 9: 302, 303, 309
V_TI_IRQ	Ch. 5: 202	V_WAK_STA
Ch. 9: 302, 303, 316	V_UP_S_RX	Ch. 9: 303, 318
V_TI_IRQMSK	Ch. 5: 176, 213	V_WD_EN
Ch. 9: 302, 303, 309	V_UP_S_TX	Ch. 8: 284
V_UNIDIR_MD	Ch. 5: 176, 200 , 204	V_WD_RES
Ch. 4: 128, 135	V_UP_VIO	Ch. 8: 284
V_UNIDIR_RX	Ch. 5: 202	Ch. 9: 305
Ch. 4: 128, 135	V_USAGE	V_WD_TS
V_UP_DC_OFF	Ch. 4: 141	Ch. 9: 305, 312
Ch. 5: 177, 202	V_WAIT_PROC	V_Z1
V_UP_DC_STR	Ch. 9: 306	Ch. 4: 138
Ch. 5: 177, 202	V_WAIT_REG	V_Z2
V_UP_RPT_PAT		Ch. 4: 138

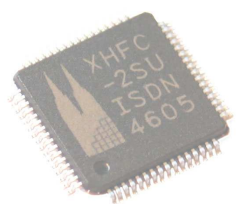
Index of pin names

Index entries are sorted by name. Pages of the pin list are printed in bold type.

<p>/CS Ch. 1: 34, 36 Ch. 2: 43, 44, 46, 47, 49, 52, 54</p> <p>/INT Ch. 1: 34, 35, 36 Ch. 2: 43, 57 Ch. 4: 152 Ch. 9: 299, 309–311, 314</p> <p>/IOR Ch. 1: 34, 39 Ch. 2: 43, 44, 47</p> <p>/IOW Ch. 1: 34, 39 Ch. 2: 43, 44, 47</p> <p>/RES Ch. 1: 34, 35, 36 Ch. 2: 43, 57 Ch. 9: 298</p> <p>/SPISEL Ch. 1: 33, 35, 36 Ch. 2: 57, 59–66</p> <p>/WAIT Ch. 1: 34, 39 Ch. 2: 43, 46, 49 Ch. 9: 306</p> <p>/WD Ch. 1: 34, 35, 37 Ch. 9: 305</p> <p>A0 Ch. 1: 34, 36 Ch. 2: 43, 44, 46–51, 53</p> <p>ADJ_0 Ch. 1: 34, 35, 39 Ch. 5: 168</p>	<p>ADJ_1 Ch. 1: 34, 35, 38 Ch. 5: 168</p> <p>ALE Ch. 1: 34, 39 Ch. 2: 43, 44, 47, 51–54</p> <p>C2IO Ch. 1: 34, 35, 37 Ch. 4: 128 Ch. 6: 223, 230, 261</p> <p>C4IO Ch. 1: 34, 35, 37 Ch. 4: 128 Ch. 6: 220, 223, 224, 226, 228, 230, 232, 237–239, 242, 245, 254, 259, 264, 265</p> <p>CLK_OUT Ch. 10: 328, 335 Ch. 1: 34, 35, 37 Ch. 9: 288, 289, 291, 292, 308</p> <p>D0 Ch. 1: 34, 36 Ch. 2: 43, 44, 46, 49</p> <p>D1 Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D2 Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D3 Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D4</p>	<p>Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D5 Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D6 Ch. 1: 34, 36 Ch. 2: 46, 49</p> <p>D7 Ch. 1: 34, 36 Ch. 2: 43, 44, 46, 49</p> <p>DN0 Ch. 1: 35, 36 Ch. 2: 57</p> <p>DN1 Ch. 1: 35, 36 Ch. 2: 57</p> <p>DN2 Ch. 1: 35, 36 Ch. 2: 57</p> <p>DN3 Ch. 1: 35, 36 Ch. 2: 57</p> <p>F0IO Ch. 1: 34, 35, 37 Ch. 4: 126, 128 Ch. 5: 166, 182, 187, 189, 190, 215 Ch. 6: 220, 223, 224, 226, 228, 230, 232, 235, 237–240, 242, 245, 254, 258, 271 Ch. 9: 302, 303</p> <p>F1_0 Ch. 10: 328, 335</p>
--	--	---

Ch. 1: 34, 35, 37	Ch. 1: 34, 35, 38	Ch. 9: 298
Ch. 6: 240, 255, 264	GPIO2	MODE1
FI_1	Ch. 10: 328, 329, 333–336, 338	Ch. 1: 34, 35, 37
Ch. 10: 328, 335	Ch. 1: 34, 35, 37	Ch. 2: 42–44, 57
Ch. 1: 34, 35, 37	GPIO3	Ch. 9: 298
Ch. 6: 240, 256, 264, 265	Ch. 10: 328, 329, 333–336, 338	NC
Ch. 9: 288, 289, 308	Ch. 1: 34, 35, 37	Ch. 1: 35, 39
GND	GPIO4	OSC_IN
Ch. 1: 34, 35, 36–39	Ch. 10: 328, 329, 333–336, 338	Ch. 1: 34, 35, 37
GPIO0	Ch. 1: 34, 35, 37	Ch. 9: 289, 291–293, 308
Ch. 10: 328, 329, 333–336, 338	GPIO5	OSC_OUT
Ch. 1: 34, 35, 37	Ch. 10: 328, 329, 333–336, 338	Ch. 1: 34, 35, 37
GPIO1	Ch. 1: 34, 35, 37	Ch. 9: 288, 289, 291
Ch. 10: 328, 329, 333–336, 338	GPIO6	PWM0
Ch. 1: 34, 35, 37	Ch. 10: 328, 333–335, 338	Ch. 10: 328, 329, 335
GPIO10	Ch. 1: 34, 35, 36	Ch. 1: 34, 35, 37
Ch. 10: 328, 330, 331, 337	GPIO7	Ch. 7: 276–278
Ch. 1: 34, 35, 39	Ch. 10: 328, 333–335, 338	PWM1
GPIO11	Ch. 1: 34, 35, 37	Ch. 10: 328, 335
Ch. 10: 326, 328, 330, 331, 337	GPIO8	Ch. 1: 34, 35, 37
Ch. 1: 34, 35, 39	Ch. 10: 326–328, 330, 331, 337	Ch. 7: 276–278
GPIO12	Ch. 1: 34, 35, 39	R_A0
Ch. 10: 328, 330, 331, 337	GPIO9	Ch. 10: 327, 328
Ch. 1: 34, 35, 38	Ch. 10: 327, 328, 330, 331, 337	Ch. 1: 34, 35, 39
GPIO13	Ch. 1: 34, 35, 39	Ch. 5: 157, 166–168, 182, 183
Ch. 10: 328, 330, 331, 337	L_A0	R_A1
Ch. 1: 34, 35, 38	Ch. 10: 327, 328	Ch. 10: 328
GPIO14	Ch. 1: 34, 35, 39	Ch. 1: 34, 35, 38
Ch. 10: 328, 330, 331, 337	L_A1	Ch. 5: 166, 182
Ch. 1: 34, 35, 38	Ch. 10: 328	R_B0
GPIO15	Ch. 1: 34, 35, 39	Ch. 10: 328
Ch. 10: 328, 330, 331, 337	Ch. 5: 168	Ch. 1: 34, 35, 39
Ch. 1: 34, 35, 38	L_B0	Ch. 5: 157, 166–168, 182, 183
GPIO16	Ch. 10: 327, 328	R_B1
Ch. 10: 326–328, 331, 332, 337	Ch. 1: 34, 35, 38	Ch. 10: 328
Ch. 1: 34, 35, 38	L_B1	Ch. 1: 34, 35, 38
GPIO17	Ch. 10: 328	Ch. 5: 166, 182
Ch. 10: 326, 328, 331, 332, 337	Ch. 1: 34, 35, 39	SPI_CLK
Ch. 1: 34, 35, 38	Ch. 5: 168	Ch. 1: 35, 39
GPIO18	MODE0	Ch. 2: 57, 65, 66
Ch. 10: 328, 331, 332, 337	Ch. 1: 34, 35, 37	SPI_INV
Ch. 1: 34, 35, 38	Ch. 2: 42–44, 57	Ch. 1: 35, 39
GPIO19	Ch. 1: 34, 35, 37	Ch. 2: 57, 65, 66
Ch. 10: 328, 331, 332, 337	Ch. 2: 42–44, 57	SPI_RX

<p>Ch. 1: 35, 36 Ch. 2: 57, 59, 60, 62–64</p> <p>SPI_TX Ch. 1: 35, 36 Ch. 2: 57, 59–64</p> <p>STIO1 Ch. 10: 328, 335 Ch. 1: 34, 35, 37 Ch. 3: 89–92, 96, 97, 105, 116, 122 Ch. 6: 220–224, 226, 237, 242, 243, 245, 246, 248, 258, 259, 269, 274</p> <p>STIO2 Ch. 10: 328, 335 Ch. 1: 34, 35, 37 Ch. 3: 90–92, 96, 97, 105, 117, 123 Ch. 6: 220–224, 226, 237, 242, 243, 245, 246, 248, 259, 269, 274</p>	<p>SYNC_I Ch. 1: 34, 35, 36 Ch. 6: 228, 231–234, 237, 258, 261, 262, 266 Ch. 8: 285</p> <p>SYNC_O Ch. 10: 328, 335 Ch. 1: 34, 35, 36 Ch. 6: 230–234, 261</p> <p>T_A0 Ch. 10: 327, 328 Ch. 1: 34, 35, 38 Ch. 5: 157, 167, 169, 183</p> <p>T_A1 Ch. 10: 328 Ch. 1: 34, 35, 38</p> <p>T_B0 Ch. 10: 328 Ch. 1: 34, 35, 38 Ch. 5: 157, 167, 169, 183</p>	<p>T_B1 Ch. 10: 328 Ch. 1: 34, 35, 38</p> <p>VDD Ch. 10: 327–329 Ch. 1: 34, 35, 36–39, 40</p> <p>VDD_SU0 Ch. 10: 326–328 Ch. 1: 34, 35, 38 Ch. 5: 169</p> <p>VDD_SU1 Ch. 10: 326, 328 Ch. 1: 34, 35, 38 Ch. 5: 169</p> <p>WAKEUP Ch. 1: 34, 35, 37 Ch. 2: 70 Ch. 7: 278 Ch. 9: 288, 306, 316, 318</p>
--	---	--



Cologne Chip AG
Data Sheet of XHFC-2SU

