

# XHFC - 2SU

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



## Revision History of XHFC-2SU Data Sheet

Date	Remarks
October 2007	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.
July 2007	C1 and C2 in external S/T receive circuitry moved from L_A / L_B pins to R_A / R_B pins, $t_{AHD}$ 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.
March 2007	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.
March 2006	Clarifications and corrections made on interrupt output (Figure 9.4), PWM description and A_CON_HDLC register description.
February 2006	Corrections are made on PLL frequency range and setup examples. External $U_p$ circuitry has been improved. Power consumption information has been added.
December 2005	First edition of this data sheet.



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](mailto:support@CologneChip.com)

Copyright 1994 - 2007 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.

# Contents

<b>About this data sheet and Cologne Chip technical support</b>	<b>23</b>
<b>1 General description</b>	<b>27</b>
1.1 System overview . . . . .	28
1.2 Features . . . . .	29
1.3 Pin description . . . . .	30
1.3.1 Pinout diagram . . . . .	30
1.3.2 Pin list . . . . .	32
<b>2 Microprocessor bus interface</b>	<b>37</b>
2.1 Mode selection . . . . .	38
2.2 Parallel processor interface . . . . .	39
2.2.1 Overview . . . . .	39
2.2.2 Interface signals . . . . .	39
2.2.3 Register access . . . . .	39
2.2.3.1 Non-multiplexed / multiplexed access . . . . .	39
2.2.3.2 Read* access . . . . .	40
2.2.3.3 Register address read-back capability . . . . .	40
2.2.3.4 Problems with interrupts between address and data accesses . . . . .	41
2.2.4 Signal and timing characteristics . . . . .	41
2.2.4.1 Bus interface in mode 2 and mode 3 (non-multiplexed) . . . . .	42
2.2.4.2 Bus interface in mode 2m and mode 3m (multiplexed) . . . . .	45
2.2.5 Microprocessor connection circuitries . . . . .	48
2.3 Serial processor interface (SPI) . . . . .	50
2.3.1 SPI control byte . . . . .	50
2.3.2 SPI transactions . . . . .	51
2.3.3 Transaction duration . . . . .	52
2.3.4 Register write access . . . . .	54
2.3.5 Register read access . . . . .	56
2.3.6 Register access duration . . . . .	57
2.3.7 Register address read-back capability . . . . .	57

2.3.8	Problems with interrupts during transaction sequences . . . . .	58
2.3.9	SPI timing diagrams . . . . .	58
2.3.10	SPI connection circuitry . . . . .	60
2.4	Auto-EEPROM mode . . . . .	61
2.5	Register description . . . . .	62
2.5.1	Write only registers . . . . .	62
2.5.2	Read only registers . . . . .	64
2.5.3	Read / write register . . . . .	66
<b>3</b>	<b>XHFC-2SU data flow</b>	<b>67</b>
3.1	Data flow concept . . . . .	68
3.1.1	Overview . . . . .	68
3.1.2	Term definitions . . . . .	68
3.2	Flow controller . . . . .	69
3.2.1	Overview . . . . .	69
3.2.2	Switching buffers . . . . .	70
3.2.3	Timed sequence . . . . .	70
3.2.4	Transmit operation (FIFO in transmit data direction) . . . . .	71
3.2.5	Receive operation (FIFO in receive data direction) . . . . .	71
3.2.6	Connection summary . . . . .	72
3.3	Assigners . . . . .	74
3.3.1	HFC-channel assigner . . . . .	74
3.3.2	PCM slot assigner . . . . .	74
3.3.3	Line interface assigner . . . . .	74
3.3.4	Assigner summary . . . . .	75
3.4	Data flow modes . . . . .	77
3.4.1	Simple Mode (SM) . . . . .	77
3.4.1.1	Mode description . . . . .	77
3.4.1.2	Subchannel processing . . . . .	79
3.4.1.3	Example for SM . . . . .	79
3.4.2	Channel Select Mode (CSM) . . . . .	85
3.4.2.1	Mode description . . . . .	85
3.4.2.2	HFC-channel assigner . . . . .	85
3.4.2.3	Subchannel Processing . . . . .	85
3.4.2.4	Example for CSM . . . . .	86
3.4.3	FIFO Sequence Mode (FSM) . . . . .	91
3.4.3.1	Mode description . . . . .	91
3.4.3.2	FIFO sequence . . . . .	91
3.4.3.3	FSM programming . . . . .	93

3.4.3.4	Example for FSM . . . . .	94
3.5	Subchannel Processing . . . . .	100
3.5.1	Overview . . . . .	100
3.5.2	Subchannel registers . . . . .	100
3.5.3	Details of the FIFO oriented part of the subchannel processor (part A) . . . . .	101
3.5.3.1	FIFO transmit operation in transparent mode . . . . .	101
3.5.3.2	FIFO transmit operation in HDLC mode . . . . .	103
3.5.3.3	FIFO receive operation in transparent mode . . . . .	103
3.5.3.4	FIFO receive operation in HDLC mode . . . . .	103
3.5.4	Details of the HFC-channel oriented part of the subchannel processor (part B) . . . . .	103
3.5.4.1	FIFO transmit operation in SM . . . . .	103
3.5.4.2	FIFO transmit operation in CSM and FSM . . . . .	104
3.5.4.3	FIFO receive operation in SM . . . . .	104
3.5.4.4	FIFO receive operation in CSM and FSM . . . . .	104
3.5.5	Subchannel example for SM . . . . .	105
3.5.6	Subchannel example for CSM . . . . .	109
<b>4</b>	<b>FIFO handling and HDLC controller</b>	<b>117</b>
4.1	Overview . . . . .	118
4.2	FIFO counters . . . . .	118
4.3	FIFO size setup . . . . .	119
4.4	FIFO operation . . . . .	120
4.4.1	HDLC transmit FIFOs . . . . .	120
4.4.2	Automatic D-channel frame repetition (for S/T in TE mode only) . . . . .	121
4.4.3	HDLC receive FIFOs . . . . .	121
4.4.4	Transparent mode of XHFC-2SU . . . . .	122
4.5	Register description . . . . .	124
4.5.1	Write only registers . . . . .	124
4.5.2	Read only registers . . . . .	128
4.5.3	Read / write registers . . . . .	135
<b>5</b>	<b>Universal ISDN Port</b>	<b>145</b>
5.1	General overview of the S/T and U <sub>p</sub> interfaces . . . . .	146
5.1.1	Array registers and multiregisters . . . . .	146
5.1.2	Block diagram of the Universal ISDN Port module . . . . .	146
5.2	S/T interface description . . . . .	148
5.2.1	Overview . . . . .	148
5.2.2	Frame Structure . . . . .	148
5.2.3	Multiframe structure . . . . .	149

5.2.4	Data transmission . . . . .	149
5.2.5	INFO signals . . . . .	151
5.2.6	State machine . . . . .	153
5.2.7	Clock synchronization . . . . .	156
5.2.8	External circuitries . . . . .	158
5.2.8.1	External receive circuitry . . . . .	158
5.2.8.2	External transmit circuitry . . . . .	159
5.2.8.3	Transformer and ISDN jack connection . . . . .	160
5.2.9	S/T transformers . . . . .	161
5.3	U <sub>p</sub> interface description . . . . .	165
5.3.1	Overview . . . . .	165
5.3.2	Frame Structure . . . . .	165
5.3.3	Superframe structure . . . . .	165
5.3.4	Data transmission . . . . .	166
5.3.5	INFO signals . . . . .	167
5.3.6	State machine . . . . .	169
5.3.7	Clock synchronization . . . . .	172
5.3.8	External circuitry . . . . .	174
5.3.9	U <sub>p</sub> transformers . . . . .	175
5.4	Common features of the S/T and U <sub>p</sub> interfaces . . . . .	176
5.4.1	Direct data access for test purposes . . . . .	176
5.4.2	Clock synchr. with several TEs connected to different central office switches . . . . .	177
5.4.3	Combined S/T and U <sub>p</sub> circuitry . . . . .	182
5.5	Register description . . . . .	183
5.5.1	Write only registers . . . . .	183
5.5.2	Read only registers . . . . .	199
<b>6</b>	<b>PCM interface</b>	<b>209</b>
6.1	PCM interface function . . . . .	210
6.2	PCM data flow . . . . .	211
6.3	PCM initialization . . . . .	213
6.4	PCM timing . . . . .	213
6.4.1	Mode selection . . . . .	213
6.4.2	Master mode . . . . .	213
6.4.3	Slave mode . . . . .	213
6.5	PCM clock synchronization . . . . .	218
6.5.1	Overview . . . . .	218
6.5.2	Manual or automatic synchronization source selection . . . . .	218
6.5.3	PLL programming for F0IO generation . . . . .	218

6.5.4	Manual PLL adjustment . . . . .	218
6.5.5	C2IO signal . . . . .	220
6.5.6	Application examples for XHFC-2SU synchronization schemes . . . . .	220
6.5.6.1	System with internal PCM bus expanded by ISDN interfaces . . . . .	220
6.5.6.2	Application with multiple XHFC-2SU . . . . .	222
6.6	Multiframe / superframe synchronization to the PCM interface . . . . .	224
6.6.1	Overview . . . . .	224
6.6.2	Frame counter . . . . .	224
6.6.3	Synchronization input signal . . . . .	224
6.6.4	Synchronization output signal (F0IO pulse modification) . . . . .	227
6.6.5	Line interface synchronization . . . . .	228
6.7	External CODECs . . . . .	229
6.8	GCI / IOM-2 mode . . . . .	231
6.8.1	Overview . . . . .	231
6.8.2	GCI frame structure . . . . .	231
6.8.3	GCI register programming . . . . .	232
6.8.3.1	Enable CGI functionality . . . . .	232
6.8.3.2	Monitor channel . . . . .	233
6.8.3.3	Command / indication bits (C/I-channel) . . . . .	233
6.8.3.4	Examples for GCI frame embedding in the PCM data structure . . . . .	233
6.8.4	GCI protocol . . . . .	238
6.8.4.1	XHFC-2SUtransmit procedure . . . . .	238
6.8.4.2	XHFC-2SUreceive procedure . . . . .	239
6.8.4.3	Receiver abort . . . . .	241
6.9	Register description . . . . .	242
6.9.1	Write only registers . . . . .	242
6.9.2	Read only registers . . . . .	259
6.9.3	Read / write register . . . . .	262
<b>7</b>	<b>Pulse width modulation (PWM) outputs</b>	<b>263</b>
7.1	Overview . . . . .	264
7.2	Standard PWM usage . . . . .	264
7.3	Alternative PWM usage . . . . .	264
7.4	Register description . . . . .	265
<b>8</b>	<b>Bit Error Rate Test (BERT)</b>	<b>267</b>
8.1	BERT functionality . . . . .	268
8.2	BERT transmitter . . . . .	268
8.3	BERT receiver . . . . .	268

8.4	Register description . . . . .	272
8.4.1	Write only registers . . . . .	272
8.4.2	Read only registers . . . . .	273
<b>9</b>	<b>Clock, PLL, reset, interrupt, timer and watchdog</b>	<b>275</b>
9.1	Clock . . . . .	276
9.1.1	Clock output . . . . .	276
9.1.2	Clock distribution . . . . .	276
9.1.3	Clock oscillator circuitry . . . . .	278
9.1.3.1	Frequency accuracy . . . . .	278
9.1.3.2	Pierce oscillator . . . . .	278
9.1.3.3	3rd overtone oscillator . . . . .	279
9.1.3.4	Crystal oscillator circuitry . . . . .	279
9.2	Phase locked loop (PLL) . . . . .	280
9.2.1	Overview . . . . .	280
9.2.2	PLL structure . . . . .	280
9.2.3	PLL operation . . . . .	281
9.2.4	PLL configuration . . . . .	281
9.3	Reset . . . . .	285
9.4	Interrupt . . . . .	286
9.4.1	Common features . . . . .	286
9.4.2	ST/U <sub>p</sub> interface interrupt . . . . .	286
9.4.3	FIFO interrupt . . . . .	286
9.4.4	Miscellaneous interrupts . . . . .	289
9.4.4.1	Line interface frequency slip interrupt . . . . .	289
9.4.4.2	Timer interrupt . . . . .	290
9.4.4.3	Processing / non-processing interrupt . . . . .	290
9.4.4.4	Command / indication interrupt (GCI interface) . . . . .	290
9.4.4.5	Wakeup interrupt . . . . .	290
9.4.4.6	Interrupt for GCI monitor byte transmission . . . . .	291
9.4.4.7	Interrupt after GCI monitor byte received . . . . .	291
9.5	Watchdog reset . . . . .	292
9.6	Register description . . . . .	293
9.6.1	Write only registers . . . . .	293
9.6.2	Read only registers . . . . .	300
9.6.3	Read / write registers . . . . .	309
<b>10</b>	<b>General purpose I/O pins (GPIO)</b>	<b>311</b>
10.1	GPIO functionality . . . . .	312

---

10.2 GPIO output voltage . . . . .	312
10.3 Register description . . . . .	315
10.3.1 Write only registers . . . . .	315
10.3.2 Read only registers . . . . .	320
<b>11 Electrical characteristics</b>	<b>323</b>
<b>12 XHFC-2SU package dimensions</b>	<b>327</b>
<b>References</b>	<b>329</b>
<b>List of register and bitmap abbreviations</b>	<b>331</b>



# List of Figures

1.1	XHFC-2SU block diagram . . . . .	27
1.2	XHFC-2SU pinout in parallel processor interface mode . . . . .	30
1.3	XHFC-2SU pinout in serial processor interface mode . . . . .	31
2.1	Bus interface read access in mode 2 (Motorola) and mode 3 (Intel) . . . . .	43
2.2	Bus interface write access in mode 2 (Motorola) and mode 3 (Intel) . . . . .	44
2.3	Bus interface read access in modes 2m and 3m (multiplexed) . . . . .	46
2.4	Bus interface write access in mode 2m and 3m (multiplexed) . . . . .	47
2.5	8 bit Motorola processor circuitry example (mode 2) . . . . .	48
2.6	8 bit Motorola processor circuitry example (mode 2m) . . . . .	48
2.7	8 bit Intel processor circuitry example (mode 3) . . . . .	49
2.8	8 bit Intel processor circuitry example (mode 3m) . . . . .	49
2.9	16 bit SPI write transaction (R = '0', one data byte) . . . . .	52
2.10	16 bit SPI read transaction (R = '1', one data byte) . . . . .	52
2.11	40 bit SPI write transaction (R = '0', four data bytes) . . . . .	53
2.12	40 bit SPI read transaction (R = '1', four data bytes) . . . . .	53
2.13	Split 40 bit SPI read transaction (R = '1') . . . . .	53
2.14	Register write access (transaction sequence) . . . . .	55
2.15	Multiple write accesses to the same register . . . . .	55
2.16	Register write access with a 40 bit transaction . . . . .	55
2.17	Register read access (transaction sequence) . . . . .	56
2.18	Multiple read accesses to the same register . . . . .	57
2.19	Register read access with a 40 bit transaction . . . . .	57
2.20	SPI timing diagram for data write transactions . . . . .	58
2.21	SPI timing diagram for data read transactions . . . . .	59
2.22	SPI connection circuitry . . . . .	60
3.1	Data flow block diagram . . . . .	67
3.2	Areas of FIFO oriented, HFC-channel oriented and PCM time slot oriented numbering . . . . .	69
3.3	The flow controller in transmit operation . . . . .	71

3.4	The flow controller in receive FIFO operation . . . . .	72
3.5	Overview of the assigner programming . . . . .	76
3.6	SM example . . . . .	80
3.7	HFC-channel assigner in CSM . . . . .	85
3.8	CSM example . . . . .	86
3.9	HFC-channel assigner in FSM . . . . .	91
3.10	FSM list processing . . . . .	92
3.11	FSM list programming . . . . .	93
3.12	FSM example . . . . .	94
3.13	General structure of the subchannel processor . . . . .	100
3.14	Location of the subchannel parts A and B in the data flow diagram . . . . .	101
3.15	Part A of the subchannel processor . . . . .	102
3.16	Part B of the subchannel processor . . . . .	104
3.17	SM example with subchannel processor . . . . .	105
3.18	CSM example with subchannel processor . . . . .	110
4.1	FIFO organization . . . . .	120
4.2	FIFO data organization in HDLC mode . . . . .	121
5.1	Overview of the Universal ISDN Port module . . . . .	147
5.2	Frame structure at reference point S and T . . . . .	148
5.3	S/T frame composition for B1-, B2-, D-channel and multiframe bits . . . . .	151
5.4	S/T frame decomposition for B1-, B2-, D-, E-channel and multiframe bits . . . . .	152
5.5	S/T clock synchronization . . . . .	157
5.6	External S/T receive circuitry for TE and NT mode . . . . .	158
5.7	External S/T transmit circuitry for TE and NT mode . . . . .	159
5.8	Transformer and connector circuitry in TE mode . . . . .	160
5.9	Transformer and connector circuitry in NT mode . . . . .	160
5.10	U <sub>p</sub> interface frame structure . . . . .	165
5.11	U <sub>p</sub> frame composition for B1-, B2-, D-channel and superframe bits . . . . .	167
5.12	U <sub>p</sub> frame decomposition for B1-, B2-, D-channel and superframe bits . . . . .	168
5.13	U <sub>p</sub> clock synchronization . . . . .	173
5.14	External U <sub>p</sub> circuitry for TE and LT mode . . . . .	174
5.15	Synchronization scenario with TEs connected to several central office switches . . . . .	177
5.16	Synchronization registers (detail of Figure 5.15) . . . . .	178
5.17	Timing example of one transmit and one receive transmission . . . . .	179
5.18	Data transmission with $f_{FSC\_1} > f_{FSC\_0}$ (i.e. too fast FSC from unsynchronized TE) . . . . .	180
5.19	Data transmission with $f_{FSC\_1} < f_{FSC\_0}$ (i.e. too slow FSC from unsynchronized TE) . . . . .	181

6.1	PCM data flow for transmit and receive time slots . . . . .	211
6.2	Global setting for all PCM time slots (detail of Figure 6.1) . . . . .	212
6.3	PCM timing for master mode . . . . .	214
6.4	PCM timing for slave mode . . . . .	216
6.5	PCM clock synchronization . . . . .	219
6.6	Expanding an existing system with ISDN ports (XHFC-2SU as PCM slave) . . . . .	221
6.7	Expanding an existing system with ISDN ports (XHFC-2SU as PCM master) . . . . .	221
6.8	Multiple XHFC-2SU synchronized with an open loop of SYNC_O / SYNC_I . . . . .	222
6.9	Multiple XHFC-2SU synchronized with a closed loop of SYNC_O / SYNC_I . . . . .	223
6.10	Multiframe / superframe synchronization to the PCM interface . . . . .	225
6.11	Timing specification of the SYNC_IN change detection . . . . .	226
6.12	F0IO pulse modification . . . . .	227
6.13	Example for two CODEC enable signal shapes . . . . .	229
6.14	Single channel GCI format . . . . .	232
6.15	GCI application example 1 (XHFC-2SU is PCM master and GCI master) . . . . .	234
6.16	GCI application example 2 (XHFC-2SU is PCM slave and GCI master) . . . . .	236
6.17	GCI application example 3 (XHFC-2SU is PCM slave and GCI slave) . . . . .	236
6.18	GCI protocol for monitor bytes transmission . . . . .	238
6.19	GCI protocol for monitor bytes receiving . . . . .	240
8.1	BERT transmitter block diagram . . . . .	269
8.2	BERT receiver block diagram . . . . .	270
9.1	CLK_OUT generation and clock distribution . . . . .	277
9.2	Standard XHFC-2SU quartz circuitry . . . . .	278
9.3	PLL block diagram . . . . .	281
9.4	Interrupt output . . . . .	286
9.5	ST/U <sub>p</sub> interface interrupt . . . . .	287
9.6	Enable FIFO interrupt condition with V_FIFO_IRQ_EN . . . . .	287
9.7	FIFO interrupt . . . . .	288
9.8	Miscellaneous interrupts . . . . .	289
9.9	Wakeup interrupt . . . . .	290
10.1	GPIO block diagram . . . . .	313
12.1	XHFC-2SU package dimensions . . . . .	328



# List of Tables

2.1	Overview of the XHFC-2SU bus interface registers . . . . .	37
2.2	Microprocessor access types . . . . .	38
2.3	Parallel processor interface mode selection . . . . .	39
2.4	Pins and signal names of the parallel processor interface modes . . . . .	40
2.5	Overview of accesses in parallel microprocessor interface mode . . . . .	41
2.6	Timing diagrams of the parallel microprocessor interface . . . . .	42
2.7	Symbols of read accesses in Figure 2.1 . . . . .	43
2.8	Symbols of write accesses in Figure 2.2 . . . . .	44
2.9	Symbols of read accesses in Figure 2.3 . . . . .	46
2.10	Symbols of write accesses in Figures 2.4 . . . . .	47
2.11	Overview of the SPI interface pins . . . . .	50
2.12	SPI control byte with A = '0' . . . . .	51
2.13	SPI control byte with A = '1' . . . . .	51
2.14	SPI control commands . . . . .	51
2.15	SPI data rates . . . . .	57
2.16	Symbols of write access in Figure 2.20 . . . . .	59
2.17	Symbols of read access in Figure 2.21 . . . . .	59
3.1	Flow controller connectivity . . . . .	73
3.2	V_DATA_FLOW programming values for bidirectional connections . . . . .	73
3.3	Line interface assigner . . . . .	75
3.4	Index registers of the FIFO array registers (sorted by address) . . . . .	78
3.5	List specification of the example in Figure 3.12 . . . . .	95
3.6	Subchannel processing according to Figure 3.17 (SM ❶ TX, transparent mode) . . .	107
3.7	Subchannel processing according to Figure 3.17 (SM ❶ RX, transparent mode) . . .	107
3.8	Subchannel processing according to Figure 3.17 (SM ❷ TX, HDLC mode) . . . . .	109
3.9	Subchannel processing according to Figure 3.17 (SM ❷ RX, HDLC mode) . . . . .	110
3.10	Subchannel processing according to Figure 3.18 (CSM ❶ TX, transparent mode) . .	112
3.11	Subchannel processing according to Figure 3.18 (CSM ❶ RX, transparent mode) . .	113
3.12	Subchannel processing according to Figure 3.18 (CSM ❷ TX, HDLC mode) . . . . .	116
3.13	Subchannel processing according to Figure 3.18 (CSM ❷ RX, HDLC mode) . . . . .	116

4.1	Overview of the XHFC-2SU FIFO registers . . . . .	117
4.2	FIFO size setup . . . . .	119
5.1	Overview of the ST/U <sub>p</sub> interface registers . . . . .	145
5.2	Legend for Figure 5.2 . . . . .	149
5.3	Multiframe structure of the Q- and S-bits . . . . .	150
5.4	S/T interface activation / deactivation layer 1 matrix for NT mode . . . . .	154
5.5	S/T interface activation / deactivation layer 1 matrix for TE mode . . . . .	155
5.6	S/T transformer part numbers and manufacturers . . . . .	161
5.7	Legend for Figure 5.10 . . . . .	166
5.8	Superframe construction . . . . .	166
5.9	U <sub>p</sub> interface activation / deactivation layer 1 matrix for LT mode . . . . .	170
5.10	U <sub>p</sub> interface activation / deactivation layer 1 matrix for TE mode . . . . .	171
5.11	U <sub>p</sub> transformer part numbers and manufacturers . . . . .	175
5.12	Symbols of Figures 5.17 . . . . .	178
6.1	Overview of the XHFC-2SU PCM interface registers . . . . .	209
6.2	PCM master mode . . . . .	210
6.3	PCM data flow programming . . . . .	212
6.4	Symbols of PCM timing for master mode in Figure 6.3 . . . . .	215
6.5	Symbols of PCM timing for slave mode in Figure 6.4 . . . . .	217
6.6	Symbols of the SYNC_IN change detection timing specification in Figure 6.11 . . . . .	227
6.7	Selection of the F0IO pulse characteristic (see also Figure 6.12) . . . . .	228
6.8	Legend of figure 6.14 . . . . .	232
6.9	Programming procedure for application example 1 according to Figure 6.15 . . . . .	235
6.10	Programming procedure for application example 3 according to Figure 6.17 . . . . .	237
6.11	Rules for the handshake signals MX <sub>TX</sub> and MR <sub>TX</sub> . . . . .	239
6.12	Rules for the handshake signals MX <sub>RX</sub> and MR <sub>RX</sub> . . . . .	241
7.1	Overview of the XHFC-2SU PWM registers . . . . .	263
8.1	Overview of the XHFC-2SU BERT registers . . . . .	267
9.1	Overview of clock, PLL, reset, timer and watchdog registers . . . . .	275
9.2	PLL setup examples (P, M, N, S) with ISDN related frequencies for f <sub>out</sub> . . . . .	283
9.3	PLL setup examples (P, M, N, S) with ISDN related frequencies for f <sub>in</sub> . . . . .	284
9.4	XHFC-2SU reset groups . . . . .	285
10.1	Overview of the XHFC-2SU general purpose I/O registers . . . . .	311
10.2	GPIO pins . . . . .	314

# 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<sub>p</sub> 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

 **Please note !**

See explanation of register types on page 17.

### Write only registers:

Address	Name	Reset group	Page
0x3C	A_B1_TX[ST/Up]	0, 1, 3	196
0x3D	A_B2_TX[ST/Up]	0, 1, 3	196
0x3F	A_BAC_S_TX[ST/Up]	0, 1, 3	198
0x3E	A_D_TX[ST/Up]	0, 1, 3	197
0x0E	A_INC_RES_FIFO[FIFO]	0, 1, 2, 3, H	126
0x36	A_MS_DF[ST/Up]		194
0x34	A_MS_TX[ST/Up]	0, 1, 3	190
0x35	A_ST_CTRL3[ST/Up]	0, 1, 3	191
0x37	A_SU_CLK_DLY[ST/Up]	-	195
0x31	A_SU_CTRL0[ST/Up]	0, 1, 3	185
0x32	A_SU_CTRL1[ST/Up]	0, 1, 3	187
0x33	A_SU_CTRL2[ST/Up]	0, 1, 3	188
0x30	A_SU_WR_STA[ST/Up]	0, 1, 3	184
0x35	A_UP_CTRL3[ST/Up]	0, 1, 3	192
0x1B	R_BERT_WD_MD	0, 1	272
0x28	R_CI_TX		255
0x00	R_CIRM	H	293
0x02	R_CLK_CFG	H	295
0x01	R_CTRL	H	62
0x0D	R_FIFO_MD	H	125
0x0C	R_FIFO_THRES		124
0x0F	R_FIFO	0, 1	127
0x0B	R_FIRST_FIFO	0, 1	124
0x0F	R_FSM_IDX	0, 1	127
0x29	R_GCI_CFG0		256
0x2A	R_GCI_CFG1		258
0x4A	R_GPIO_EN0	0	318
0x42	R_GPIO_EN1	0	316
0x47	R_GPIO_EN2	0	317
0x48	R_GPIO_OUT0	0	318
0x40	R_GPIO_OUT1	0	315
0x45	R_GPIO_OUT2	0	317
0x44	R_GPIO_SEL_BL	0	316
0x4C	R_GPIO_SEL		319
0x13	R_IRQ_CTRL	0	297
0x11	R_MISC_IRQMSK	H	296
0x2B	R_MON_TX		258
0x15	R_MSS0	0, 2	246
0x15	R_MSS1	0, 2	251
0x14	R_PCM_MD0	0, 2	243
0x15	R_PCM_MD1	0, 2	248
0x15	R_PCM_MD2	0, 2	249
0x50	R_PLL_CTRL		299

Address	Name	Reset group	Page
0x1E	R_PWM_CFG		265
0x46	R_PWM_MD	0	266
0x38	R_PWM0	0, 1, 3	265
0x39	R_PWM1	0, 1, 3	266
0x08	R_RAM_ADDR	0	63
0x09	R_RAM_CTRL	0	63
0x15	R_SH0H	0, 2	252
0x15	R_SH0L	0, 2	252
0x15	R_SH1H	0, 2	253
0x15	R_SH1L	0, 2	252
0x15	R_SL_SEL0	0, 2	244
0x15	R_SL_SEL1	0, 2	245
0x15	R_SL_SEL7	0, 2	245
0x10	R_SLOT	0, 2	242
0x12	R_SU_IRQMSK	3	296
0x16	R_SU_SEL	0, 3	183
0x17	R_SU_SYNC	0, 3	254
0x1A	R_TI_WD	0, 1	298

### Read only registers:

Address	Name	Reset group	Page
0x3C	A_B1_RX[ST/Up]	0, 3	205
0x3D	A_B2_RX[ST/Up]	0, 3	205
0x3E	A_D_RX[ST/Up]	0, 3	206
0x3F	A_E_RX[ST/Up]	0, 3	207
0x0C	A_F1[FIFO]	0, 1	128
0x0D	A_F2[FIFO]	0, 1	129
0x0E	A_FIFO_STA[FIFO]		130
0x34	A_MS_RX[ST/Up]	0, 3	203
0x32	A_SU_DLYH[ST/Up]	0, 3	202
0x31	A_SU_DLYL[ST/Up]	0, 3	201
0x30	A_SU_RD_STA[ST/Up]	0, 3	200
0x35	A_SU_STA[ST/Up]		204
0x14	A_USAGE[FIFO]	H	131
0x04	A_Z1[FIFO]	0, 1	128
0x06	A_Z2[FIFO]	0, 1	128
0x13	R_AF0_OVIEW		199
0x1B	R_BERT_ECH	0, 1	274
0x1A	R_BERT_ECL	0, 1	273
0x17	R_BERT_STA	0, 1	273
0x16	R_CHIP_ID	H	64

Address	Name	Reset group	Page
0x1F	R_CHIP_RV	–	64
0x28	R_CI_RX		260
0x19	R_F0_CNTH	0, 1	259
0x18	R_F0_CNTL	0, 1	259
0x20	R_FIFO_BL0_IRQ		305
0x21	R_FIFO_BL1_IRQ		306
0x22	R_FIFO_BL2_IRQ		307
0x23	R_FIFO_BL3_IRQ		308
0x24	R_FILL_BL0		131
0x25	R_FILL_BL1		132
0x26	R_FILL_BL2		133
0x27	R_FILL_BL3		134
0x29	R_GCI_STA		261
0x48	R_GPIO_IN0	–	321
0x40	R_GPIO_IN1	–	320
0x45	R_GPIO_IN2	–	320
0x88	R_INT_DATA	–	65
0x10	R_IRQ_OVIEW	0, 1	300
0x11	R_MISC_IRQ	0, 1	302
0x2A	R_MON_RX		261
0x50	R_PLL_STA		308
0x15	R_RAM_USE	0, 1	64
0x1D	R_SL_MAX	–	259
0x1C	R_STATUS	–	304
0x12	R_SU_IRQ	0, 1	303

### Read / Write registers:

Address	Name	Reset group	Page
0xF4	A_CH_MSK[FIFO]	0, 1	136
0xFC	A_CHANNEL[FIFO]	0, 1	140
0xFA	A_CON_HDLC[FIFO]	0, 1	137
0xFF	A_FIFO_CTRL[FIFO]	0, 1	142
0x84	A_FIFO_DATA_NOINC[FIFO]	–	135
0x80	A_FIFO_DATA[FIFO]	–	135
0xFD	A_FIFO_SEQ[FIFO]	0, 1	141
0xD0	A_SL_CFG[SLOT]	0, 3	262
0xFB	A_SUBCH_CFG[FIFO]	0, 1	139
0x52	R_PLL_N		309
0x51	R_PLL_P		309
0x53	R_PLL_S		309
0xC0	R_RAM_DATA	–	66

**Note:** Reset group 0 = soft reset, 1 = HFC reset, 2 = PCM reset, 3 = line interface reset, H = hardware reset. See Table 9.4 on page 285 for a detailed reset group explanation.

## Registers sorted by address



**Please note !**

See explanation of register types on page 17.

### Write only registers:

Address	Name	Reset group	Page
0x00	R_CIRM	H	293
0x01	R_CTRL	H	62
0x02	R_CLK_CFG	H	295
0x08	R_RAM_ADDR	0	63
0x09	R_RAM_CTRL	0	63
0x0B	R_FIRST_FIFO	0, 1	124
0x0C	R_FIFO_THRES		124
0x0D	R_FIFO_MD	H	125
0x0E	A_INC_RES_FIFO[FIFO]	0, 1, 2, 3, H	126
0x0F	R_FSM_IDX	0, 1	127
0x0F	R_FIFO	0, 1	127
0x10	R_SLOT	0, 2	242
0x11	R_MISC_IRQMSK	H	296
0x12	R_SU_IRQMSK	3	296
0x13	R_IRQ_CTRL	0	297
0x14	R_PCM_MD0	0, 2	243
0x15	R_MSS0	0, 2	246
0x15	R_MSS1	0, 2	251
0x15	R_PCM_MD1	0, 2	248
0x15	R_PCM_MD2	0, 2	249
0x15	R_SH0H	0, 2	252
0x15	R_SH1H	0, 2	253
0x15	R_SH0L	0, 2	252
0x15	R_SH1L	0, 2	252
0x15	R_SL_SEL0	0, 2	244
0x15	R_SL_SEL1	0, 2	245
0x15	R_SL_SEL7	0, 2	245
0x16	R_SU_SEL	0, 3	183
0x17	R_SU_SYNC	0, 3	254
0x1A	R_TI_WD	0, 1	298
0x1B	R_BERT_WD_MD	0, 1	272
0x1E	R_PWM_CFG		265
0x28	R_CI_TX		255
0x29	R_GCI_CFG0		256
0x2A	R_GCI_CFG1		258
0x2B	R_MON_TX		258
0x30	A_SU_WR_STA[ST/Up]	0, 1, 3	184
0x31	A_SU_CTRL0[ST/Up]	0, 1, 3	185
0x32	A_SU_CTRL1[ST/Up]	0, 1, 3	187
0x33	A_SU_CTRL2[ST/Up]	0, 1, 3	188
0x34	A_MS_TX[ST/Up]	0, 1, 3	190
0x35	A_ST_CTRL3[ST/Up]	0, 1, 3	191
0x35	A_UP_CTRL3[ST/Up]	0, 1, 3	192

Address	Name	Reset group	Page
0x36	A_MS_DF[ST/Up]		194
0x37	A_SU_CLK_DLY[ST/Up]	-	195
0x38	R_PWM0	0, 1, 3	265
0x39	R_PWM1	0, 1, 3	266
0x3C	A_B1_TX[ST/Up]	0, 1, 3	196
0x3D	A_B2_TX[ST/Up]	0, 1, 3	196
0x3E	A_D_TX[ST/Up]	0, 1, 3	197
0x3F	A_BAC_S_TX[ST/Up]	0, 1, 3	198
0x40	R_GPIO_OUT1	0	315
0x42	R_GPIO_EN1	0	316
0x44	R_GPIO_SEL_BL	0	316
0x45	R_GPIO_OUT2	0	317
0x46	R_PWM_MD	0	266
0x47	R_GPIO_EN2	0	317
0x48	R_GPIO_OUT0	0	318
0x4A	R_GPIO_EN0	0	318
0x4C	R_GPIO_SEL		319
0x50	R_PLL_CTRL		299

### Read only registers:

Address	Name	Reset group	Page
0x04	A_Z1[FIFO]	0, 1	128
0x06	A_Z2[FIFO]	0, 1	128
0x0C	A_F1[FIFO]	0, 1	128
0x0D	A_F2[FIFO]	0, 1	129
0x0E	A_FIFO_STA[FIFO]		130
0x10	R_IRQ_OVIEW	0, 1	300
0x11	R_MISC_IRQ	0, 1	302
0x12	R_SU_IRQ	0, 1	303
0x13	R_AF0_OVIEW		199
0x14	A_USAGE[FIFO]	H	131
0x15	R_RAM_USE	0, 1	64
0x16	R_CHIP_ID	H	64
0x17	R_BERT_STA	0, 1	273
0x18	R_F0_CNTRL	0, 1	259
0x19	R_F0_CNTH	0, 1	259
0x1A	R_BERT_ECL	0, 1	273
0x1B	R_BERT_ECH	0, 1	274
0x1C	R_STATUS	-	304
0x1D	R_SL_MAX	-	259

Address	Name	Reset group	Page
0x1F	R_CHIP_RV	–	64
0x20	R_FIFO_BL0_IRQ		305
0x21	R_FIFO_BL1_IRQ		306
0x22	R_FIFO_BL2_IRQ		307
0x23	R_FIFO_BL3_IRQ		308
0x24	R_FILL_BL0		131
0x25	R_FILL_BL1		132
0x26	R_FILL_BL2		133
0x27	R_FILL_BL3		134
0x28	R_CI_RX		260
0x29	R_GCI_STA		261
0x2A	R_MON_RX		261
0x30	A_SU_RD_STA[ST/Up]	0, 3	200
0x31	A_SU_DLYL[ST/Up]	0, 3	201
0x32	A_SU_DLYH[ST/Up]	0, 3	202
0x34	A_MS_RX[ST/Up]	0, 3	203
0x35	A_SU_STA[ST/Up]		204
0x3C	A_B1_RX[ST/Up]	0, 3	205
0x3D	A_B2_RX[ST/Up]	0, 3	205
0x3E	A_D_RX[ST/Up]	0, 3	206
0x3F	A_E_RX[ST/Up]	0, 3	207
0x40	R_GPIO_IN1	–	320
0x45	R_GPIO_IN2	–	320
0x48	R_GPIO_IN0	–	321
0x50	R_PLL_STA		308
0x88	R_INT_DATA	–	65

### Read / Write registers:

Address	Name	Reset group	Page
0x51	R_PLL_P		309
0x52	R_PLL_N		309
0x53	R_PLL_S		309
0x80	A_FIFO_DATA[FIFO]	–	135
0x84	A_FIFO_DATA_NOINC[FIFO]	–	135
0xC0	R_RAM_DATA	–	66
0xD0	A_SL_CFG[SLOT]	0, 3	262
0xF4	A_CH_MSK[FIFO]	0, 1	136
0xFA	A_CON_HDLC[FIFO]	0, 1	137
0xFB	A_SUBCH_CFG[FIFO]	0, 1	139
0xFC	A_CHANNEL[FIFO]	0, 1	140
0xFD	A_FIFO_SEQ[FIFO]	0, 1	141
0xFF	A_FIFO_CTRL[FIFO]	0, 1	142

**Note:** Reset group 0 = soft reset, 1 = HFC reset, 2 = PCM reset, 3 = line interface reset, H = hardware reset. See Table 9.4 on page 285 for a detailed reset group explanation.



# 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 '11001001' 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

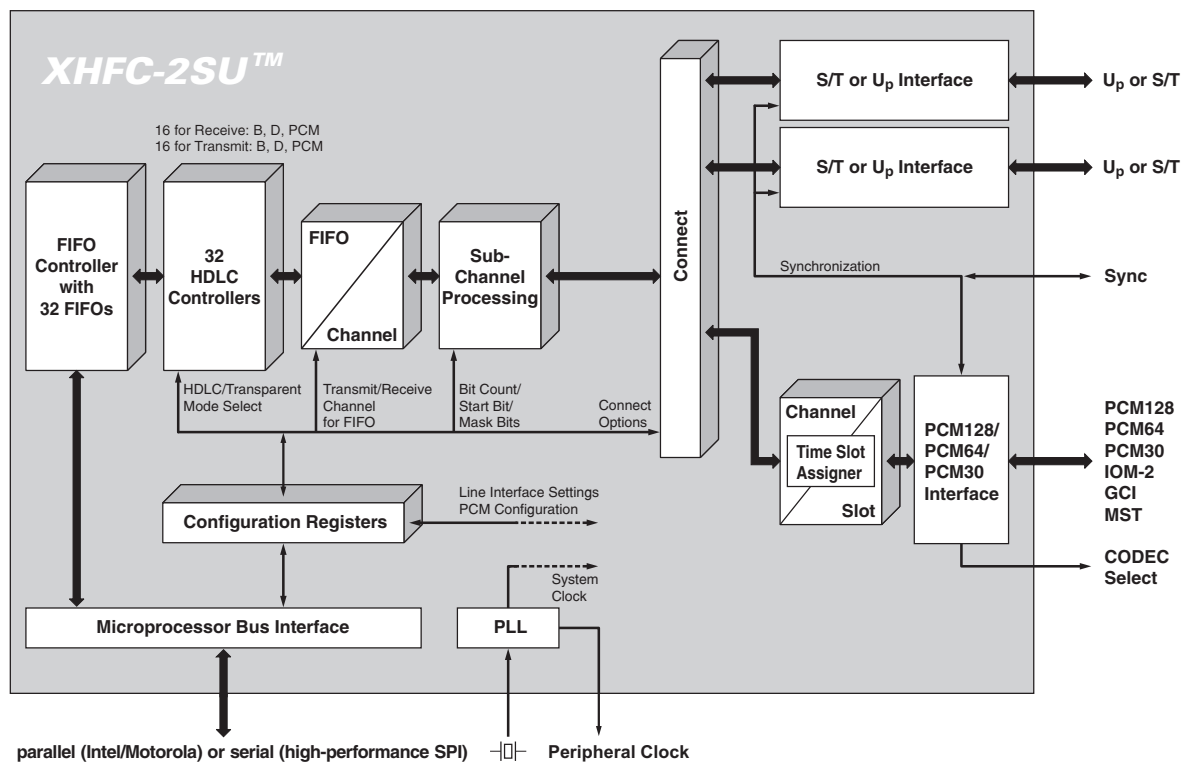


Figure 1.1: XHFC-2SU block diagram

## 1.1 System overview

XHFC-2SU is a ISDN transceiver chip for two ISDN S/T or  $U_{pN}/U_{p0}$ <sup>1</sup> 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.

---

<sup>1</sup> $U_{pN}/U_{p0}$  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 [8, 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)<sup>2</sup> or Siemens IOM<sup>TM</sup>-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

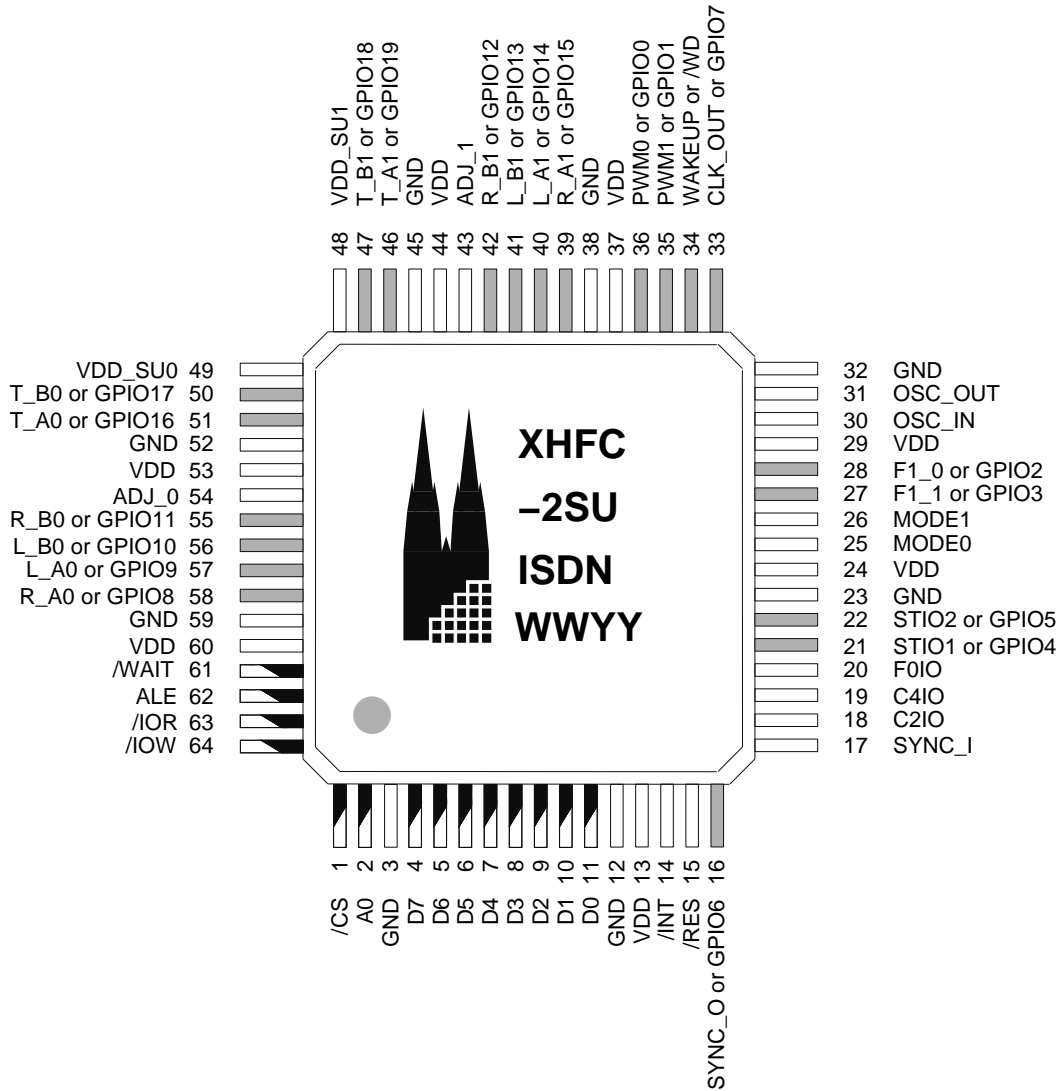
- Single 3.3 V power supply, CMOS technology 3.3 V, 5 V tolerant on nearly all inputs
- LQFP 64 package, 0.5 mm pin pitch
- RoHS compliant




---

<sup>2</sup>Mitel Serial Telecom bus

### 1.3 Pin description

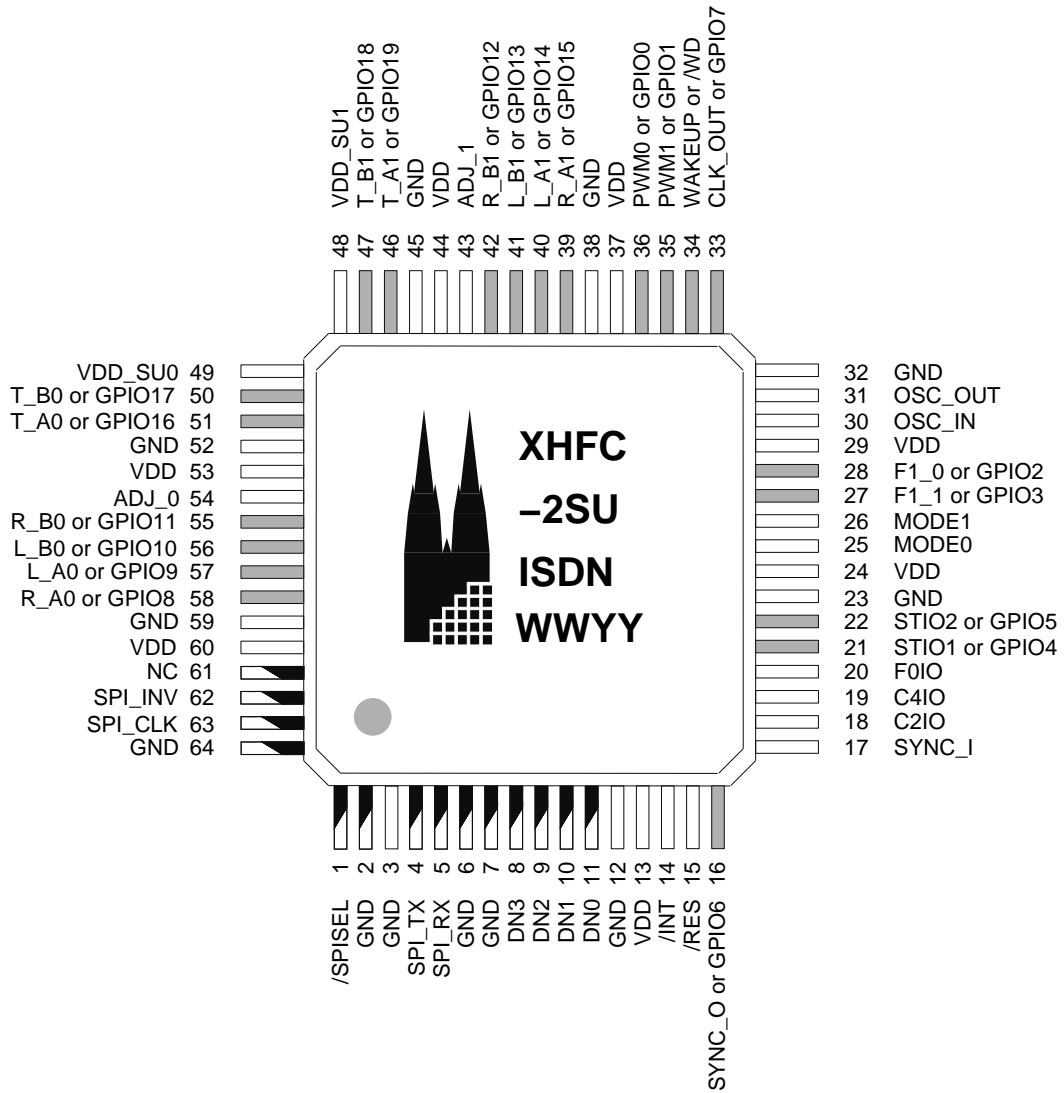
#### 1.3.1 Pinout diagram






-  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 <sub>in</sub> /V	I <sub>out</sub> /mA
<b>Microprocessor bus interface</b>						
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	
<b>Miscellaneous</b>						
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	
<b>PCM interface</b>						
16	1st function	SYNC_O	IO (2)	Synchronization output	TTL	8 mA
	2nd function	GPIO6	IO (2)	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
<b>Clock</b>						
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
<b>Miscellaneous</b>						
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		
<b>Universal ISDN Ports</b>						
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		

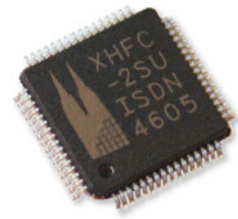
(continued on next page)

(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		
<b>Microprocessor bus interface</b>						
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		

<b>Legend:</b>	IO	(type 1)	Bidirectional pin, 9 pF pin capacitance
	IO	(type 2)	Bidirectional pin, input 5 V tolerant, 9 pF pin capacitance
	IO	(type 3)	Bidirectional pin
	IOPu		Bidirectional pin with internal pull-up resistor of 50 .. 80 k $\Omega$ to VDD , input 5 V tolerant, 9 pF pin capacitance
	I	(type 1)	Input pin, 9 pF pin capacitance
	I	(type 2)	Input pin, 3 pF pin capacitance
	I	(type 3)	Line interface input pin
	I	(type 4)	Oscillator input pin, 12 pF pin capacitance
	Isch		Schmitt Trigger input pin, 9 pF pin capacitance
	O	(type 1)	Output pin, 9 pF pin capacitance
	O	(type 2)	Line interface output pin
	O	(type 3)	Oscillator output pin, 10 pF pin capacitance
	Ood		Output pin with open drain, 9 pF pin capacitance
	NC		Must not be connected

Unused input pins should be connected to ground. Unused I/O pins should be connected via a e.g. 100 k $\Omega$  resistor to ground.



## 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	62	0x15	R_RAM_USE	64
0x08	R_RAM_ADDR	63	0x16	R_CHIP_ID	64
0x09	R_RAM_CTRL	63	0x1F	R_CHIP_RV	64
			0x88	R_INT_DATA	65
			Read / write registers:		
			Address	Name	Page
			0xC0	R_RAM_DATA	66

## 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*

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



### 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 2.3. The corresponding multiplexed modes are named 2m and 3m.<sup>1</sup>

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.

**Table 2.3:** *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 <sup>*1</sup>	0 <sup>*1</sup>	 <sup>*2</sup>	 <sup>*2</sup>

<sup>\*1</sup>: This pin should be fixed to ground

<sup>\*2</sup>: 1-pulse latches register address

### 2.2.2 Interface signals

The processor interface signals have different names for Motorola and Intel microprocessors. Table 2.4 shows the mapping with the pin names of XHFC-2SU.

### 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, address and data, are transferred through the pins D7 .. D0. 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 address without

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

Table 2.4: 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 Non-multiplexed	Intel Non-multiplexed	Motorola Multiplexed	Intel 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

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 address. The multiplexed address and data bus is D7 .. D0. 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 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 address
                   - ... (execute the interrupt service routine)
                   - address write access to restore the previous 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 2.5 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 2.6 for a quick timing and symbol list finding.

**Table 2.5:** Overview of accesses in parallel microprocessor interface mode (*X* = don't care)

<b>/CS</b>	<b>/IOR</b> (/DS, /RD)	<b>/IOW</b> (R/W, /WR)	<b>ALE</b>	<b>A0</b>	<b>Operation</b>	<b>Processor interface mode</b>
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 2.6:** Timing diagrams of the parallel microprocessor interface

Mode	Access type	Timing		Timing values	
		Figure	on page	table	on page
2 & 3	read	2.1	43	2.7	43
2 & 3	write	2.2	44	2.8	44
2m & 3m	read	2.3	46	2.9	46
2m & 3m	write	2.4	47	2.10	47

### 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 2.7.

Data can be read with<sup>2</sup>

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

in mode 2 (Motorola) or with

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

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  $t_{AH}$ .

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 2.8.

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

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

<sup>2</sup> $\overline{DS} + \overline{CS}$  means logical OR function of the two signals.

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

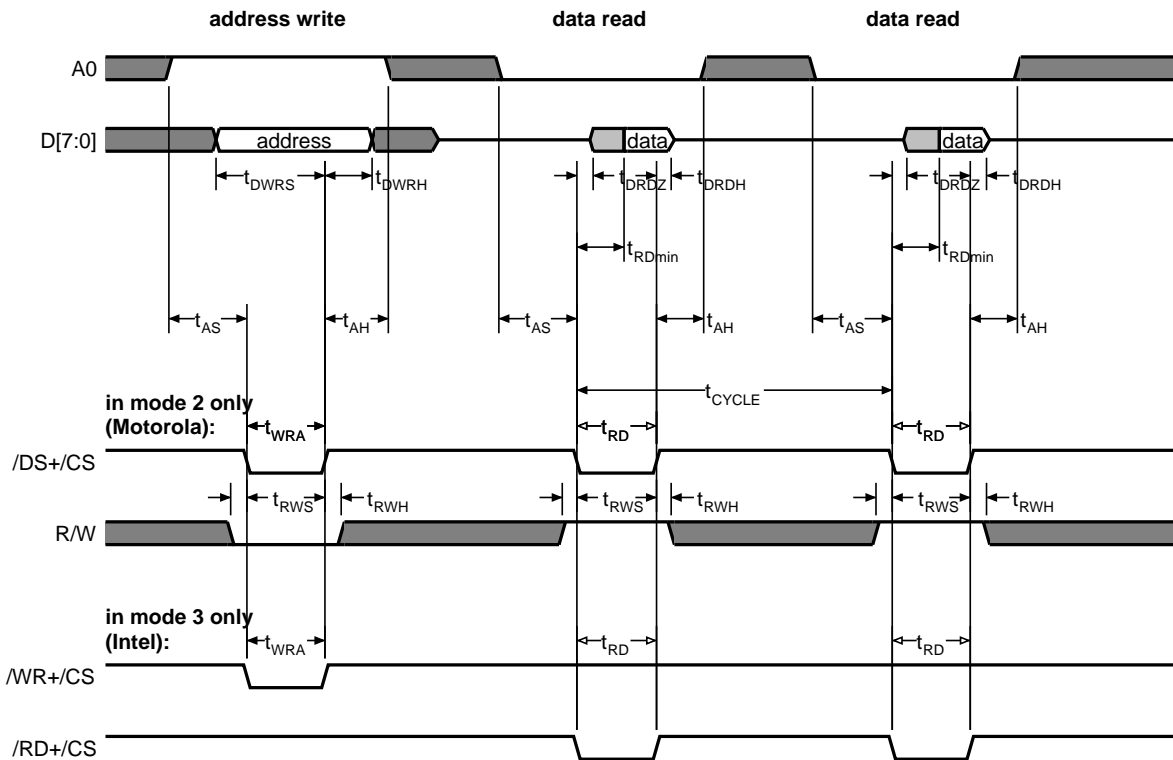


Figure 2.1: Bus interface read access in mode 2 (Motorola) and mode 3 (Intel)

Table 2.7: 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) $\downarrow$
$t_{WRA}$	20		Write time for address write
$t_{DWRH}$	25		Write data setup time to /DS+/CS (/WR+/CS) $\downarrow$
$t_{DWRH}$	0		Write data hold time from /DS+/CS (/WR+/CS) $\downarrow$
$t_{RD}$	25		Read time
$t_{CYCLE}$	30		/DS+/CS (/RD+/CS) $\downarrow$ to next /DS+/CS (/RD+/CS) $\downarrow$
	$3.5 \cdot t_{SYS}$		D7 = '0' with A0 = '1' (address range 0x00 .. 0x7F)
			D7 = '1' with A0 = '1' (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) $\downarrow$ 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 $\downarrow$ (in mode 2 only)

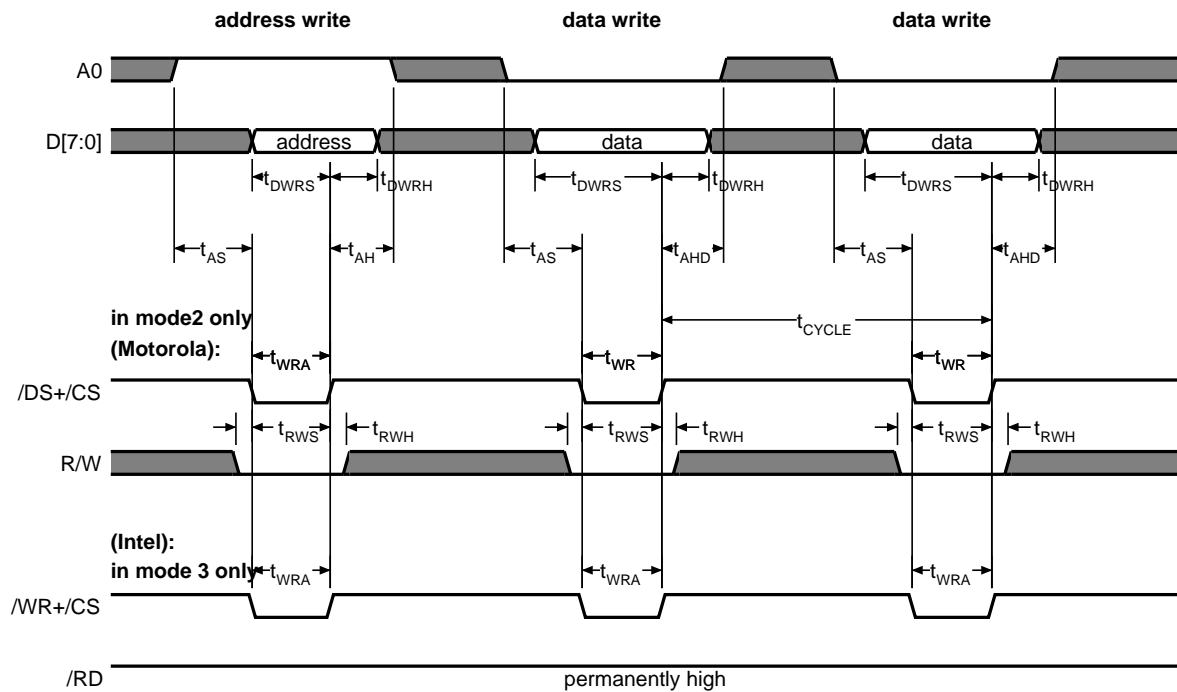


Figure 2.2: Bus interface write access in mode 2 (Motorola) and mode 3 (Intel)

Table 2.8: 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) $\downarrow$
	$1.5 \cdot t_{SYS}$		D7 = '0' with A0 = '1' (address range 0x00 .. 0x7F)
	$3.5 \cdot t_{SYS}$		D7 = '1' with A0 = '1' (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)

### 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 2.9.

Data can be read with<sup>4</sup>

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

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 requires a setup time  $t_{AS}$  in relation to  $\overline{\text{A}}$  of ALE. The hold time of these lines is  $t_{AH}$ . If consecutive read accesses are on the same address, multiple register address writes are not required.

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

#### Write access

8 bit processors write data like shown in Figure 2.4. Timing values are listed in Table 2.10.

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

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

<sup>4</sup> $/DS + /CS$  means logical OR function of the two signals.

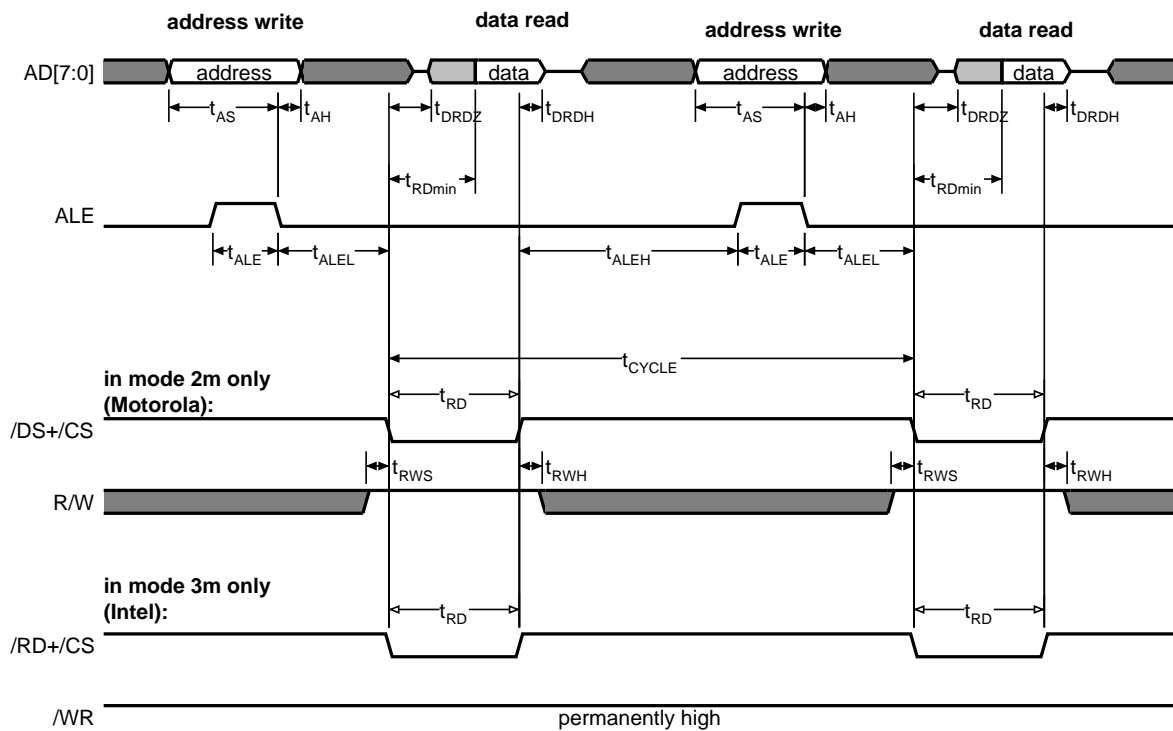


Figure 2.3: Bus interface read access in modes 2m and 3m (multiplexed)

Table 2.9: 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
$t_{CYCLE}$	30		/DS+/CS (/RD+/CS) $\downarrow$ to next /DS+/CS (/RD+/CS) $\downarrow$ AD7 = '0' with ALE = '1' (address range 0x00 .. 0x7F) $3.5 \cdot t_{SYS}$ AD7 = '1' with ALE = '1' (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)

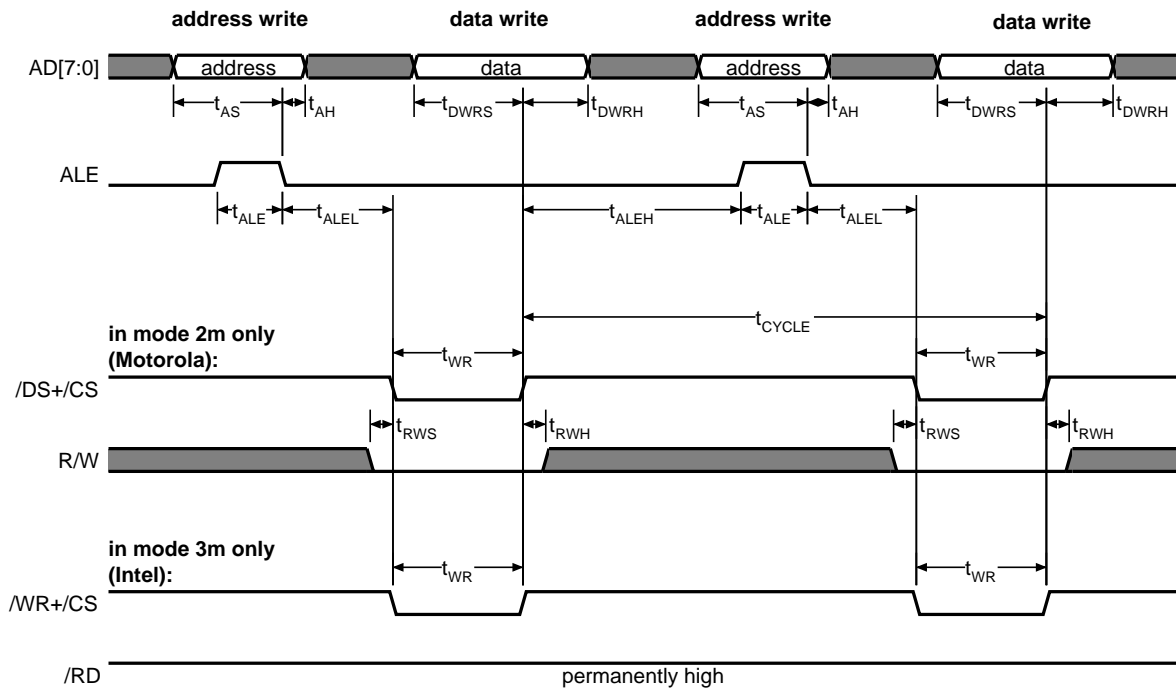


Figure 2.4: Bus interface write access in mode 2m and 3m (multiplexed)

Table 2.10: 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 $\overline{WR+}/CS \downarrow$
$t_{ALE}$	10		Address latch time
$t_{ALEL}$	0		ALE $\downarrow$ to $\overline{WR+}/CS \downarrow$
$t_{ALEH}$	0		$\overline{WR+}/CS \downarrow$ to ALE $\downarrow$
$t_{DWRS}$	20		Write data setup time to $\overline{WR+}/CS \downarrow$
$t_{DWRH}$	0		Write data hold time from $\overline{WR+}/CS \downarrow$
$t_{WR}$	20		Write time
$t_{CYCLE}$			$\overline{DS+}/CS (\overline{WR+}/CS) \downarrow$ to next $\overline{DS+}/CS (\overline{WR+}/CS) \downarrow$
	$1.5 \cdot t_{SYS}$		AD7 = '0' with ALE = '1' (address range 0x00 .. 0x7F)
	$3.5 \cdot t_{SYS}$		AD7 = '1' with ALE = '1' (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 $\overline{DS+}/CS \downarrow$ (in mode 2 only)

### 2.2.5 Microprocessor connection circuitries

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

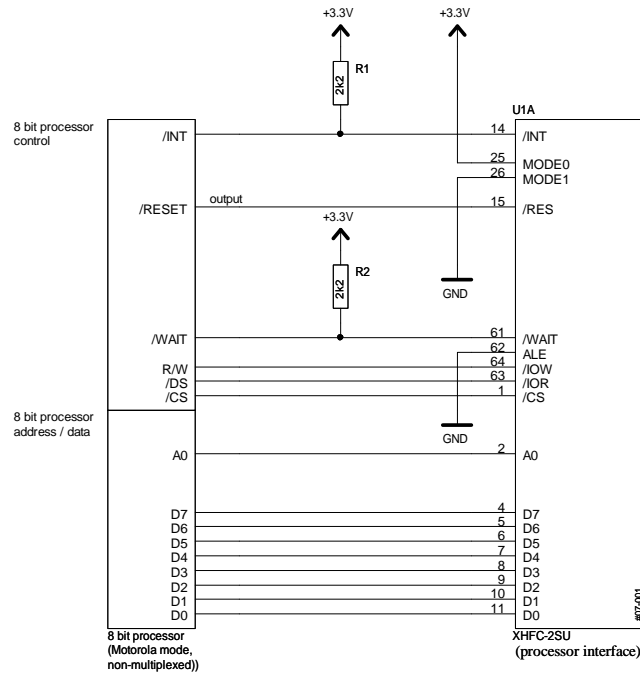


Figure 2.5: 8 bit Motorola processor circuitry example (mode 2)

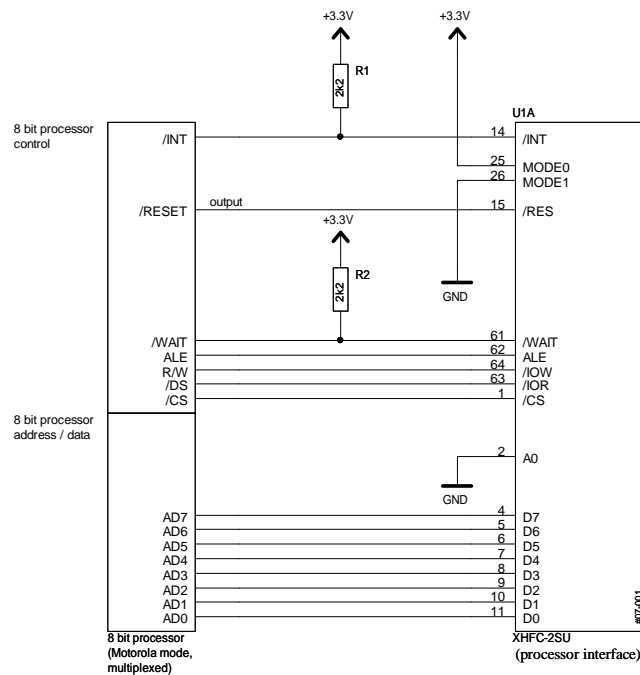


Figure 2.6: 8 bit Motorola processor circuitry example (mode 2m)

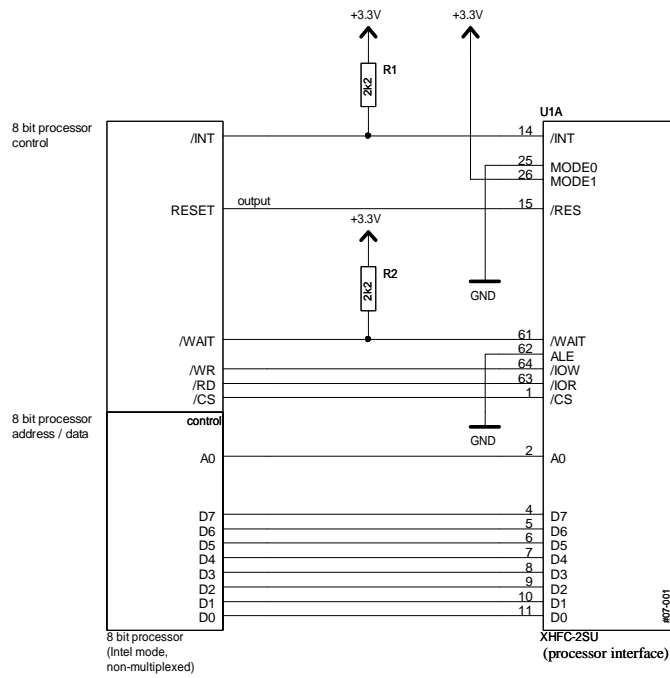


Figure 2.7: 8 bit Intel processor circuitry example (mode 3)

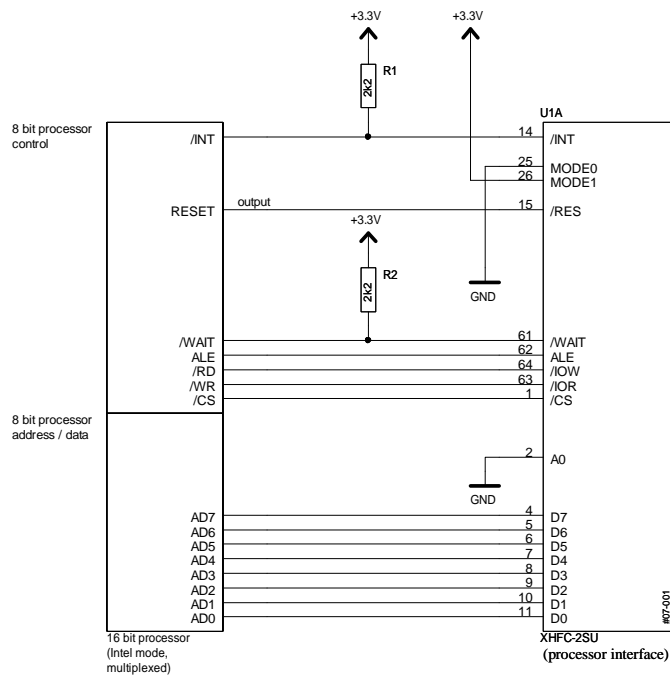


Figure 2.8: 8 bit Intel processor circuitry example (mode 3m)

## 2.3 Serial processor interface (SPI)

**Table 2.11:** Overview of the SPI interface pins

Number	Name	Description
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 2.11. Additional pins are used for SPI clock inversion and SPI device number selection.

SPI interface mode is selected by  $MODE0 = '0'$  and  $MODE1 = '0'$  (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 2.12 and 2.13 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  $C3..C0 = DN3..DN0$  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 2.14 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 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.

**Table 2.12:** SPI control byte with A = '0'

Bit	Name	Description
7	R	'0' = write '1' = read
6	A	'0' = 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 2.13:** SPI control byte with A = '1'

Bit	Name	Description
7	R	'0' = write '1' = read
6	A	'1' = 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 2.14:** SPI control commands

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			–

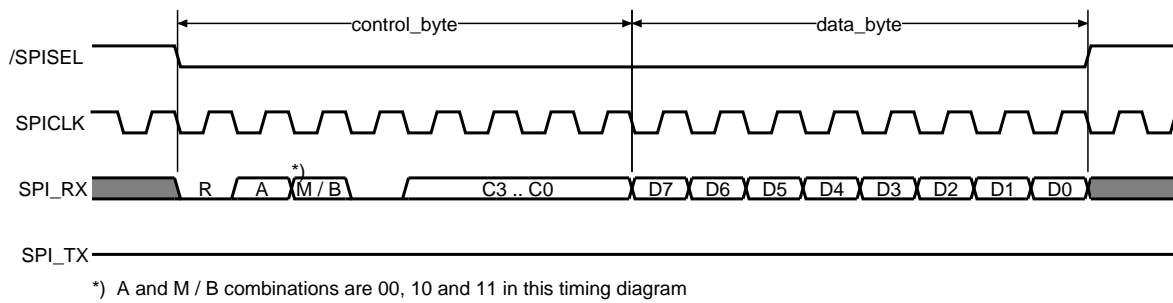


Figure 2.9: 16 bit SPI write transaction (R = '0', one data byte)

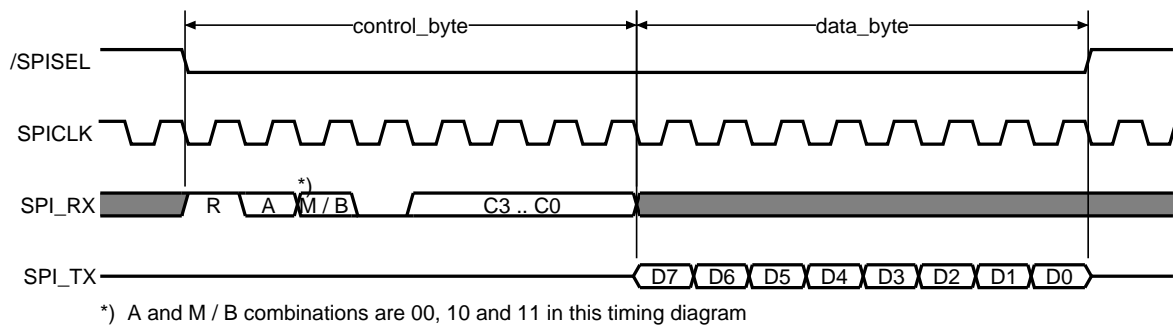


Figure 2.10: 16 bit SPI read transaction (R = '1', one data byte)

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 /SPISEL = '1'. In this case the transmission pauses and will be continued after /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  $f_{SPICLK} = 4\text{MHz}$ , 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

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

40 bit SPI transaction have a minimal length of 40 clock cycles. With  $f_{SPICLK} = 4\text{MHz}$ , 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

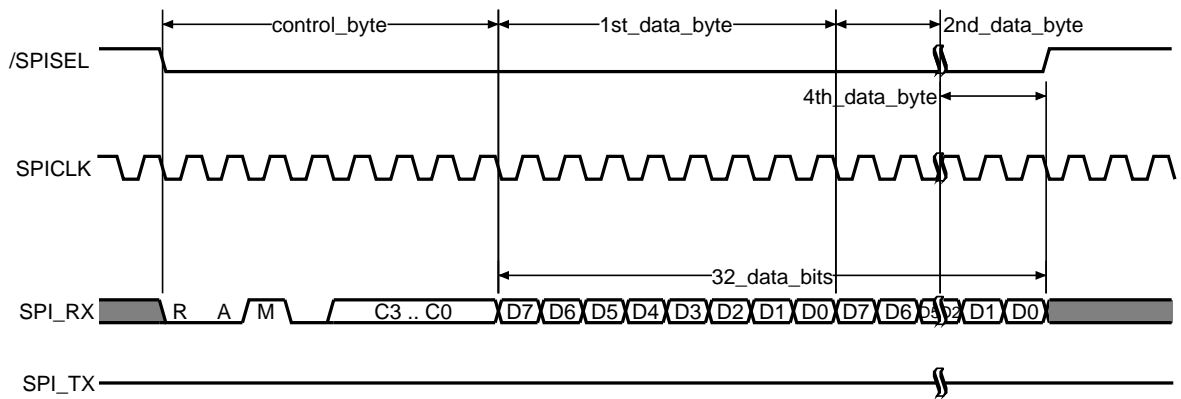


Figure 2.11: 40 bit SPI write transaction ( $R = '0'$ , four data bytes)

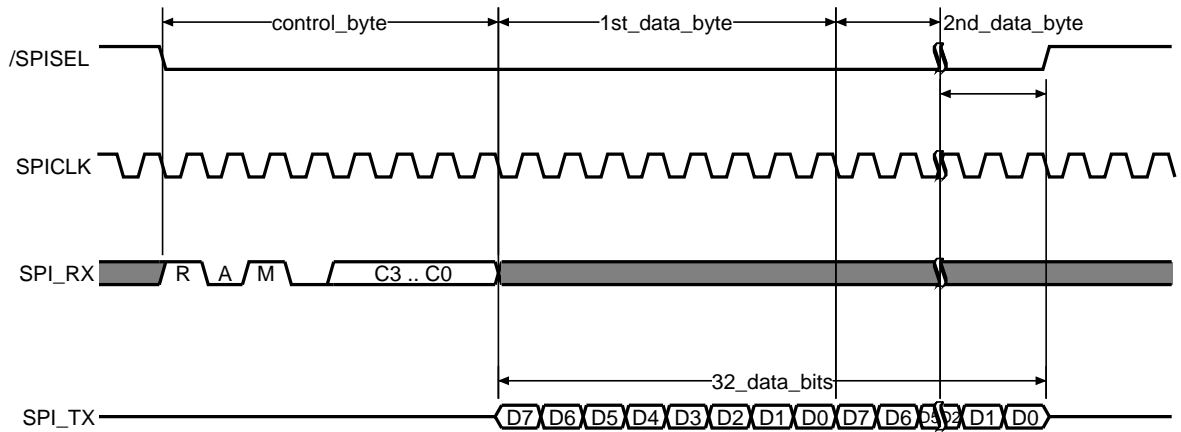


Figure 2.12: 40 bit SPI read transaction ( $R = '1'$ , four data bytes)

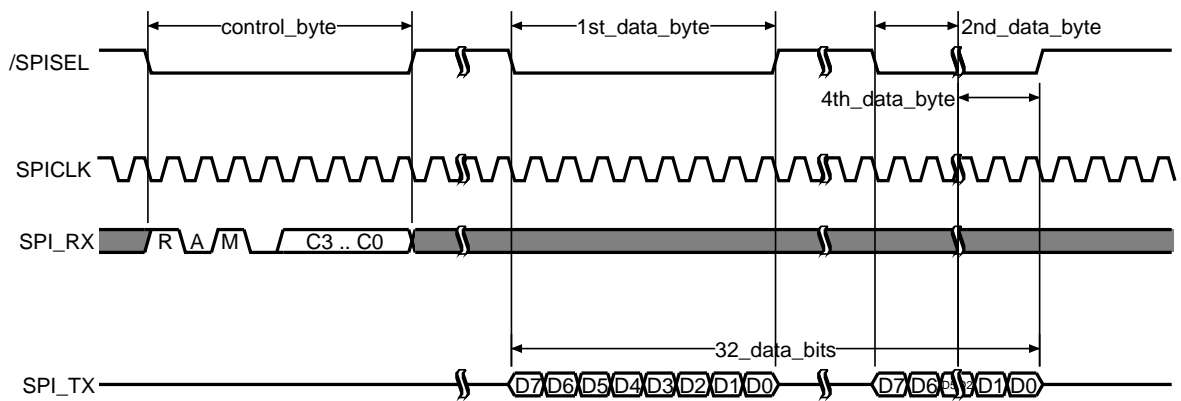


Figure 2.13: Split 40 bit SPI read transaction ( $R = '1'$ )

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 /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 /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 /SPISEL = '1' within the data byte, the transaction quits immediately (SPI\_TX is always tri-state when /SPISEL = '1'). The next received byte is expected to be the data byte again.
5. When an invalid control byte is received, it is ignored and the next received byte is expected to be a control byte.

#### 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 '01X0 CCCC'). 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 '01X0 CCCC'). 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 '01X0 CCCC' 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 /SPISEL = '1' between the transactions. This splits the sequence into a 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 /SPISEL remains '0' for several transactions.

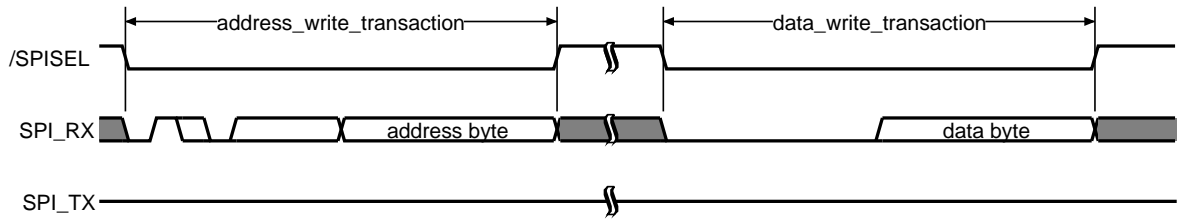


Figure 2.14: Register write access (transaction sequence)

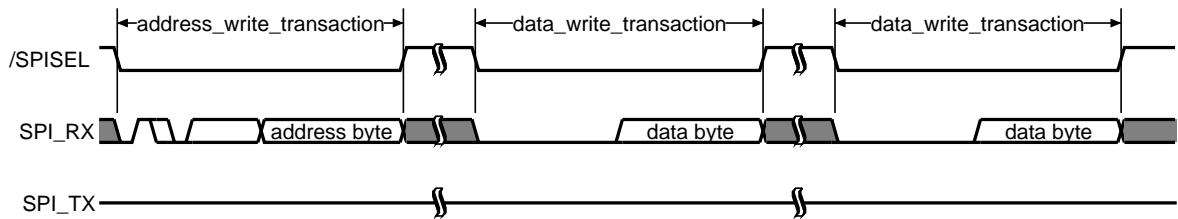


Figure 2.15: Multiple write accesses to the same register

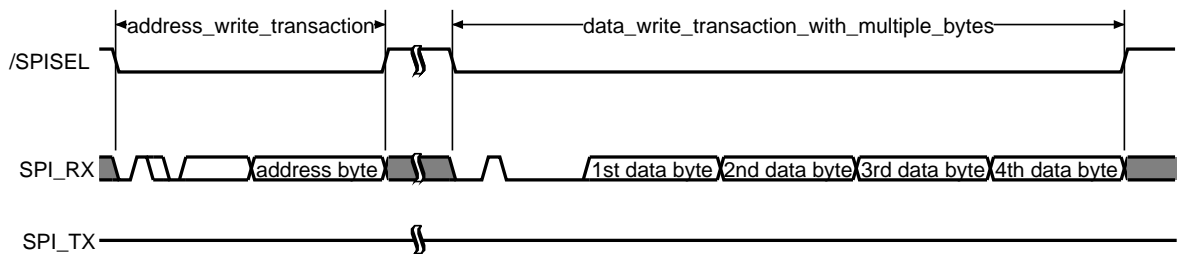


Figure 2.16: Register write access with a 40 bit transaction

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.

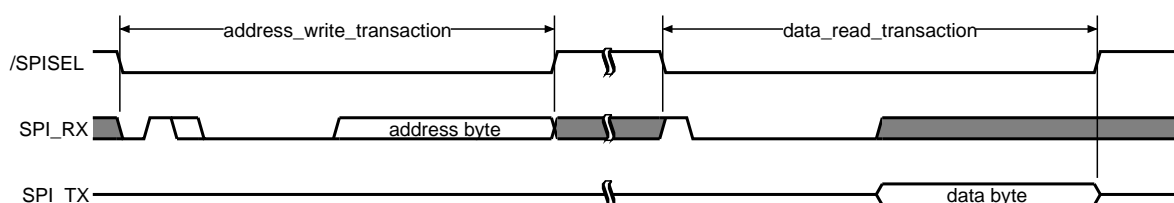


Figure 2.17: Register read access (transaction sequence)

Figures 2.17 to 2.19 show  $\text{/SPISEL} = '1'$  between the transactions. This splits the sequence into a 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  $\text{/SPISEL}$  remains '0' for several transactions.

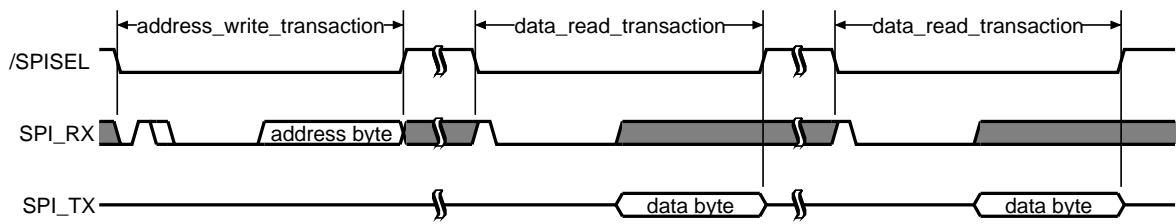


Figure 2.18: Multiple read accesses to the same register

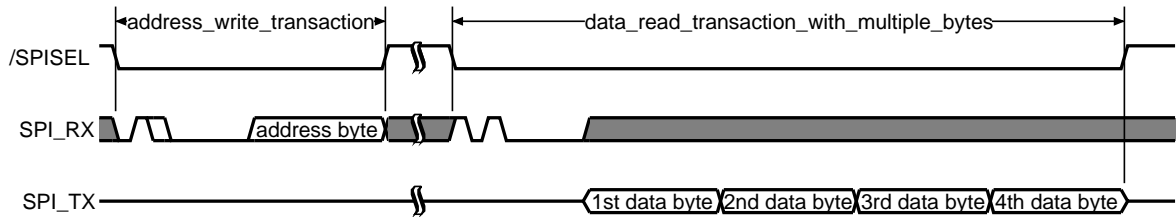


Figure 2.19: Register read access with a 40 bit transaction

### 2.3.6 Register access duration

There are several SPI data rates due to the different register access types. Table 2.15 gives some examples for  $f_{SPICLK} = 25\text{ MHz}$  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 2.15: 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 $f_{SPICLK} = 25\text{ MHz}$ and $N = 16$ , e.g.
'X000 0000'	single	1	32	$T = \frac{32}{f_{SPICLK}}$	$DR = \frac{1}{T} = 781\text{ kByte/s}$
'X000 0000'	single	$N$	$(N + 1) \cdot 16$	$T = \frac{(N+1) \cdot 16}{f_{SPICLK}}$	$DR = \frac{N}{T} = 1471\text{ kByte/s}$
'X010 0000'	multiple	4	56	$T = \frac{56}{f_{SPICLK}}$	$DR = \frac{4}{T} = 1786\text{ kByte/s}$
'X010 0000'	multiple	$4N$	$16 + N \cdot 40$	$T = \frac{16+N \cdot 40}{f_{SPICLK}}$	$DR = \frac{4N}{T} = 2439\text{ kByte/s}$

### 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 address
                   - ... (execute the interrupt service routine)
                   - address write transaction to restore the previous 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). 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  $t_{\text{DESEL}} \geq 2\text{ ns}$  after the sampling time at the earliest.

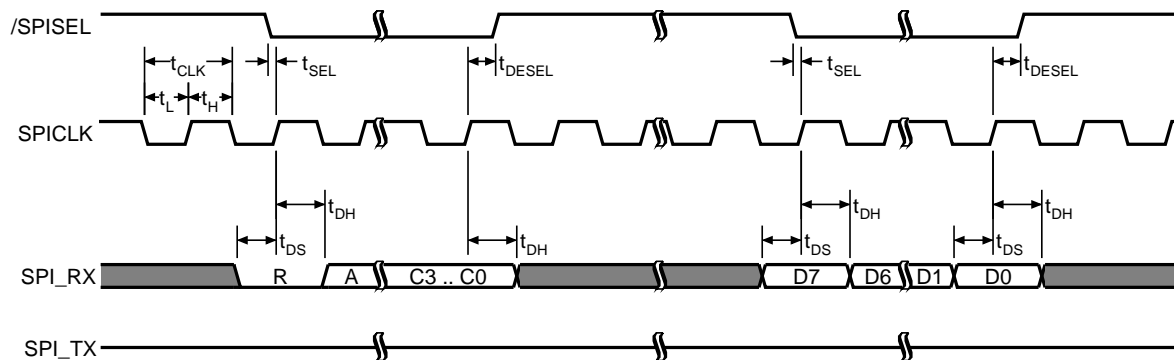


Figure 2.20: SPI timing diagram for data write transactions

The timing diagram for data read transactions (data from slave to master) is shown in Figure 2.21. 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  $t_{\text{DEN}} \leq 10\text{ ns}$  after the falling edge of SPI\_CLK at the latest.

Table 2.16: 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

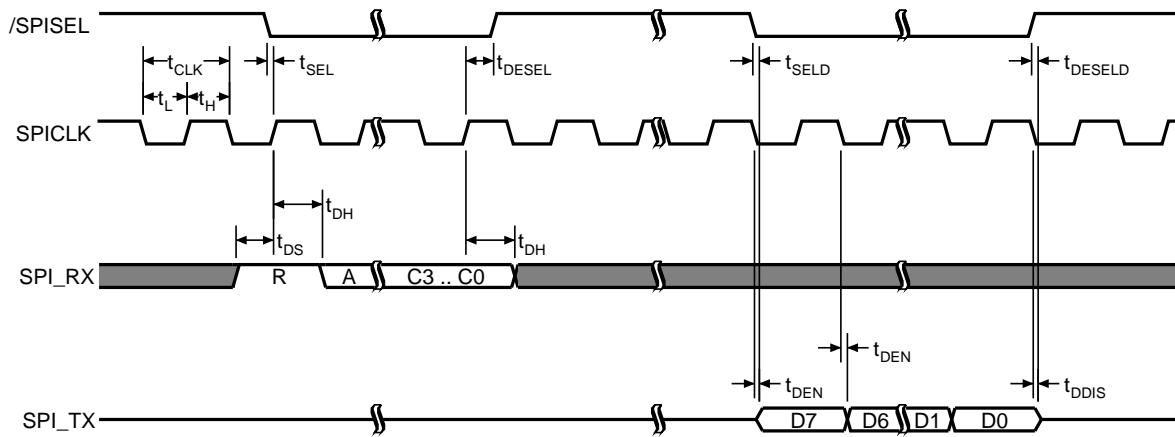


Figure 2.21: SPI timing diagram for data read transactions

Table 2.17: 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
$t_{DESEL}$	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

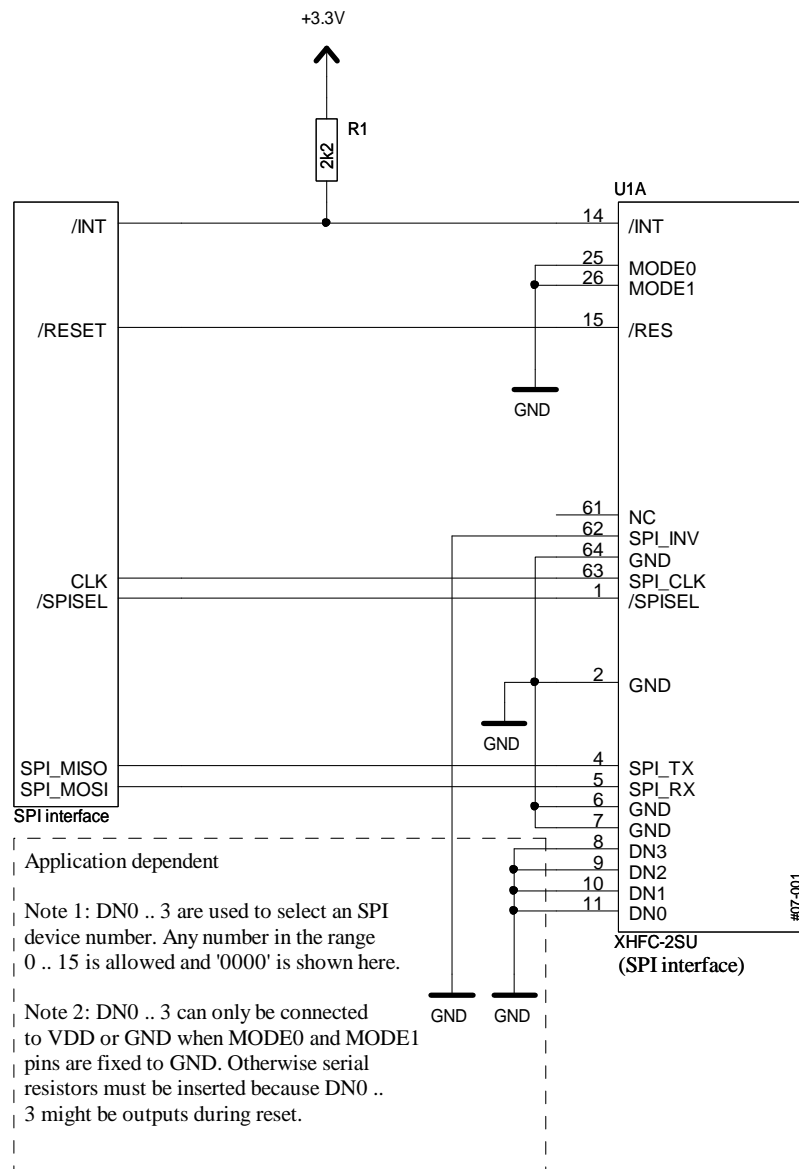


Figure 2.22: SPI connection circuitry

## 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)	0x01	
<b>Common control register</b>			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	<b>V_FIFO_LPRIO</b>	<b>FIFO access priority for host accesses</b> '0' = normal priority '1' = low priority
2	0	(reserved)	Must be '0'.
3	0	<b>V_NT_SYNC</b>	<b>Synchronization source for NT mode</b> 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) <b>Note:</b> 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	<b>V_OSC_OFF</b>	<b>Disable oscillator</b> '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	<b>V_SU_CLK</b>	<b>Line interface clock selection</b> 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)	0x08
<b>Address pointer, lower part</b>			
Lower address byte for SRAM access.			
Bits	Reset value	Name	Description
7..0	0x00	V_RAM_ADDR0	Address bits 7..0

R_RAM_CTRL		(w)	0x09
<b>SRAM access control register</b>			
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	<b>Address reset</b> '0' = normal operation '1' = address bits 0..11 are set to zero This bit is automatically cleared.
7	0	V_ADDR_INC	<b>Address increment</b> '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)	0x15
<b>SRAM duty factor</b>			
Usage of SRAM access bandwidth by the internal data processor.			
Bits	Reset value	Name	Description
7..0		V_SRAM_USE	<b>Relative duty factor</b> 0x00 = 0% bandwidth used 0x7C = 100% bandwidth used

R_CHIP_ID		(r)	0x16
<b>Chip identification register</b>			
Bits	Reset value	Name	Description
7..0	0x61	V_CHIP_ID	<b>Chip identification code</b> 0x61 means XHFC-2SU.

R_CHIP_RV		(r)	0x1F
<b>XHFC-2SU revision</b>			
Bits	Reset value	Name	Description
3..0	0	V_CHIP_RV	<b>Chip revision 0</b>
7..4	0	(reserved)	

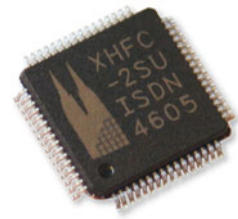
<b>R_INT_DATA</b>	(r)	0x88								
<p><b>Internal data register</b></p> <p>This register can be read to access data with Read* method.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Reset value</th> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7..0</td> <td></td> <td><b>V_INT_DATA</b></td> <td><b>Internal data buffer</b></td> </tr> </tbody> </table>			Bits	Reset value	Name	Description	7..0		<b>V_INT_DATA</b>	<b>Internal data buffer</b>
Bits	Reset value	Name	Description							
7..0		<b>V_INT_DATA</b>	<b>Internal data buffer</b>							

(See Section 2.2.3.2 on page 40 for details on Read\* access.)

### 2.5.3 Read/write register

<b>R_RAM_DATA</b>		<b>(r*/w)</b>		<b>0xC0</b>
<b>SRAM data access</b>				
Direct access to the internal SRAM				
Bits	Reset value	Name	Description	
7..0	0x00	<b>V_RAM_DATA</b>	<b>SRAM data access</b> The address must be written into registers R_RAM_ADDR and R_RAM_CTRL in advance.	

(See Section 2.2.3.2 on page 40 for details on Read\* access.)



## Chapter 3

# XHFC-2SU data flow

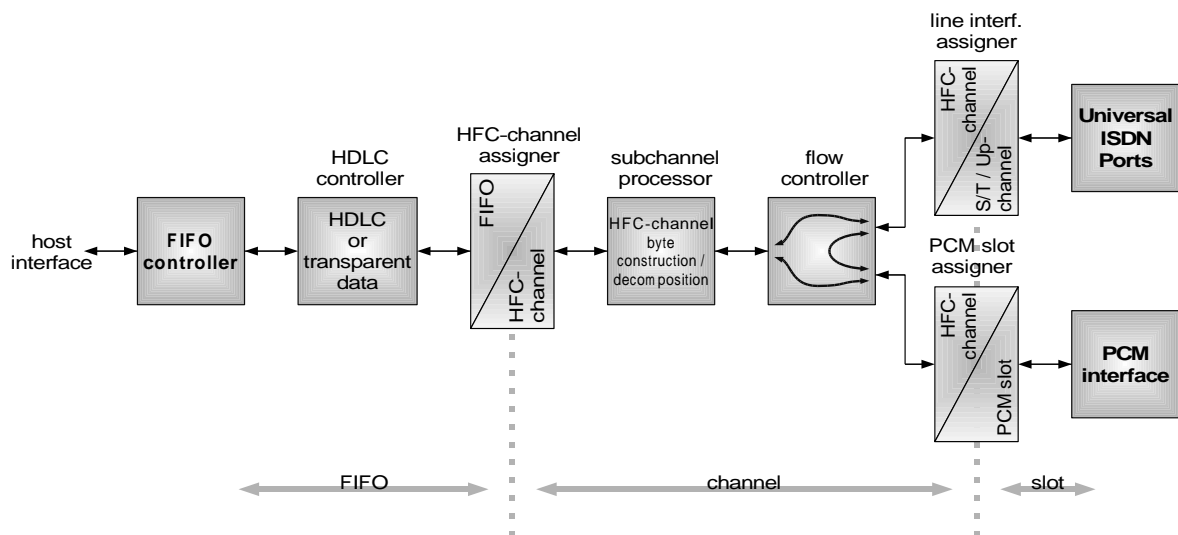


Figure 3.1: Data flow block diagram

## 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 other<sup>1</sup>.

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

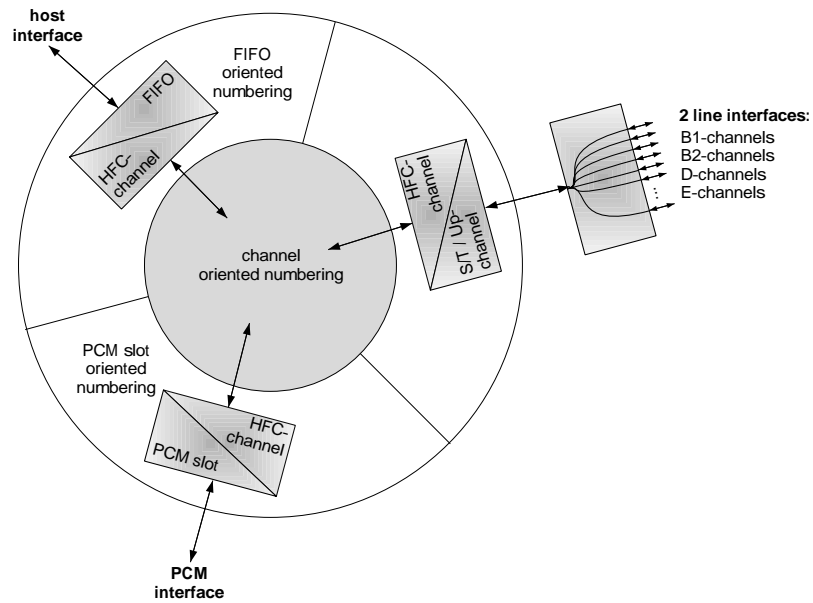
Figure 3.2 clarifies the relationship and the differences between the numbering of FIFOs, HFC-channels and PCM time slots. The inner circle symbolizes the HFC-channel oriented part of the data flow, while the outer circle shows the connection of three data sources and data drains respectively. The ST/ $U_p$  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.

---

<sup>1</sup>In 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  $\mu$ 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  $\mu$ 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  $\mu$ 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  $\mu$ s cycle, then copied into the transmit buffer during the second 125  $\mu$ s cycle and finally transmitted from the interface during the third 125  $\mu$ 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  $\mu$ 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 119). In any case  $n$  cannot exceed 15. There are other configurations with  $n = 7$  and  $n = 3$ .

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  $R\_FIFO$ .

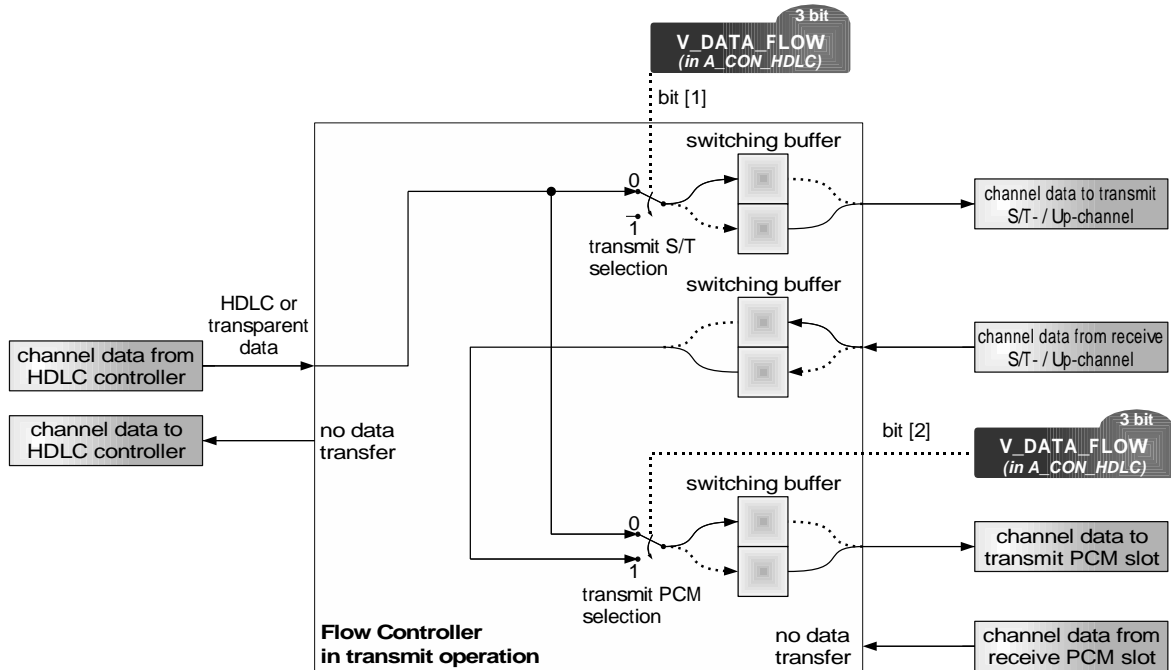


Figure 3.3: The flow controller in transmit operation

- 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 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.

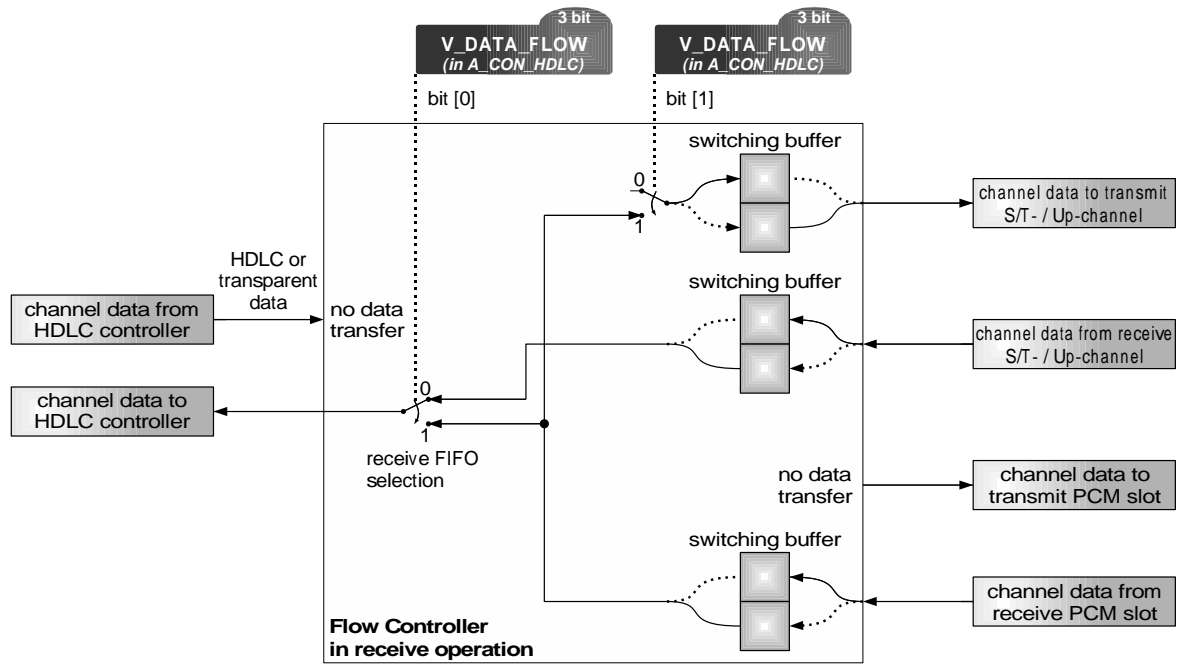


Figure 3.4: The flow controller in receive FIFO operation

- 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 connections<sup>2</sup> 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.

<sup>2</sup>In 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 <sub>p</sub>	FIFO ← ST/U <sub>p</sub>
'001'	FIFO → ST/U <sub>p</sub>	FIFO → PCM	FIFO ← PCM	
'010'	FIFO → PCM		FIFO ← ST/U <sub>p</sub>	ST/U <sub>p</sub> ← PCM
'011'		FIFO → PCM	FIFO ← PCM	ST/U <sub>p</sub> ← PCM
'100'	ST/U <sub>p</sub> → PCM	FIFO → ST/U <sub>p</sub>	FIFO ← ST/U <sub>p</sub>	
'101'	FIFO → ST/U <sub>p</sub>	ST/U <sub>p</sub> → PCM	FIFO ← PCM	
'110'		ST/U <sub>p</sub> → PCM	ST/U <sub>p</sub> ← PCM	FIFO ← ST/U <sub>p</sub>
'111'		ST/U <sub>p</sub> → PCM	ST/U <sub>p</sub> ← 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 <sub>p</sub>	'0' (TX)	'x0x'	'000'
FIFO ← ST/U <sub>p</sub>	'1' (RX)	'xx0'	
FIFO → PCM	'0' (TX)	'0xx'	'001'
FIFO ← PCM	'1' (RX)	'xx1'	
ST/U <sub>p</sub> → PCM	'0' (TX)	'1xx'	'110'
ST/U <sub>p</sub> ← PCM	'1' (RX)	'x1x'	



Table 3.3: Line interface assigner

HFC-channel		ST/U <sub>p</sub> -channel			HFC-channel		ST/U <sub>p</sub> -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<sub>p</sub>-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.

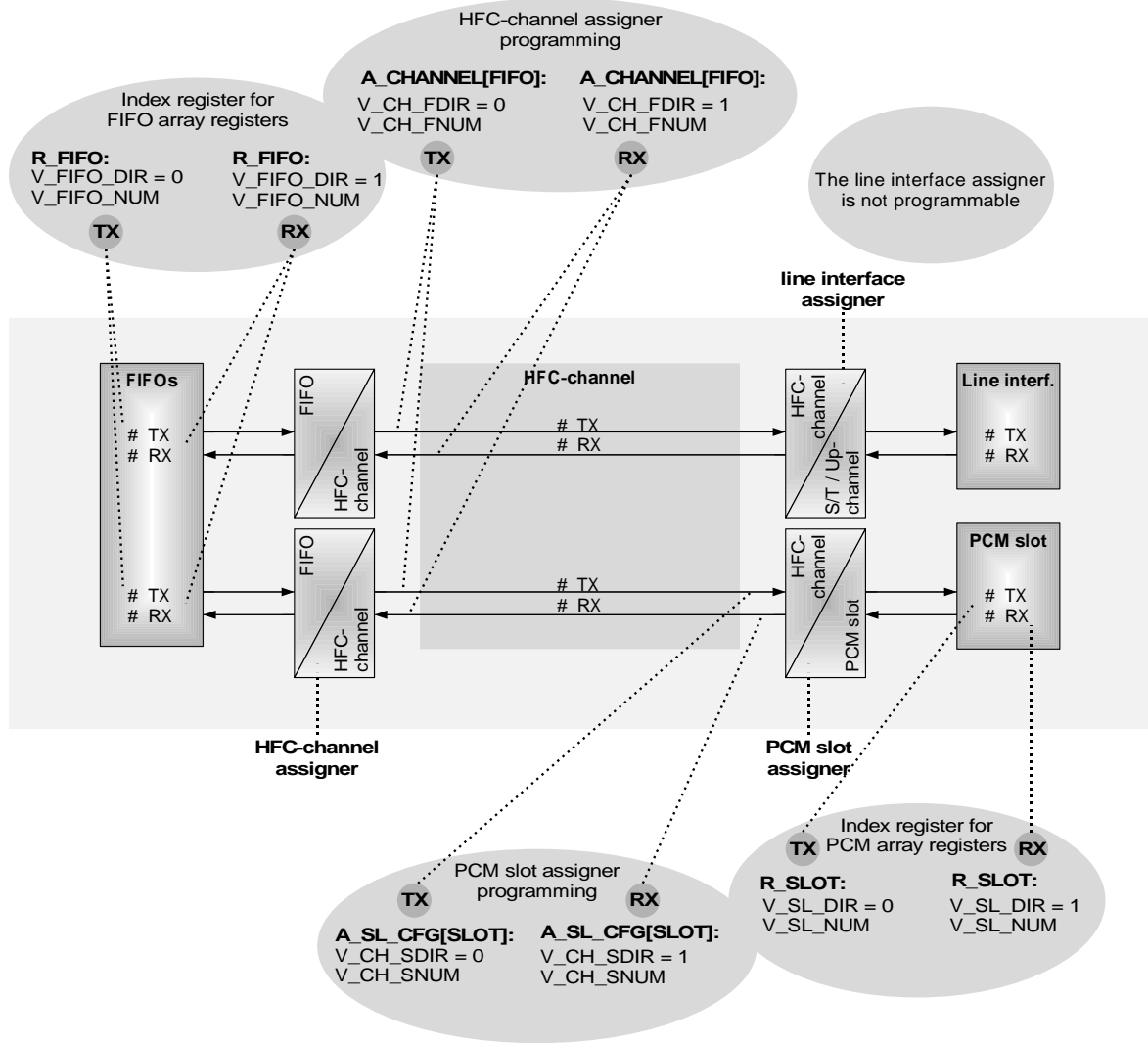


Figure 3.5: Overview of the assigner programming

## 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<sub>p</sub>-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 FIFOs<sup>5</sup>. 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.

---

<sup>5</sup>Except 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			Index register in			Index meaning in		
	name	address	I/O mode	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  $\mu$ 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 100 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<sub>p</sub>, ❷ FIFO-to-PCM, ❸ PCM-to-ST/U<sub>p</sub> 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<sub>p</sub> and PCM boxes on the right side contain the ST/U<sub>p</sub>-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.

#### ❶ FIFO-to-ST/U<sub>p</sub>

As HFC-channel and FIFO numbers are the same in SM, a selected ST/U<sub>p</sub>-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<sub>p</sub> connection, the desired ST/U<sub>p</sub>-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.

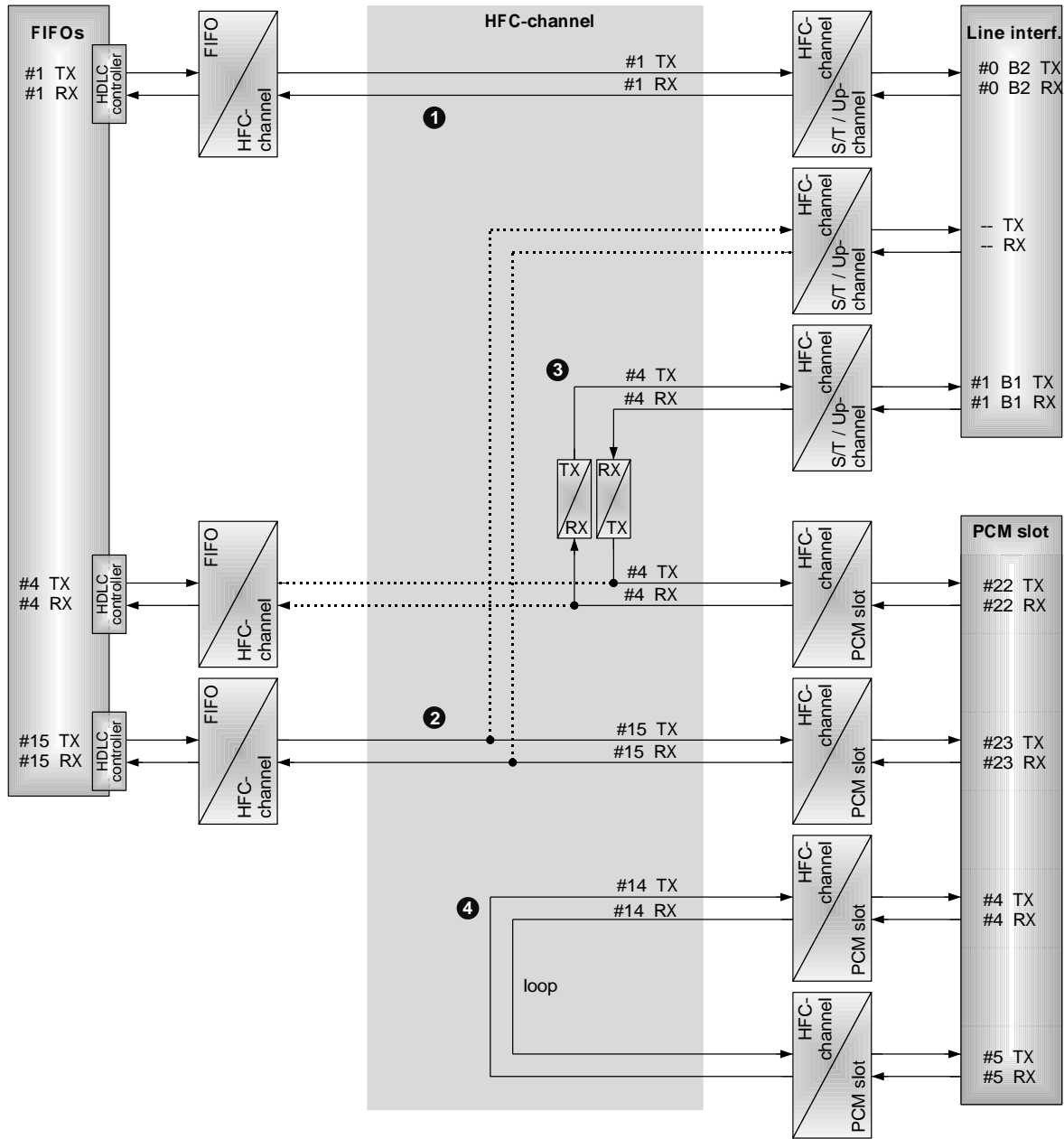


Figure 3.6: SM example

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 <sub>p</sub> , 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 <sub>p</sub> )

## 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<sub>p</sub>-channels, every used HFC-channel blocks the connected ST/U<sub>p</sub>-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 <sub>p</sub> , 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)
A_CON_HDLC[15,RX]	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO ← PCM)
	<hr/>	
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<sub>p</sub>

A direct PCM-to-ST/U<sub>p</sub> 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<sub>p</sub> interface neither in receive nor in transmit direction. A PCM-to-ST/U<sub>p</sub> connection operates always in transparent mode.

For a PCM-to-ST/U<sub>p</sub> 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<sub>p</sub> 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 <sub>p</sub> → PCM)
<hr/>		
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 <sub>p</sub> , ST/U <sub>p</sub> ← 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<sub>p</sub> -channel to avoid waste of resources.

Register setup:		(SM 4 no. 1)
R_SLOT	: V_SL_DIR = 1	(receive 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 100 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/U<sub>p</sub>, ❷ FIFO-to-PCM and ❸ PCM-to-ST/U<sub>p</sub>). 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.

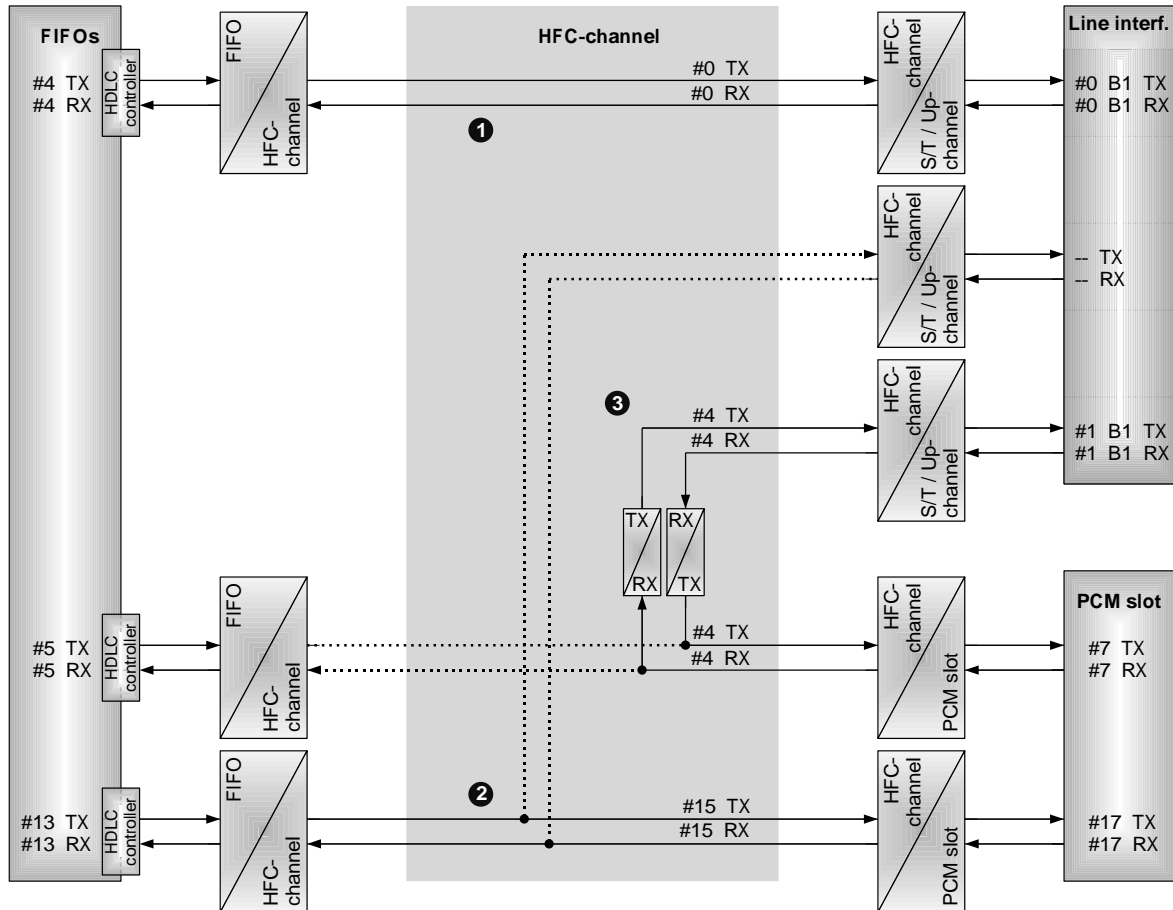


Figure 3.8: CSM example

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.

#### ❶ FIFO-to-ST/U<sub>p</sub>

HFC-channel and FIFO numbers can be chosen independently from each other. This is shown in the FIFO-to-ST/U<sub>p</sub> 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 <sub>p</sub> , 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 <sub>p</sub> )
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<sub>p</sub>-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 <sub>p</sub> , 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<sub>p</sub>

The PCM-to-ST/U<sub>p</sub> 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<sub>p</sub> 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<sub>p</sub> interface neither in receive nor in transmit direction. A PCM-to-ST/U<sub>p</sub> connection operates always in transparent mode.

For a PCM-to-ST/U<sub>p</sub> 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.<sup>6</sup>

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 <sub>p</sub> → 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 <sub>p</sub> , ST/U <sub>p</sub> ← 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)

<sup>6</sup>It is not forbidden to connect the blocked FIFOs at the ST/U<sub>p</sub> 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 78).

#### 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.

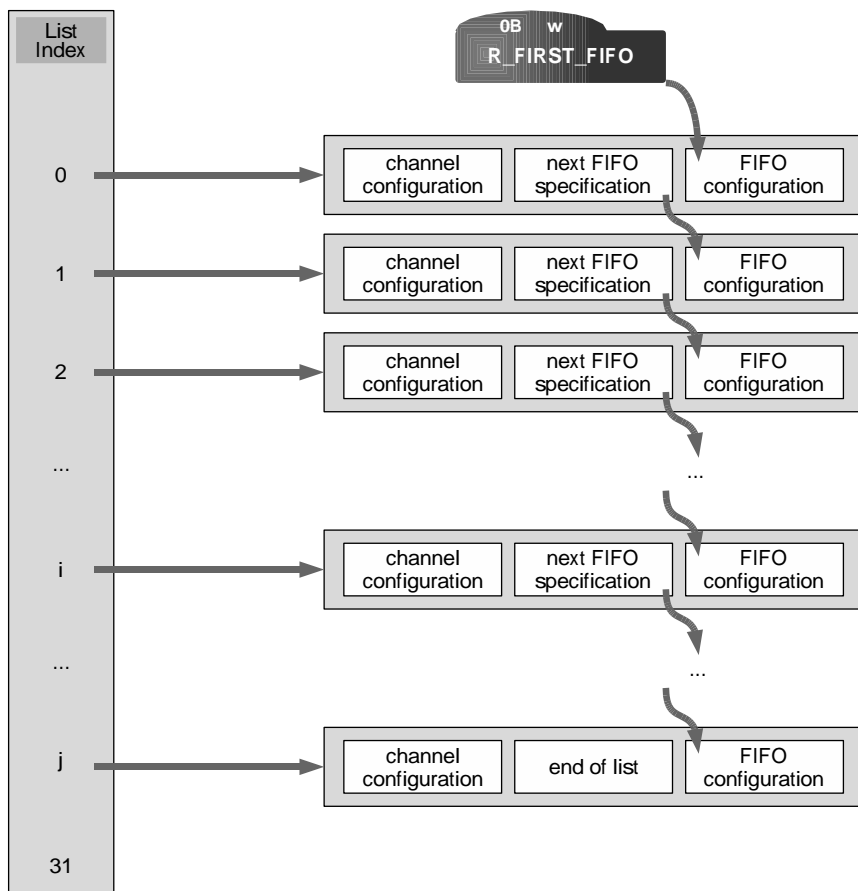


Figure 3.10: FSM list processing

### 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 78). All other FIFO array registers remain indexed by R\_FIFO.

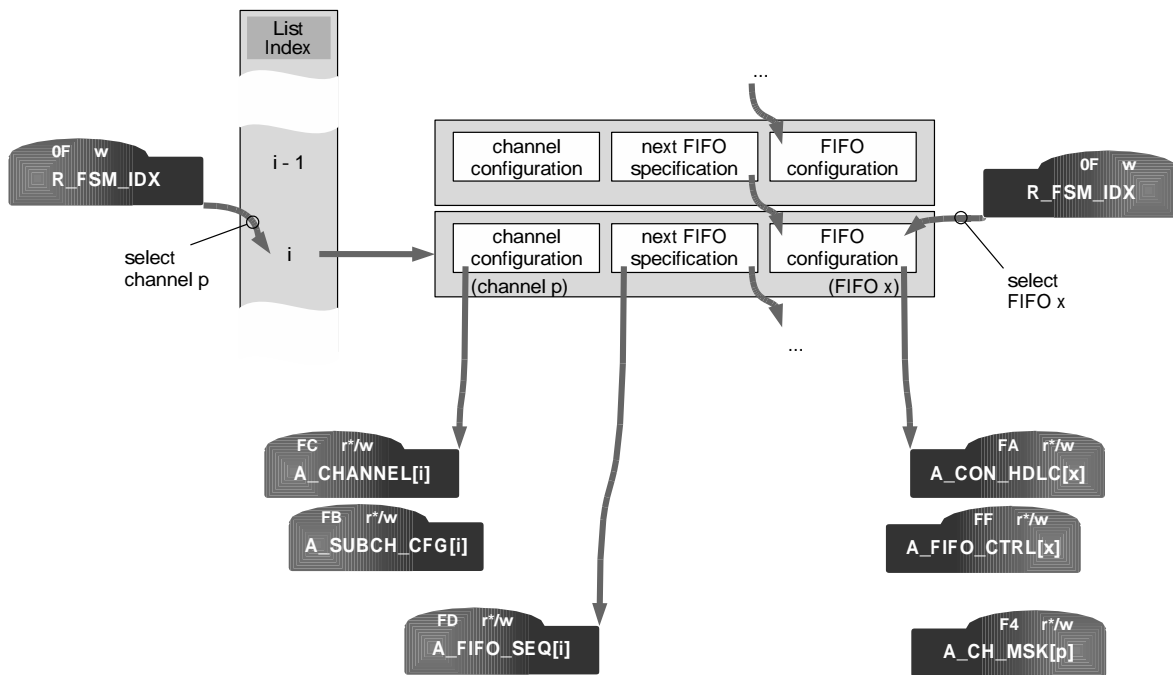


Figure 3.11: FSM list programming

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  $A\_FIFO\_SEQ[i - 1] = \text{FIFO } x$ , 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 101).

### 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<sub>p</sub> , ❷ 8 kByte/s-FIFO-to-PCM and ❸ 16 kByte/s-FIFO-to-ST / U<sub>p</sub> ). 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.

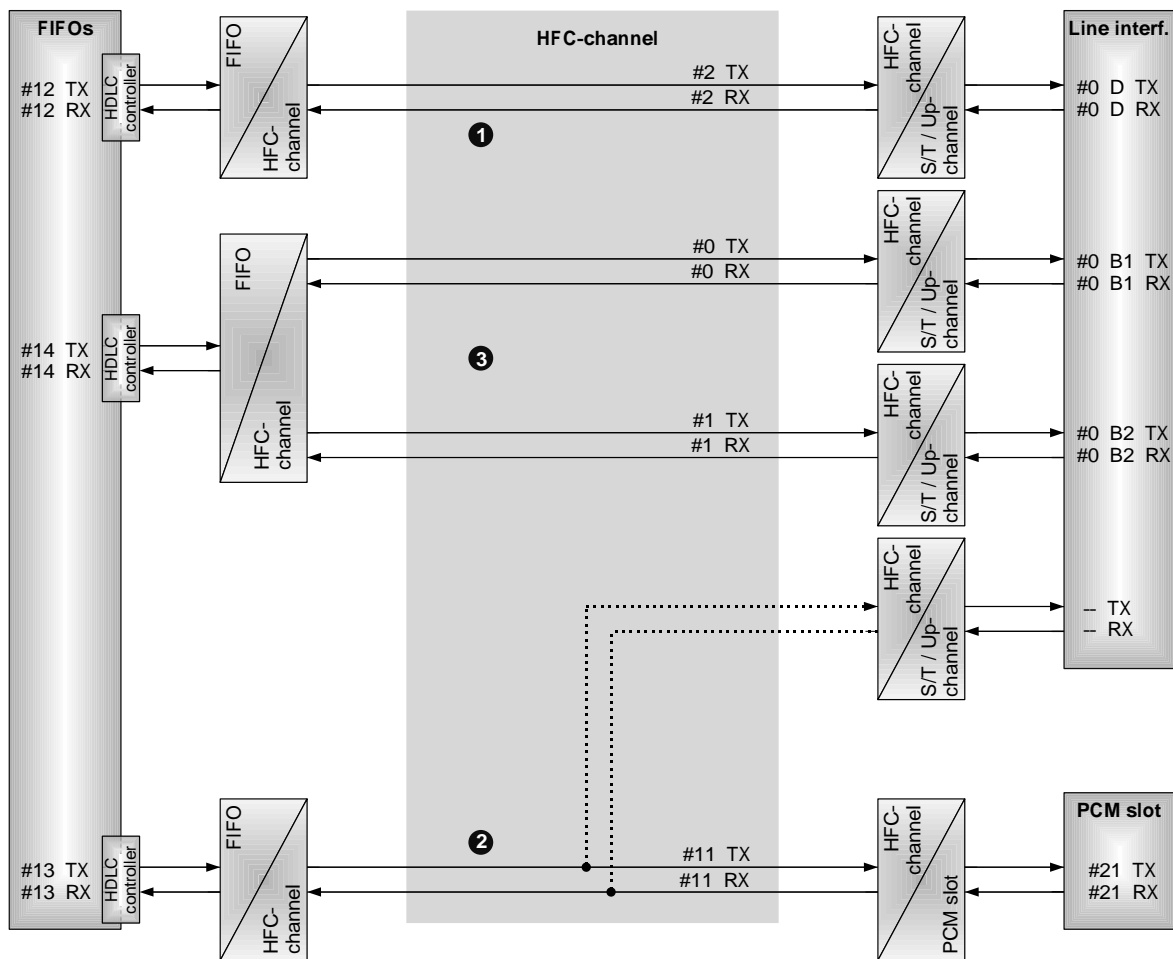


Figure 3.12: FSM example

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
❶	0	FIFO[12,TX] → ST/U <sub>p</sub> interf. #0, D TX
❶	1	FIFO[12,RX] ← ST/U <sub>p</sub> interf. #0, D RX
❷	2	FIFO[13,RX] ← PCM time slot[21,RX]
❷	3	FIFO[13,TX] → PCM time slot[21,TX]
❸	4	FIFO[14,TX] → ST/U <sub>p</sub> interf. #0, B1 TX
❸	5	FIFO[14,RX] ← ST/U <sub>p</sub> interf. #0, B1 RX
❸	6	FIFO[14,TX] → ST/U <sub>p</sub> interf. #0, B2 TX
❸	7	FIFO[14,RX] ← ST/U <sub>p</sub> interf. #0, B2 RX

**❶ FIFO-to-ST/U<sub>p</sub>**

The bidirectional FIFO-to-ST/U<sub>p</sub> connection use the list indices 0 and 1. Registers A\_CHANNEL and A\_FIFO\_SEQ are indexed by the list index.

<b>Register setup:</b>		(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.

<b>Register setup:</b>		(FSM ① 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 <sub>p</sub> , 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 <sub>p</sub> )

## ② 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.

<b>Register setup:</b>		(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 <sub>p</sub> , 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<sub>p</sub>-channels

The last setting shows a channel bundling configuration of one FIFO to two B-channels of the ST/U<sub>p</sub> 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 <sub>p</sub> , 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 <sub>p</sub> )	

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/U<sub>p</sub> 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.

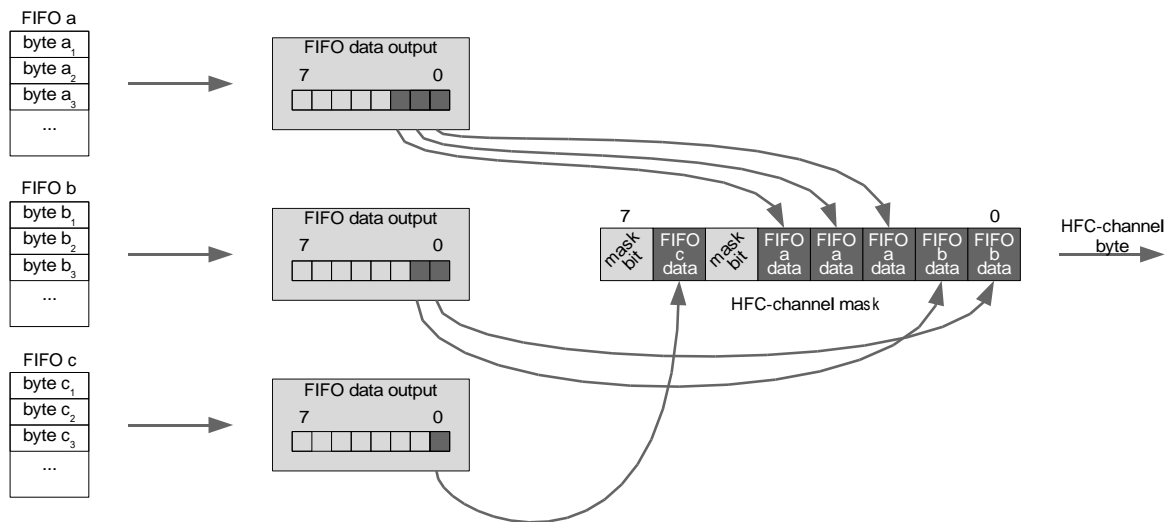


Figure 3.13: General structure of the subchannel processor shown with an example of three connected FIFOs

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.

#### 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

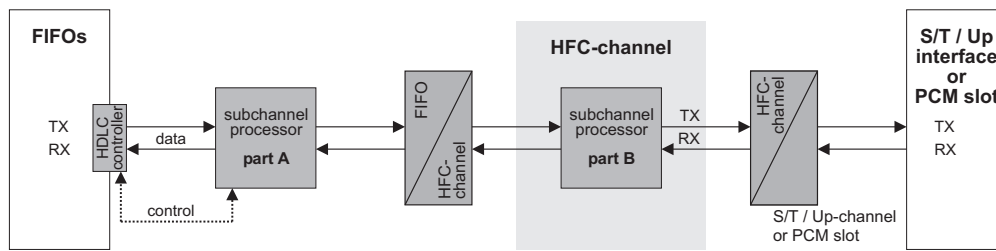


Figure 3.14: Location of the subchannel parts A and B in the data flow diagram

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  $\mu$ 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  $\mu$ 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  $\mu$ 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.

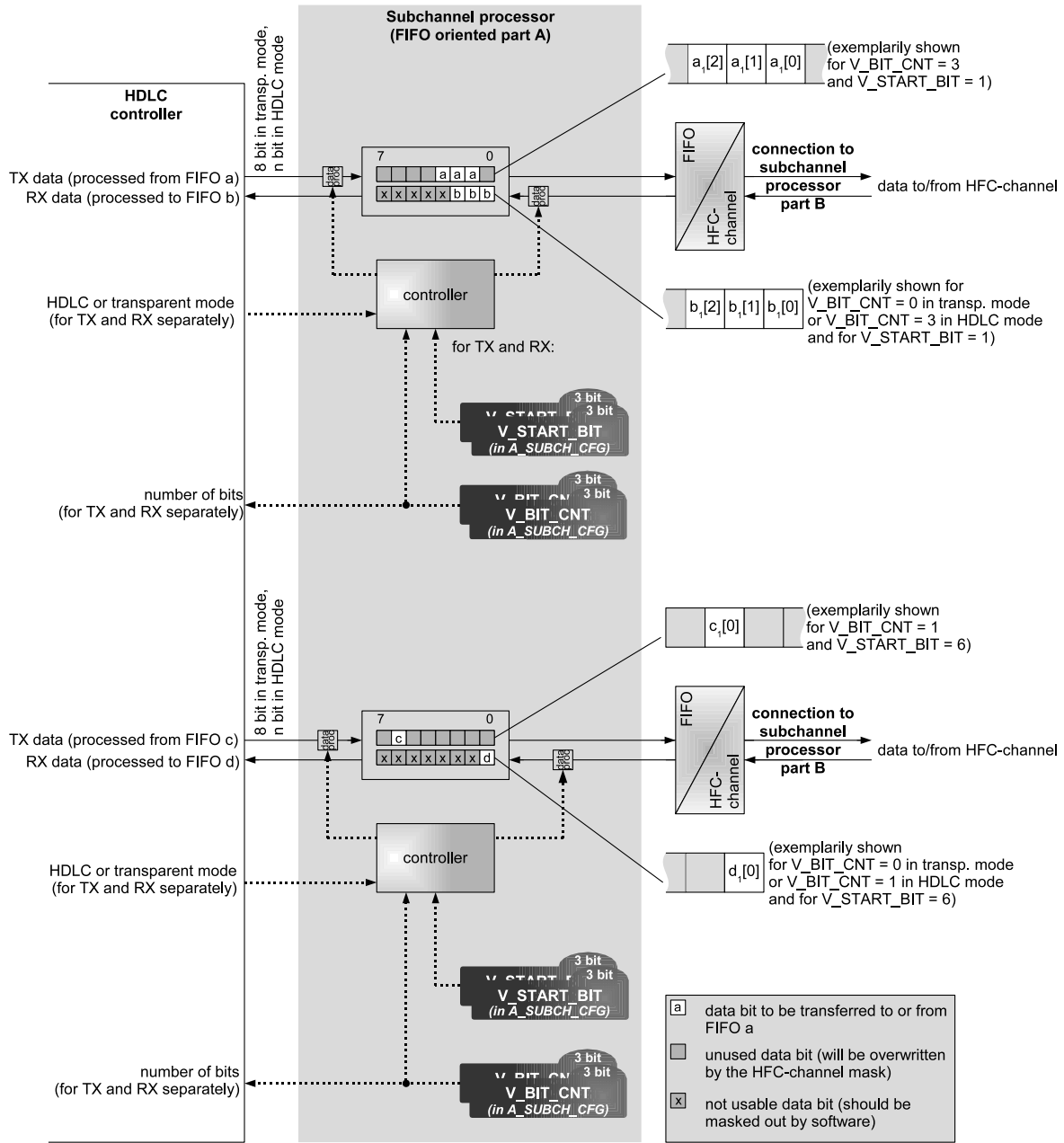


Figure 3.15: Part A of the subchannel processor

### 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 µs 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 µs 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 µs cycle it transmits and receives always one data byte to/from the connected interface (either PCM or ST/U<sub>p</sub> 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.

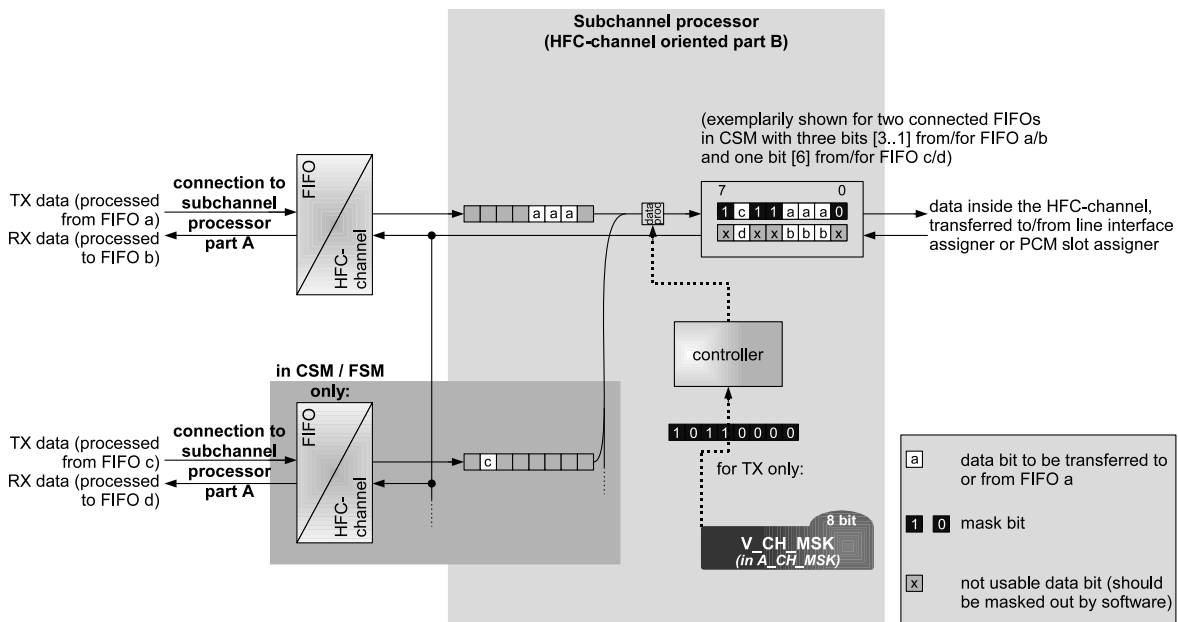


Figure 3.16: Part B of the subchannel processor

### 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<sub>p</sub> 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.

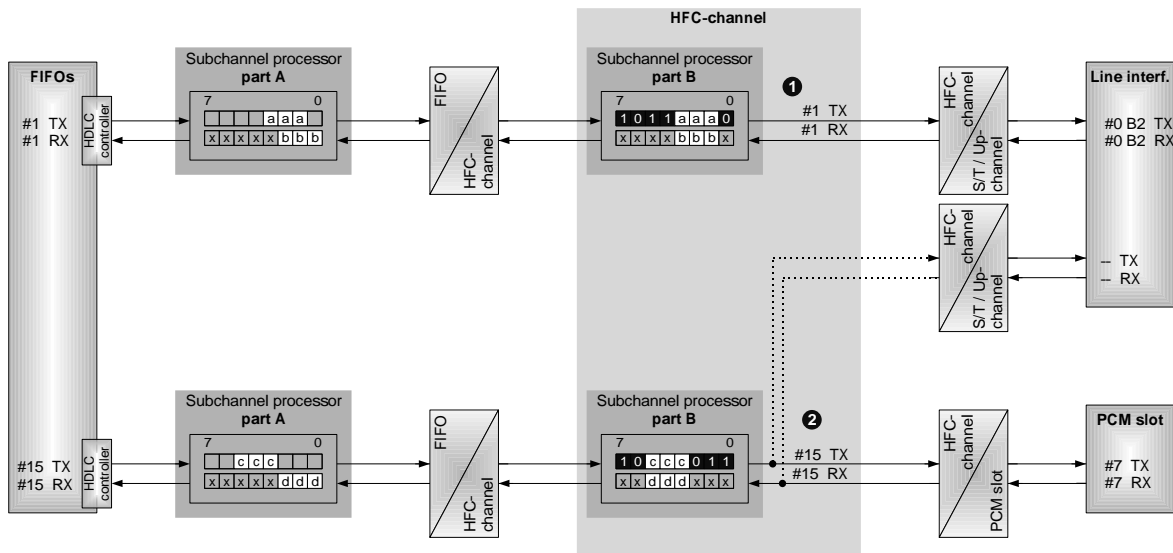


Figure 3.17: SM example with subchannel processor

#### ❶ FIFO-to-ST/U<sub>p</sub> (TX)

The first setting shows a FIFO-to-ST/U<sub>p</sub> 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 <sub>p</sub> , 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 = '10110000'	(mask byte)

### ① FIFO-to-ST/U<sub>p</sub> (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 <sub>p</sub> )
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)

### ② FIFO-to-PCM (TX)

The second *Simple Mode* configuration connects a FIFO in HDLC mode with the PCM interface<sup>7</sup>. Bitmap  $V\_BIT\_CNT$  in register  $A\_SUBCH\_CFG[FIFO]$  defines three FIFO data bits to

<sup>7</sup>HDLC bit stuffing is not shown in this example.

**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:	$x$	$x$	$x$	$x$	$b_3[2]$	$b_3[1]$	$b_3[0]$	$x$
...								

be transmitted during every 125  $\mu$ 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.

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 <sub>p</sub> , 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 = '10110011'	(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 <sub>1</sub> [2]	c <sub>1</sub> [1]	c <sub>1</sub> [0]	0	1	1
HFC-channel transmit byte 2:	1	0	c <sub>1</sub> [5]	c <sub>1</sub> [4]	c <sub>1</sub> [3]	0	1	1
HFC-channel transmit byte 3:	1	0	c <sub>2</sub> [0]	c <sub>1</sub> [7]	c <sub>1</sub> [6]	0	1	1
HFC-channel transmit byte 4:	1	0	c <sub>2</sub> [3]	c <sub>2</sub> [2]	c <sub>2</sub> [1]	0	1	1
...								

### 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 (① FIFO-to-ST/U<sub>p</sub> and ② FIFO-to-PCM) with two FIFOs per direction each.

#### ① FIFO-to-ST/U<sub>p</sub> (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.

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

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

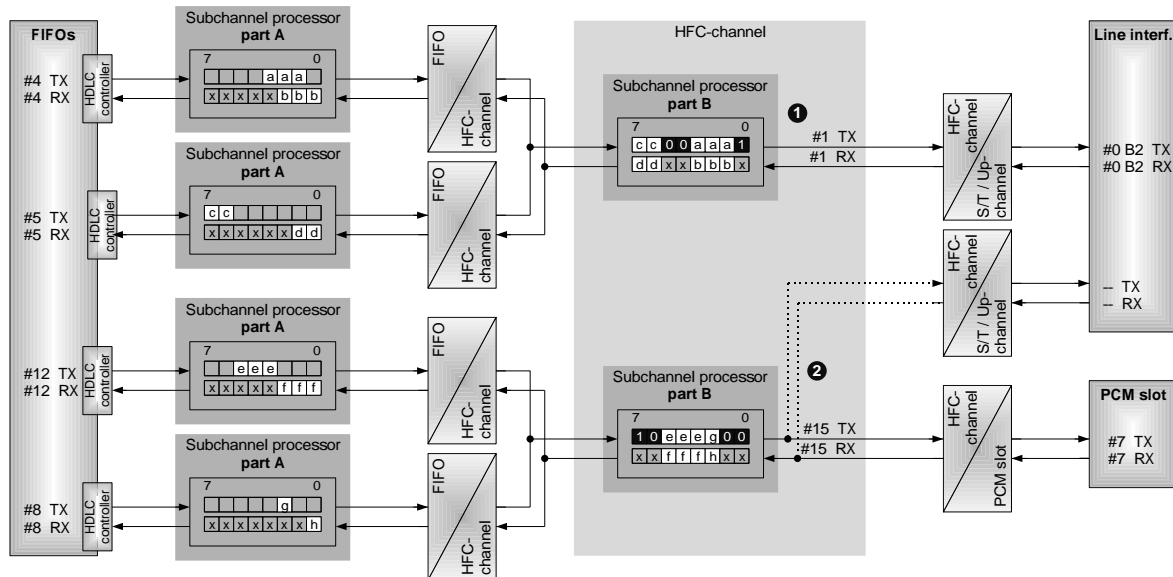


Figure 3.18: CSM example with subchannel processor

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<sub>1</sub>, the second byte is a<sub>2</sub>, and so on. FIFO[5,TX] is represented by the data bytes c<sub>1</sub>, c<sub>2</sub>, 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 <sub>p</sub> , 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 <sub>p</sub> , 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<sub>p</sub> (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 <sub>p</sub> )
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 <sub>p</sub> )
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	6	5	4	3	2	1	0
HFC-channel mask:	0	0	0	0	0	0	0	1
HFC-channel transmit byte 1:	c <sub>1</sub> [1]	c <sub>1</sub> [0]	0	0	a <sub>1</sub> [2]	a <sub>1</sub> [1]	a <sub>1</sub> [0]	1
HFC-channel transmit byte 2:	c <sub>2</sub> [1]	c <sub>2</sub> [0]	0	0	a <sub>2</sub> [2]	a <sub>2</sub> [1]	a <sub>2</sub> [0]	1
HFC-channel transmit byte 3:	c <sub>3</sub> [1]	c <sub>3</sub> [0]	0	0	a <sub>3</sub> [2]	a <sub>3</sub> [1]	a <sub>3</sub> [0]	1
...								

**② FIFO-to-PCM (TX)**

A FIFO-to-PCM configuration in HDLC mode with two FIFOs in transmit and receive direction

**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:	$d_3[1]$	$d_3[0]$	$x$	$x$	$b_3[2]$	$b_3[1]$	$b_3[0]$	$x$
...	...							

each is shown in the second example setting<sup>8</sup>.

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  $\mu$ 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.

<sup>8</sup>HDLC 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 <sub>p</sub> , 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 <sub>p</sub> , 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.

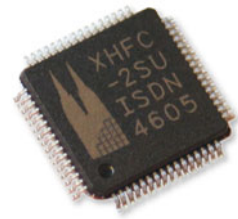
Register setup:		(CSM 2 RX)
R_FIFO	: V_FIFO_DIR = 1	(receive FIFO)
	: V_FIFO_NUM = 12	(FIFO #12)
	: V_REV = 0	(normal bit order)
A_CON_HDLC[12,RX]	: V_IFF = 0	(0x7E as inter frame fill)
	: V_HDLC_TRP = 0	(HDLC mode)
	: V_FIFO_IRQ = 7	(enable FIFO)
	: V_DATA_FLOW = '001'	(FIFO ← PCM)
A_CHANNEL[12,RX]	: 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	124	0x80	A_FIFO_DATA	135
0x0C	R_FIFO_THRES	124	0x84	A_FIFO_DATA_NOINC	135
0x0D	R_FIFO_MD	125	0xF4	A_CH_MSK	136
0x0E	A_INC_RES_FIFO	126	0xFA	A_CON_HDLC	137
0x0F	R_FIFO	127	0xFB	A_SUBCH_CFG	139
0x0F	R_FSM_IDX	127	0xFC	A_CHANNEL	140
Read only register:			0xFD	A_FIFO_SEQ	141
Address	Name	Page	0xFF	A_FIFO_CTRL	142
0x04	A_Z1	128			
0x06	A_Z2	128			
0x0C	A_F1	128			
0x0D	A_F2	129			
0x0E	A_FIFO_STA	130			
0x14	A_USAGE	131			
0x24	R_FILL_BL0	131			
0x25	R_FILL_BL1	132			
0x26	R_FILL_BL2	133			
0x27	R_FILL_BL3	134			

## 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<sub>p</sub> 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<sub>p</sub> 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<sub>p</sub> 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<sub>p</sub> 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, soft 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<sub>p</sub> interface (from a transmit FIFO) and HDLC bytes are read from the ST/U<sub>p</sub> 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 soft 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.

---

<sup>1</sup>See  $Z_{max}$  value in Table 4.2.

In addition, a hardware reset initializes the counters.

 **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 ( $\sim 1 \mu s$ ). 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 soft 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 C4IO clocks for operation. These clocks are generated from XHFC-2SU if PCM master mode is selected ( $V\_PCM\_MD = '1'$ ). When XHFC-2SU is configured to PCM slave mode ( $V\_PCM\_MD = '0'$ ) F0IO and C4IO clocks must be feed into these pins.

### 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<sub>p</sub> or the PCM bus interface.

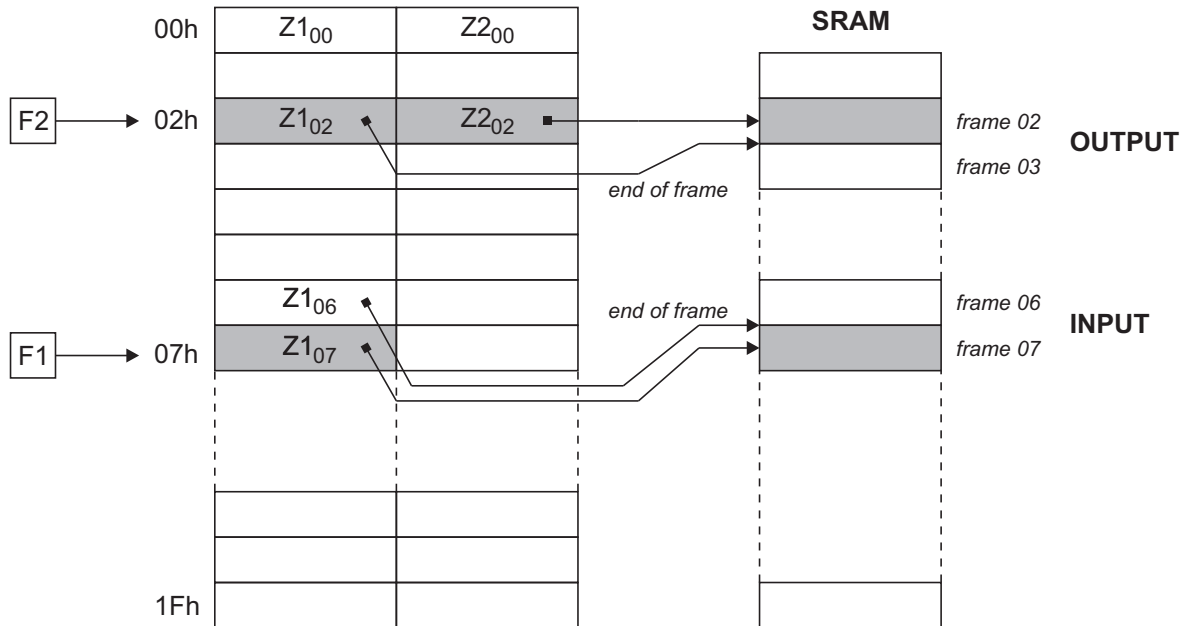


Figure 4.1: FIFO organization

XHFC-2SU checks Z1 and Z2. If  $Z1 = Z2$  (FIFO empty), XHFC-2SU generates a HDLC flag ('01111110') or continuous '1's (depending on bit V\_IFF in register A\_CON\_HDLC) and transmits it to the ST/U<sub>p</sub> interface. In this case Z2 is not incremented. If also  $F1 = F2$  only HDLC flags or continuous '1's are sent to the ST/U<sub>p</sub> 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.

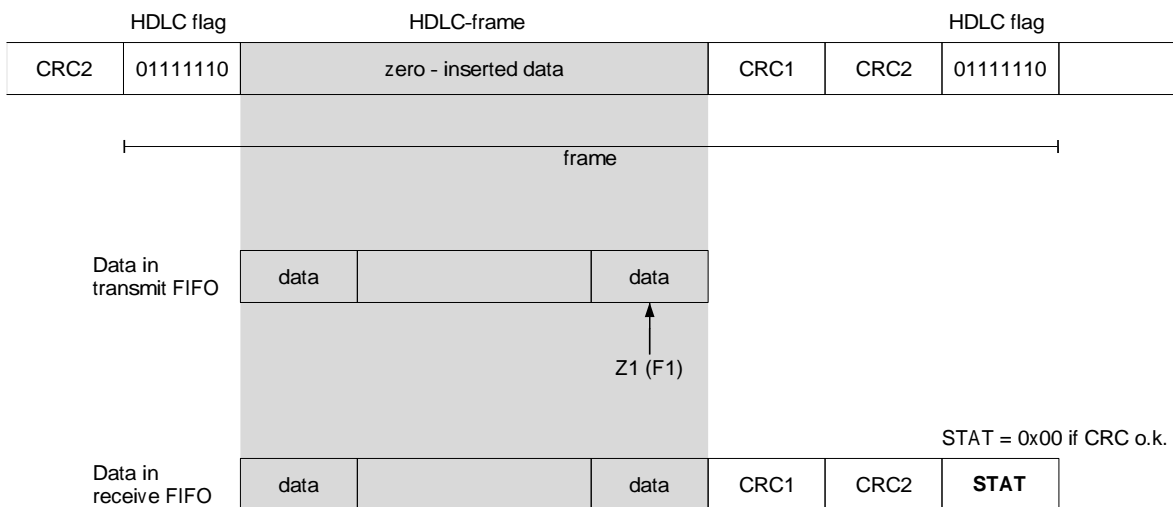


Figure 4.2: FIFO data organization in HDLC mode

#### 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<sub>p</sub> 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<sub>p</sub> 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)	0x0B	
<b>First FIFO of the FIFO sequence</b>			
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	<b>Data direction</b> 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	<b>FIFO number</b> This bitmap defines the number of the first FIFO in the FIFO sequence.
7..5	0	(reserved)	Must be '000'.

R_FIFO_THRES	(w)	0x0C	
<b>FIFO fill level control register</b>			
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	<b>Threshold for all transmit FIFOs</b> 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	<b>Threshold for all receive FIFOs</b> The threshold is a multiple of 16 bytes. 0 = 0 bytes 1 = 16 bytes 0 = 32 bytes ... 15 = 240 bytes

R_FIFO_MD	(w)	0x0D	
<b>FIFO mode configuration</b>			
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><b>FIFO mode</b> This bitmap and V_UNIDIR_MD are used to organize the FIFOs.</p> <p><b>FIFO mode with V_UNIDIR_RX = '0':</b> '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><b>FIFO mode with V_UNIDIR_RX = '1':</b> '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><b>Data flow mode selection</b> '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><b>Unidirectional FIFO data direction</b> '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. <b>Note:</b> The FIFOs of the unused data direction must not be enabled.</p>
5	0	V_UNIDIR_RX	<p><b>FIFO data direction</b> 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'.

A_INC_RES_FIFO [FIFO]		(w)	0x0E
<b>Increment and reset FIFO register</b>			
Before writing this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
0	0	V_INC_F	<b>Increment the <i>F</i>-counters of the selected FIFO</b> '0' = no increment '1' = increment This bit is automatically cleared after the counter increment has been processed.
1	0	V_RES_FIFO	<b>FIFO reset</b> '0' = no reset '1' = reset selected FIFO ( <i>F</i> - and <i>Z</i> -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	<b>LOST error bit reset</b> '0' = no reset '1' = reset LOST This bit is automatically cleared with the LOST error bit reset.
3	0	V_RES_FIFO_ERR	<b>FIFO error reset</b> '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. <b>Note:</b> 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)	0x0F	
<b>FIFO selection register</b>			
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><b>Note:</b> 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	<b>FIFO data direction</b> '0' = transmit FIFO data '1' = receive FIFO data
4..1	0	V_FIFO_NUM	<b>FIFO number</b>
6..5	0	(reserved)	Must be '00'.
7	0	V_REV	<b>Bit order</b> '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)	0x0F	
<b>Index register of the FIFO sequence</b>			
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><b>Note:</b> 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	<b>List index</b> 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)	0x04
<b>FIFO input counter Z1</b>			
Before reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		V_Z1	<b>Counter value of Z1</b> The reset value is $Z_{max}$ and depends on the FIFO configuration.

(See Table 4.2 for reset value.)

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

(See Table 4.2 for reset value.)

A_F1 [FIFO]		(r)	0x0C
<b>FIFO input HDLC frame counter F1</b>			
Before reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0	7	V_F1	<b>Counter value</b> Up to 8 HDLC frames can be stored in every FIFO.

<b>A_F2</b> [FIFO]	(r)	0x0D								
<p><b>FIFO output HDLC frame counter <i>F2</i></b></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><b>V_F2</b></td> <td><b>Counter value</b> Up to 8 HDLC frames can be stored in every FIFO.</td> </tr> </tbody> </table>			Bits	Reset value	Name	Description	7..0	7	<b>V_F2</b>	<b>Counter value</b> Up to 8 HDLC frames can be stored in every FIFO.
Bits	Reset value	Name	Description							
7..0	7	<b>V_F2</b>	<b>Counter value</b> Up to 8 HDLC frames can be stored in every FIFO.							

A_FIFO_STA [FIFO]	(r)	0x0E
<b>FIFO status register</b>		
Before reading this array register the FIFO must be selected by register R_FIFO.		
Bits	Reset value	Name
0		<b>V_FIFO_ERR</b>
		<p><b>FIFO error</b> This status bit has different meaning for transmit and receive FIFOs.</p> <p><b>Transmit FIFO:</b> 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 <math>F1 = F2</math> gets true after <math>Z2</math> increment. (2) An invalid HDLC frame has been sent when the FIFO runs empty within a frame, i.e. <math>F1 = F2</math> is already valid during data transmission and <math>Z2</math> increment cannot be executed.</p> <p>When <math>V\_FR\_ABO = '0'</math> 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 <math>V\_FR\_ABO = '1'</math> 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><b>Receive FIFO:</b> This bit is set to '1' to indicate either a FIFO overflow (<math>Z1 = Z2</math> after <math>Z1</math> increment) or a frame counter overflow (<math>F1 = F2</math> after <math>F1</math> increment).</p> <p><b>Note:</b> V_FIFO_ERR must be reset from host processor with <math>V\_RES\_FIFO\_ERR = '1'</math>.</p>
3..1		(reserved)
4		<b>V_ABO_DONE</b>
		<p><b>Frame abort done</b> 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)

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

<b>R_FILL_BL0</b>	(r)	0x24	
<b>FIFO fill level for FIFO block 0</b>			
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	<b>V_FILL_FIFO0_TX</b>	<b>FIFO[0,TX] fill level</b>
1	0	<b>V_FILL_FIFO0_RX</b>	<b>FIFO[0,RX] fill level</b>
2	0	<b>V_FILL_FIFO1_TX</b>	<b>FIFO[1,TX] fill level</b>
3	0	<b>V_FILL_FIFO1_RX</b>	<b>FIFO[1,RX] fill level</b>
4	0	<b>V_FILL_FIFO2_TX</b>	<b>FIFO[2,TX] fill level</b>
5	0	<b>V_FILL_FIFO2_RX</b>	<b>FIFO[2,RX] fill level</b>
6	0	<b>V_FILL_FIFO3_TX</b>	<b>FIFO[3,TX] fill level</b>
7	0	<b>V_FILL_FIFO3_RX</b>	<b>FIFO[3,RX] fill level</b>

<b>R_FILL_BL1</b>	(r)	0x25	
<b>FIFO fill level for FIFO block 1</b>			
<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)			0x26
<b>FIFO fill level for FIFO block 2</b>				
<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)

0x27

**FIFO fill level for FIFO block 3**

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_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**

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

<b>A_FIFO_DATA_NOINC</b> [FIFO]	(r/w)	<b>0x84</b>	
<b>FIFO data register</b>			
Before writing or reading this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
7..0		<b>V_FIFO_DATA_NOINC</b>	<b>Data byte</b> 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.)

<b>A_CH_MSK</b> [FIFO]	<b>(r*/w)</b>	<b>0xF4</b>	
<p><b>HFC-channel data mask for the selected transmit HFC-channel</b></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>			
Bits	Reset value	Name	Description
7..0	0xFF	<b>V_CH_MSK</b>	<p><b>Mask byte</b></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 <math>V\_BIT\_CNT \neq 0</math> in register A_SUBCH_CFG.</p>

(See Section 2.2.3.2 on page 40 for details on Read\* access.)

A_CON_HDLC [FIFO]		(r*/w)	0xFA
<b>HDLC and connection settings of the selected FIFO</b>			
Before writing this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
0	0	V_IFF	<b>Inter frame fill</b> '0' = write HDLC flags 0x7E as inter frame fill '1' = write all '1's as inter frame fill <b>Note:</b> For D-channel this bit must be '1'.
1	0	V_HDLC_TRP	<b>HDLC mode / transparent mode selection</b> '0' = HDLC mode '1' = transparent mode <b>Note:</b> For D-channel this bit must be '0'.
4..2	0	V_FIFO_IRQ	<b>FIFO interrupt configuration</b> This bitmap has a different meaning in HDLC and transparent mode.  <b>Transparent mode (V_HDLC_TRP = '1'):</b> 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 Z1-counter (in transmit direction) or the Z2-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  <b>HDLC mode (V_HDLC_TRP = '0'):</b> 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 <b>Note:</b> 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.  <b>Note:</b> 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><b>Data flow configuration</b></p> <p>In transmit operation (<math>V\_FIFO\_DIR = '0'</math> in register R_FIFO):</p> <p>'000', '001' = FIFO <math>\rightarrow</math> ST/U<sub>p</sub>, FIFO <math>\rightarrow</math> PCM            '010', '011' = FIFO <math>\rightarrow</math> PCM            '100', '101' = FIFO <math>\rightarrow</math> ST/U<sub>p</sub>, ST/U<sub>p</sub> <math>\rightarrow</math> PCM            '110', '111' = ST/U<sub>p</sub> <math>\rightarrow</math> PCM</p> <p>In receive operation (<math>V\_FIFO\_DIR = '1'</math> in register R_FIFO):</p> <p>'000', '100' = FIFO <math>\leftarrow</math> ST/U<sub>p</sub>            '001', '101' = FIFO <math>\leftarrow</math> PCM            '010', '110' = FIFO <math>\leftarrow</math> ST/U<sub>p</sub>, ST/U<sub>p</sub> <math>\leftarrow</math> PCM            '011', '111' = FIFO <math>\leftarrow</math> PCM, ST/U<sub>p</sub> <math>\leftarrow</math> PCM</p> <p><b>Note:</b> ST/U<sub>p</sub> <math>\leftrightarrow</math> 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<sub>p</sub> data is stored in the FIFO, interrupt generation can be used. ST/U<sub>p</sub> -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 40 for details on Read\* access.)

A_SUBCH_CFG [FIFO]		(r*/w)	0xFB
<b>Subchannel parameters for bit processing of the selected FIFO</b>			
Before writing this array register the FIFO must be selected by register R_FIFO.			
<b>Note:</b> For D- and E-channels this register must be 0x02.			
Bits	Reset value	Name	Description
2..0	0	V_BIT_CNT	<p><b>Number of bits to be processed in the HFC-channel byte</b></p> <p>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.</p> <p>'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)</p>
5..3	0	V_START_BIT	<p><b>Start bit in the HFC-channel byte</b></p> <p>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.</p>
6	0	V_LOOP_FIFO	<p><b>FIFO loop</b></p> <p>'0' = normal operation                      '1' = repeat current FIFO data (useful only in transparent mode)</p> <p><b>Note:</b> This bit is ignored for receive FIFOs.</p>
7	0	V_INV_DATA	<p><b>Inverted data</b></p> <p>'0' = normal data transmission                      '1' = inverted data transmission</p>

(See Section 2.2.3.2 on page 40 for details on Read\* access.)

A_CHANNEL [FIFO]		(r*/w)	0xFC
<b>HFC-channel assignment for the selected FIFO</b>			
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		<b>V_CH_FDIR</b>	<b>HFC-channel data direction</b> '0' = HFC-channel for transmit data '1' = HFC-channel for receive data <b>Reset value:</b> 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		<b>V_CH_FNUM</b>	<b>HFC-channel number</b> (0..15) <b>Reset value:</b> 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	0	(reserved)	Must be '000'.

(See Section 2.2.3.2 on page 40 for details on Read\* access.)

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

(See Section 2.2.3.2 on page 40 for details on Read\* access.)

A_FIFO_CTRL [FIFO]		(r*/w)	0xFF
<b>Control register for the selected FIFO</b>			
Before writing this array register the FIFO must be selected by register R_FIFO.			
Bits	Reset value	Name	Description
0	0	V_FIFO_IRQMSK	<b>Interrupt mask for the selected FIFO</b> '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.
1	0	V_BERT_EN	<b>BERT enable</b> '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	<b>Mixed interrupt generation in HDLC mode</b> This bit is only used in HDLC mode and it should be '0' in transparent mode. '0' = FIFO interrupts are generated on <i>end of frame</i> (normal operation) '1' = FIFO interrupts are generated both on <i>end of frame</i> and when the bits [n-1:0] of the Z1-counter (in transmit direction) or the Z2-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	<b>V_FR_ABO</b>	<p><b>Frame abort</b> This bit has a different meaning for transmit and receive FIFOs.</p> <p><b>Transmit FIFO (FIFO underrun indication):</b> 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 (<math>Z1 = Z2</math> and already <math>F1 = F2</math>, <math>F</math>-increment cannot be executed) Empty FIFO condition can be watched with bit V_FIFO_ERR in register A_FIFO_STA.</p> <p><b>Receive FIFO (aborted frame received):</b> 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><b>Note:</b> V_FR_ABO should be '0' in transparent mode.</p>
4	0	<b>V_NO_CRC</b>	<p><b>Suppress CRC transmission</b> '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	<b>V_NO_REP</b>	<p><b>No automatic repetition on HDLC frames (D-channel)</b> Frame is deleted after D-channel contention.</p>
7..6	0	(reserved)	Must be '00'.

(See Section 2.2.3.2 on page 40 for details on Read\* access.)





## Chapter 5

# Universal ISDN Port

**Table 5.1:** Overview of the ST/U<sub>p</sub> interface registers

Write only registers:			Read only registers:		
Address	Name	Page	Address	Name	Page
0x16	R_SU_SEL	183	0x13	R_AF0_OVIEW	199
0x30	A_SU_WR_STA	184	0x30	A_SU_RD_STA	200
0x31	A_SU_CTRL0	185	0x31	A_SU_DLYL	201
0x32	A_SU_CTRL1	187	0x32	A_SU_DLYH	202
0x33	A_SU_CTRL2	188	0x34	A_MS_RX	203
0x34	A_MS_TX	190	0x35	A_SU_STA	204
0x35	A_ST_CTRL3	191	0x3C	A_B1_RX	205
0x35	A_UP_CTRL3	192	0x3D	A_B2_RX	205
0x36	A_MS_DF	194	0x3E	A_D_RX	206
0x37	A_SU_CLK_DLY	195	0x3F	A_E_RX	207
0x3C	A_B1_TX	196			
0x3D	A_B2_TX	196			
0x3E	A_D_TX	197			
0x3F	A_BAC_S_TX	198			

## 5.1 General overview of the S/T and U<sub>p</sub> interfaces

The Universal ISDN Port consists of a line interface with both S/T and U<sub>p</sub> 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<sub>p</sub> 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<sub>p</sub> 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<sub>SU\_SEL</sub> of register R<sub>SU\_SEL</sub>.
  - XHFC-2S4U: S/T interfaces 0..1 available, U<sub>p</sub> interfaces 0..3 available
  - XHFC-4SU: S/T interfaces 0..3 available, U<sub>p</sub> 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<sub>ST\_SEL</sub> = '0' in register A<sub>ST\_CTRL3</sub>. The line interface can be switched into U<sub>p</sub> mode with V<sub>ST\_SEL</sub> = '1'<sup>1</sup>. 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<sub>p</sub>) 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<sub>p</sub> 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.

<sup>1</sup>Please 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<sub>ST\_CTRL3</sub> is used while a line interface in U<sub>p</sub> mode uses A<sub>UP\_CTRL3</sub>.

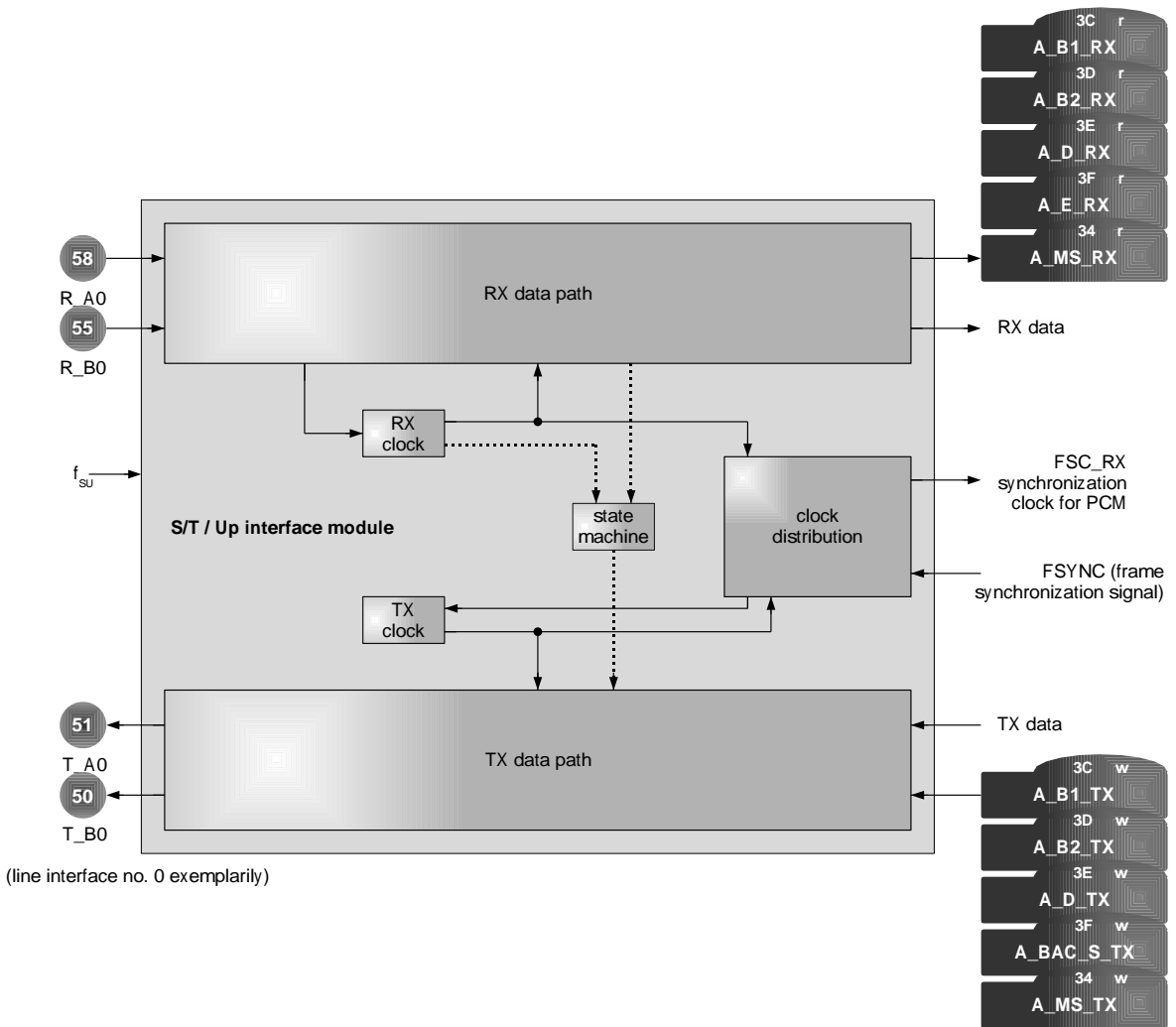


Figure 5.1: Overview of the Universal ISDN Port module

## 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 [8] 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.

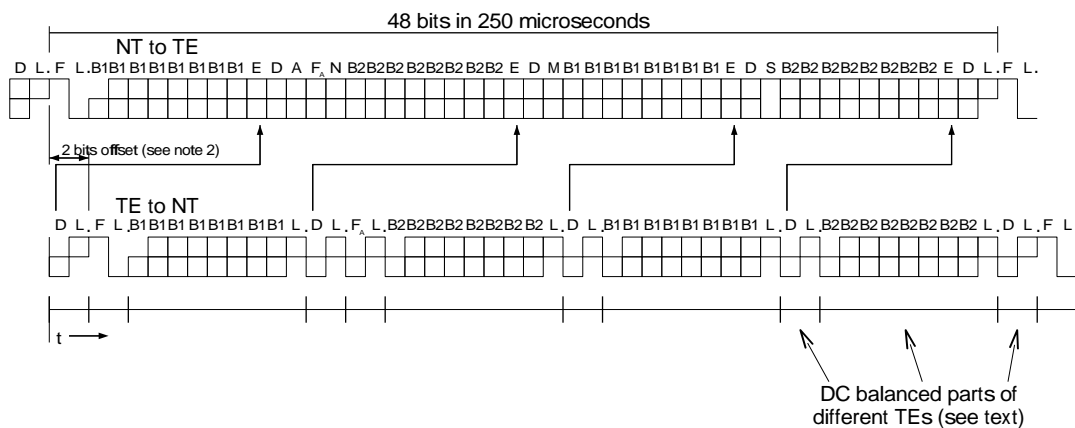


Figure 5.2: Frame structure at reference points S and T (see legend in Table 5.2 and specification [8])

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  $\mu$ 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 [8].

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

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 <sub>A</sub> -bit	M-bit	Q-bits in F <sub>A</sub> bit position <sup>*1</sup>	S-bits <sup>*2</sup>
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'
...				

<sup>\*1</sup>: 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<sub>A</sub> bits).

<sup>\*2</sup>: The specification [8] defines five subchannels for the S-multiframe. Only subchannel 1 is used from XHFC-2SU. All other subchannels are set to '0'.

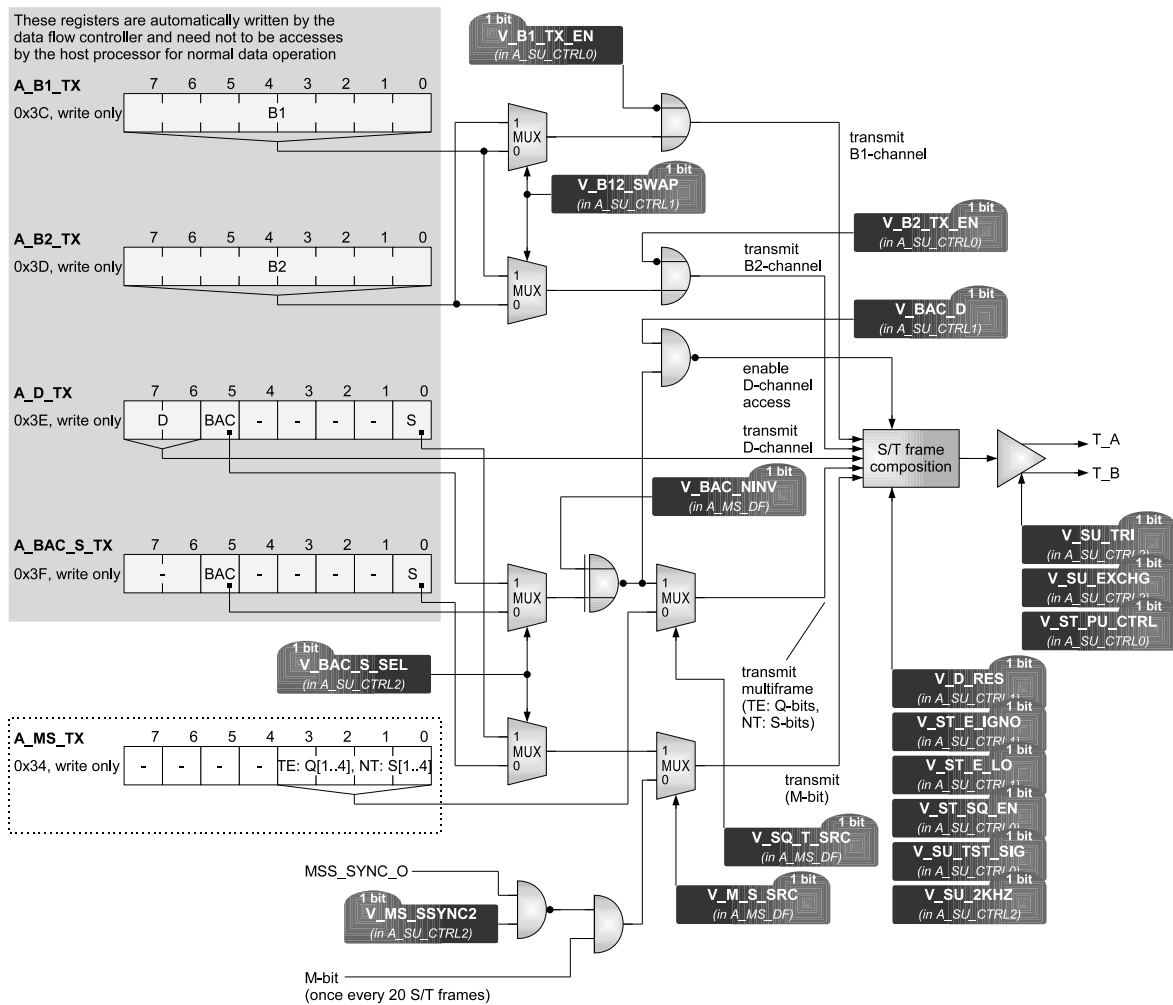


Figure 5.3: S/T frame composition for B1-, B2-, D- and multiframe bits (S/T interface mode, transmit direction)

A\_E\_RX) or manually from the application software with registers A\_MS\_TX and A\_MS\_RX.

### 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 follows<sup>2</sup>:

**INFO 0:** No signal on line.

**INFO 1:** Continuous signal at nominal bit rate with a '00111111' 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<sub>A</sub>-, N- and L-bits are set according to the normal coding rule.

<sup>2</sup>Please see [8] for a detailed description of the INFO signals.

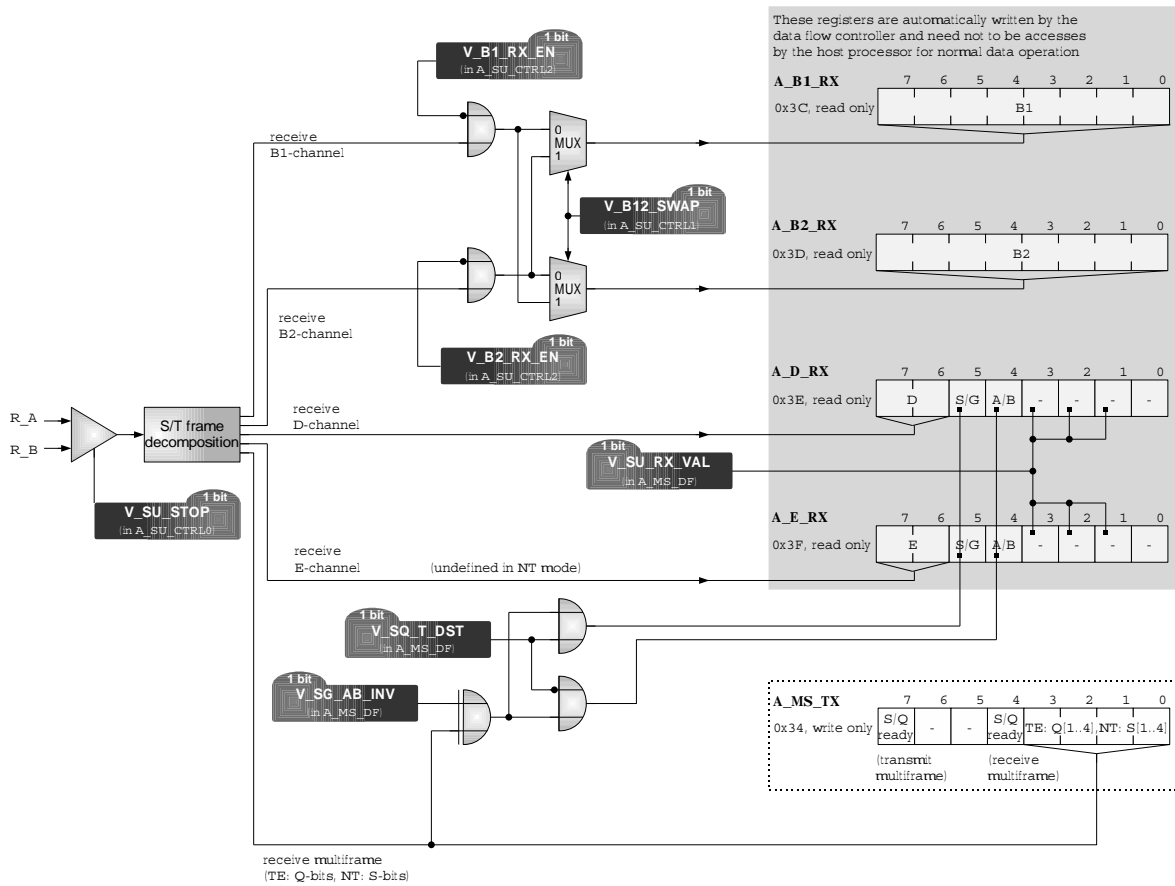


Figure 5.4: S/T frame decomposition for B1-, B2-, D-, E-channel and multiframe bits (S/T interface mode, receive direction)

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.

### 5.2.6 State machine

A specification conform state machine for TE and NT mode is implemented [8]. 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.<sup>3</sup>

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.

<sup>3</sup>It 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
<b>Event:</b>					
State machine release <sup>*3</sup>	G 1				
Activate request	start T 1 <sup>*1</sup> G 2	start T 1 <sup>*1</sup> G 2			start T 1 <sup>*1</sup> G 2
Deactivate request	—		start T 2 G 4	start T 2 G 4	
Expiry T 1 <sup>*1</sup>	—	—	start T 2 G 4	/	—
Expiry T 2 <sup>*2</sup>	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1	—	start T 1 <sup>*1</sup> G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 <sup>*1,4</sup> G 3	—	—
Lost framing	—	/	/	G 2	—
<b>Legend:</b>	—	No state change			
	/	Impossible situation			
		Impossible by the definition of the layer 1 service			

<sup>\*1</sup>: Timer T 1 is not implemented and must be implemented in software.

<sup>\*2</sup>: 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.

<sup>\*3</sup>: After reset the state machine is fixed to G 0.

<sup>\*4</sup>: 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
<b>Event:</b>								
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 219).

**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 224.

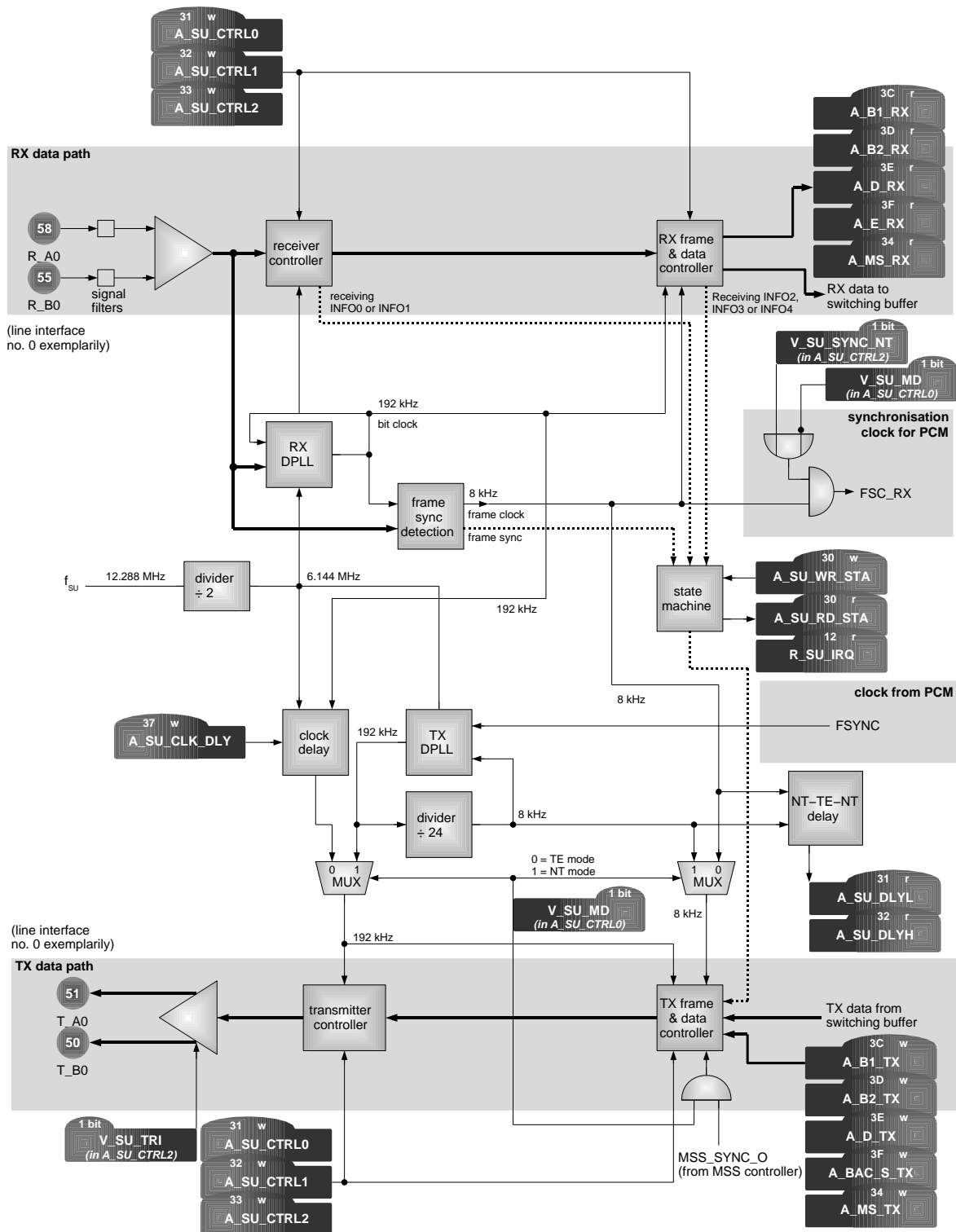


Figure 5.5: S/T clock synchronization

### 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 .

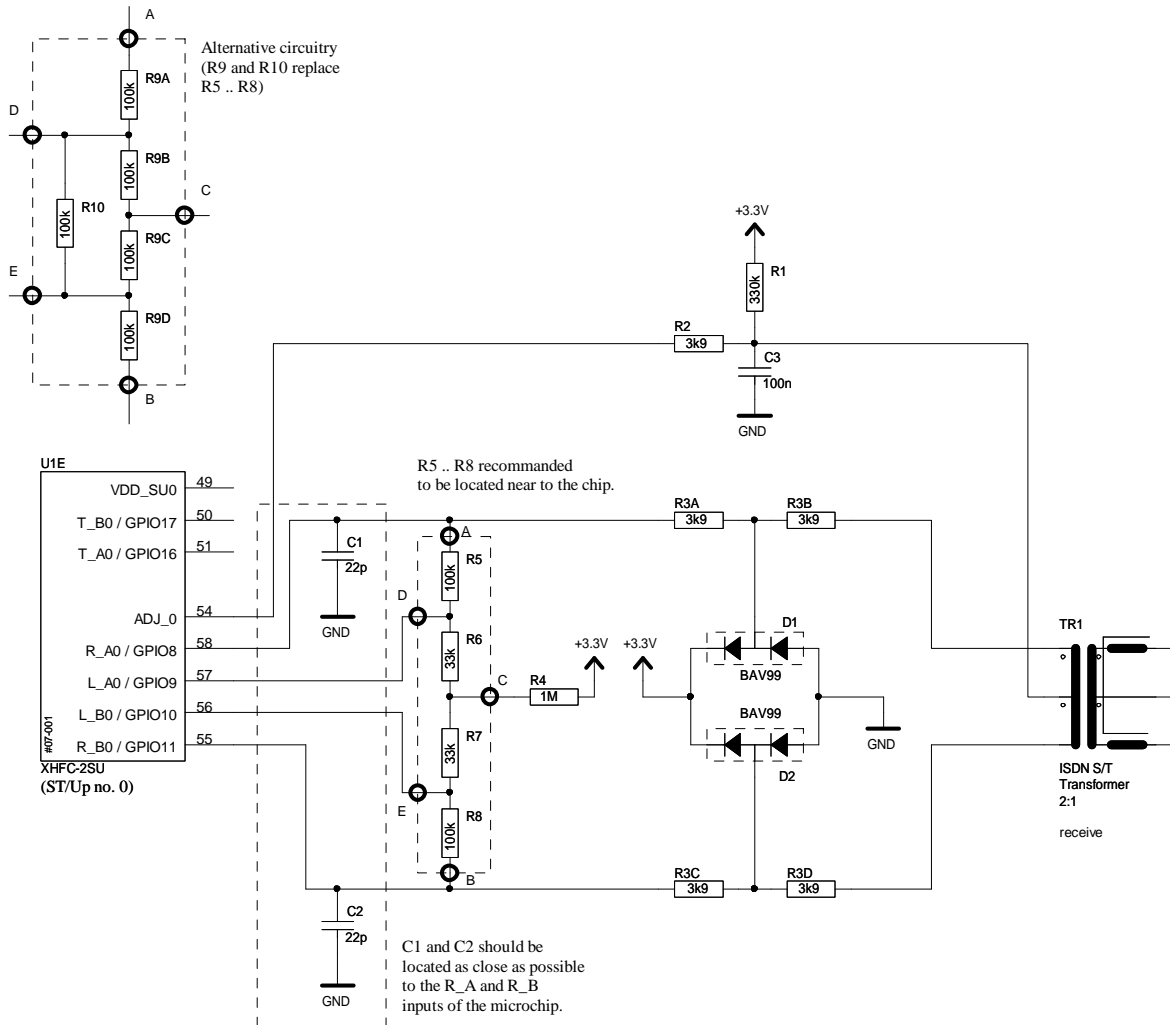


Figure 5.6: External S/T receive circuitry for TE and NT mode

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.

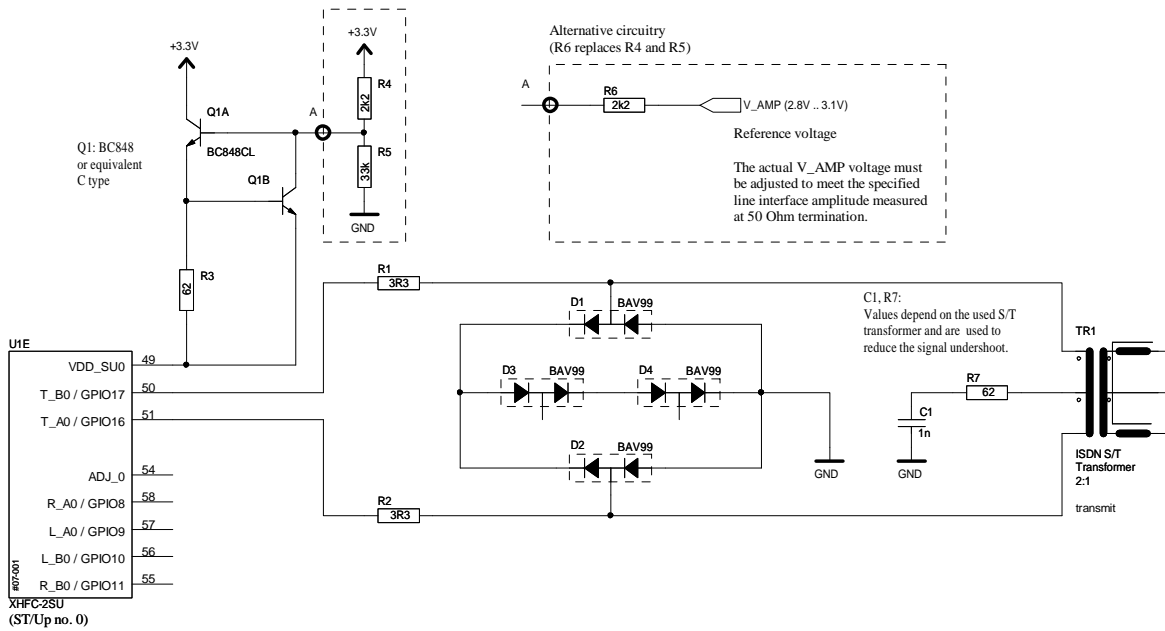


Figure 5.7: External S/T transmit circuitry for TE and NT mode

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 mode<sup>4</sup>. The termination resistors R1 and R2 are optional.

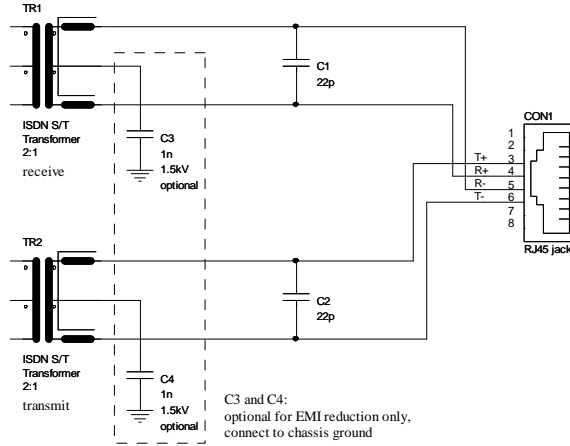


Figure 5.8: Transformer and connector circuitry in TE mode

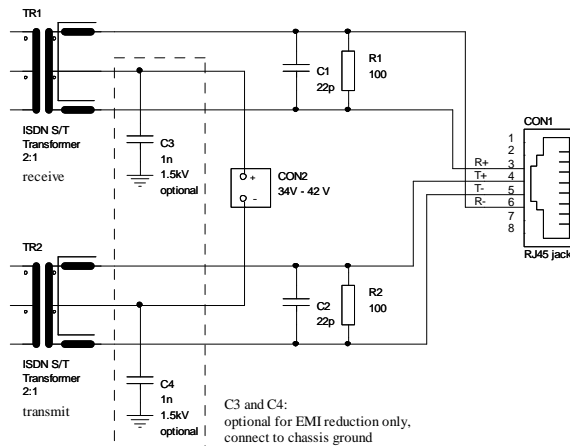
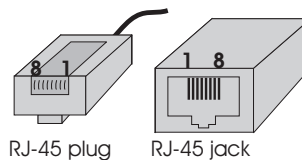


Figure 5.9: Transformer and connector circuitry in NT mode (shown with optional 100Ω termination, whole bus termination must be 50Ω)

<sup>4</sup>The 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**

<b>Bel Fuse Inc., United Kingdom, <a href="http://www.belfuse.com">http://www.belfuse.com</a></b>			
<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
	2798C (SMD)		
	2798D (SMD)	Hybrid Transformer Module with choke:	
			S803-6200-01
<b>Pulse Engineering, Inc., United States, <a href="http://www.pulseeng.com">http://www.pulseeng.com</a></b>			
<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
	T5033 (SMD)		PE-65499
	T5035 (3 kV)		PE-65795 (SMD)
	T5036 (SMD, 3 kV)		PE-65799 (SMD)
	PE-64995	Dual Transformer Module with choke:	
	PE-64999		T5012
	PE-68992		T5034 (SMD)
	ST-5069 (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		HVM-140C1
	ISV-140B1 (3 kV)		ISM-140C1
	ISHF-240B1 (3 kV)		MUJ-103A-500(SMD)
	SHJ-240B (SMD, 3 kV)		MUJ-103A-101(SMD)
	SMJ-140B (SMD)		MUJ-103A-501(SMD)
	SWJ-140B (SMD)		MUJ-103A-502(SMD)
Dual Transformer Module without choke:			MAJ-403A-470 (SMD)
	MUJ-103A-000 (SMD)		MAJ-403A-101 (SMD)
	MAJ-403-000 (SMD)		MAJ-403A-501 (SMD)
	MSJ-403A-000 (SMD)		MAJ-403A-502 (SMD)
	MHJ-240B1-000 (SMD, 3 kV)		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		UT 20495-TS (SMD)
	UT 20999		UT 20499-00TS (SMD)
	UT 21023		UT 20765-00 (SMD, 3 kV)
	UT 21595		UT 20795-00TS (SMD)
	UT 28166		UT 21624
	UT 28166-TS (SMD)		UT 21624 TS (SMD)
	UT 28428-TS (SMD)		
	UT 28729 (4 kV)	Dual Transformer Module with choke:	
			UT 20495 CV-TS (SMD)
			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 28624
			UT 28624A
			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		7-M4035-X001
	3-L4025-X095		7-M5014-X001 (SMD)
	3-L4031-X001		7-M5026-X001 (SMD)
	3-L4097-X029 (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
			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)

---

Vogt electronic AG, Germany, <http://www.vogt-electronic.com>

<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 (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)
		Dual Transformer Module with choke:	
			503 16 017 00 (SMD)
			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<sub>p</sub> interface description

#### 5.3.1 Overview

The Universal ISDN Port is able to provide a ping-pong type 2-wire interface according to U<sub>p0</sub> and U<sub>pN</sub> 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<sub>p</sub> 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 153).

The U<sub>p</sub> 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<sub>p</sub> frame structure has a length of 250 μs. Within this period, a transmit phase and a receive phase with 99 μs 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.

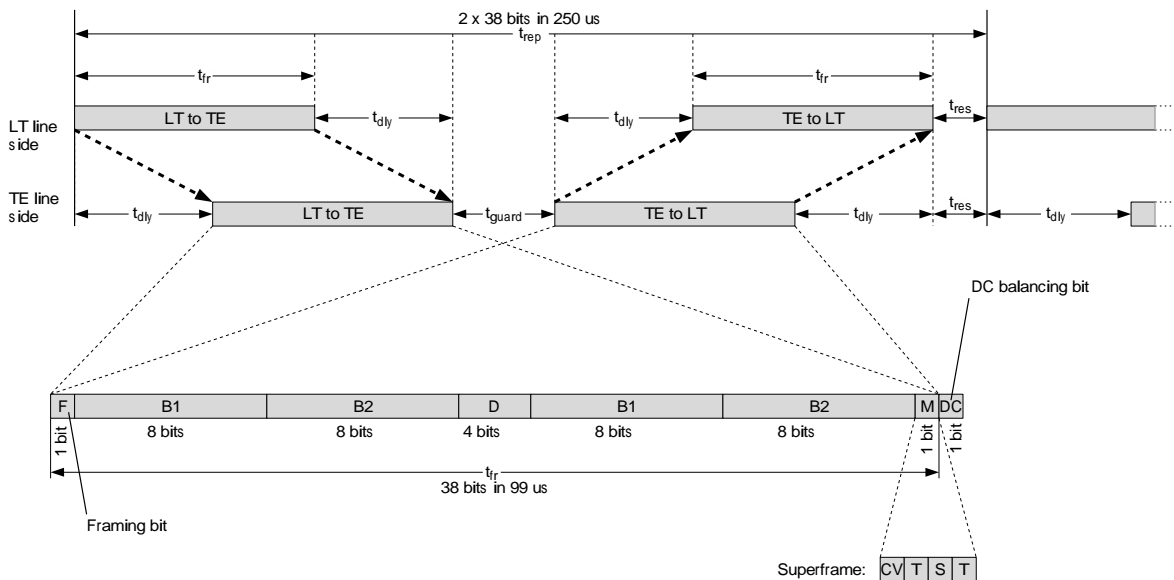


Figure 5.10: U<sub>p</sub> interface frame structure

#### 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

**Table 5.7:** Legend for Figure 5.10

Symbol	min / $\mu$ s	typ / $\mu$ s	max / $\mu$ 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

one superframe bit within every frame, a complete superframe is transmitted and received every 1 ms. 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

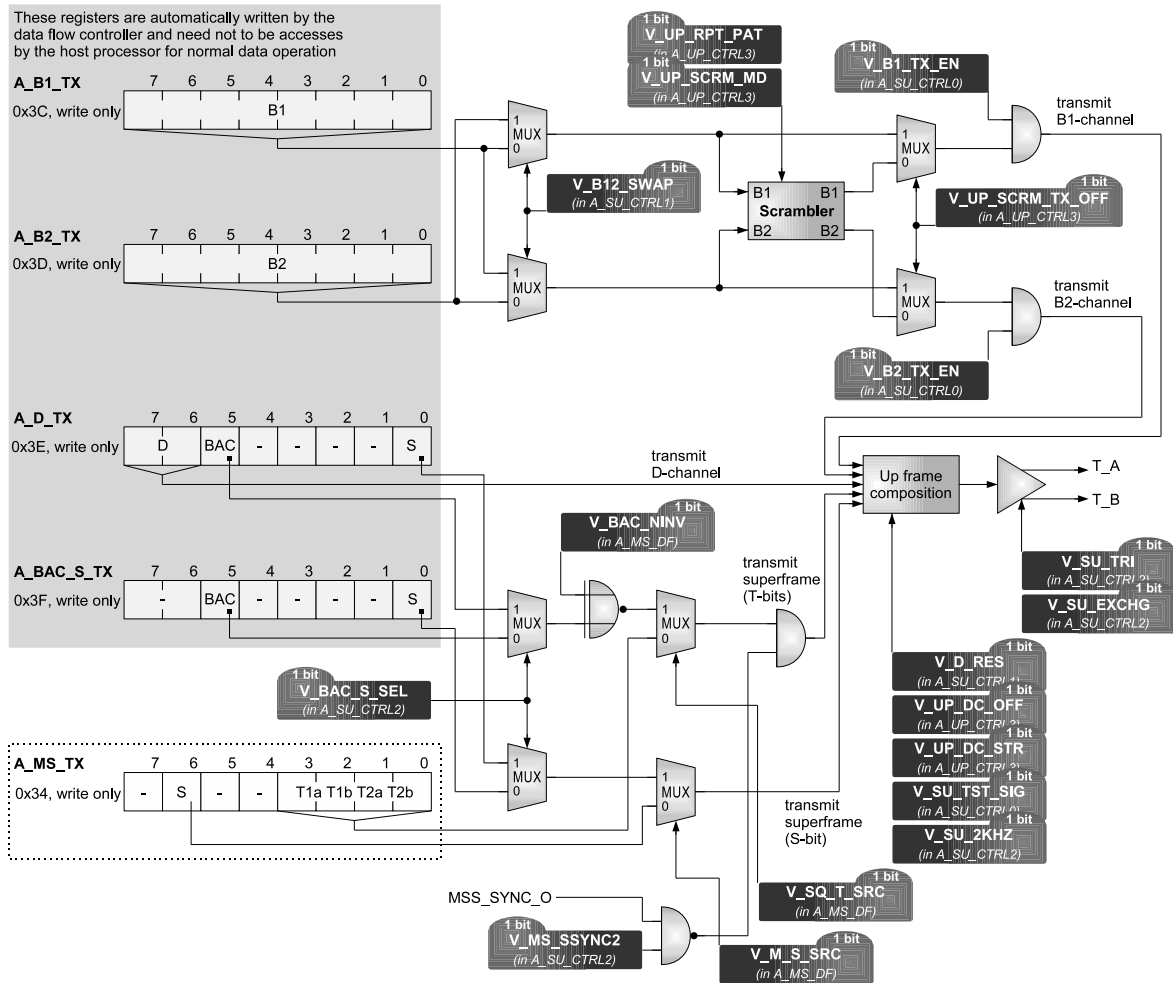
### 5.3.4 Data transmission

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.



**Figure 5.11:** U<sub>p</sub> frame composition for B1-, B2-, D-channel and superframe bits (U<sub>p</sub> interface mode, transmit direction)

Figures 5.11 and 5.12 show the composition and decomposition of the U<sub>p</sub> 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<sub>p</sub> interface and receive data from the U<sub>p</sub> 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<sup>5</sup> 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

<sup>5</sup>Please note, that A\_E\_RX does only contain E-channel bits in S/T interface mode.

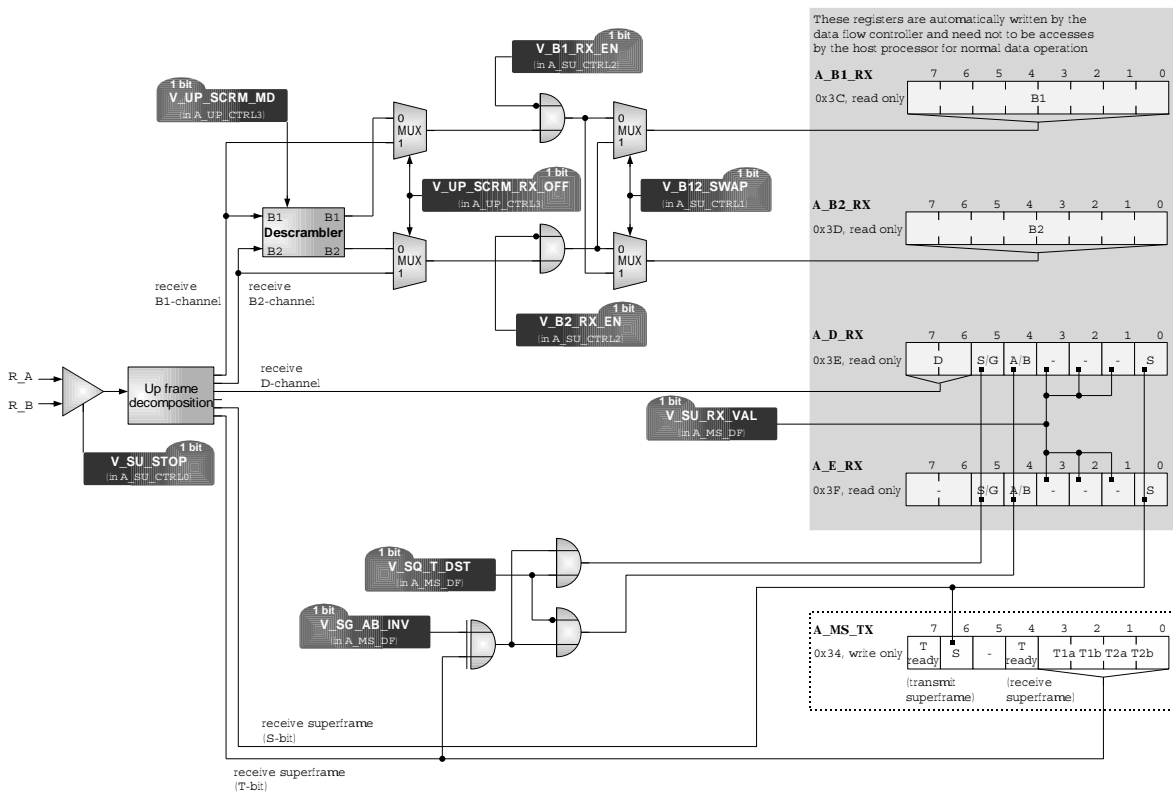


Figure 5.12: U<sub>p</sub> frame decomposition for B1-, B2-, D-channel and superframe bits (U<sub>p</sub> interface mode, receive direction)

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<sub>p</sub> interface must be set. The default value is 0xF for TE mode.

Tables 5.9 and 5.10 show the U<sub>p</sub> 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 [8].

Table 5.9: U<sub>p</sub> 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
<b>Event:</b>					
State machine release <sup>*3</sup>	G 1				
Activate request	start T 1 <sup>*1</sup> G 2	start T 1 <sup>*1</sup> G 2			start T 1 <sup>*1</sup> G 2
Deactivate request	—		start T 2 G 4	start T 2 G 4	
Expiry T 1 <sup>*1</sup>	—	—	start T 2 G 4	/	—
Expiry T 2 <sup>*2</sup>	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1 or INFO 1W	—	start T 1 <sup>*1</sup> G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 <sup>*1,4</sup> G 3	—	—
Lost framing	—	/	/	G 2	—
<b>Legend:</b>	—	No state change			
	/	Impossible situation			
		Impossible by the definition of the layer 1 service			

<sup>\*1</sup>: Timer T 1 is not implemented and must be implemented in software.

<sup>\*2</sup>: 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.

<sup>\*3</sup>: After reset the state machine is fixed to G 0.

<sup>\*4</sup>: 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<sub>p</sub> 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
<b>Event:</b>								
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<sub>p</sub> state machine is stuck at F0 or G0 after a reset. The interface sends no signal on the U<sub>p</sub> line and is not able to activate it by incoming INFO x in this state. Writing '0' into bit V<sub>SU\_LD\_STA</sub> of register A<sub>SU\_WR\_STA</sub> 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<sub>G2\_G3</sub> in register A<sub>SU\_RD\_STA</sub> or it can permanently be activated by setting bit V<sub>G2\_G3\_EN</sub> in register A<sub>SU\_CTRL1</sub>.

### 5.3.7 Clock synchronization

A detailed view inside the line interface block diagram of Figure 5.1 is shown for the U<sub>p</sub> 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<sub>p</sub> frame.

Received data from the pins R<sub>A0</sub> / R<sub>A1</sub> and R<sub>B0</sub> / R<sub>B1</sub> 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<sub>SL\_SEL7</sub> and Figure 6.5 on page 219).

**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<sub>UP\_SCRM\_RX\_OFF</sub> = '1' (for the receive data path) and V<sub>UP\_SCRM\_TX\_OFF</sub> = '1' (for the transmit data path) in register A<sub>UP\_CTRL3</sub>. The scrambler mode is configurable according to ITU-T V.27 specification [7] or OCTAT-P compatibility [5] with V<sub>UP\_SCRM\_MD</sub> in the same register.

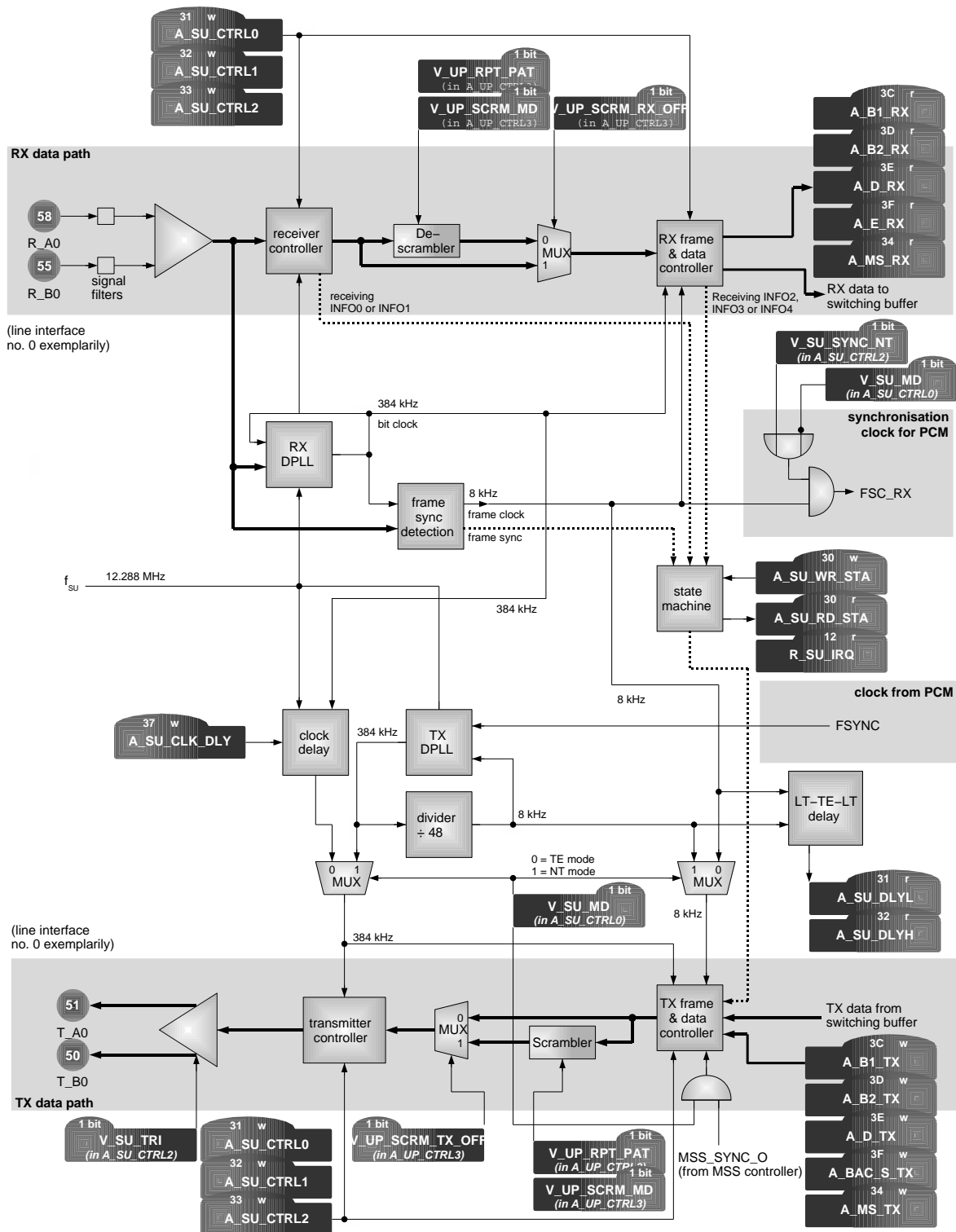


Figure 5.13: U<sub>p</sub> clock synchronization

### 5.3.8 External circuitry

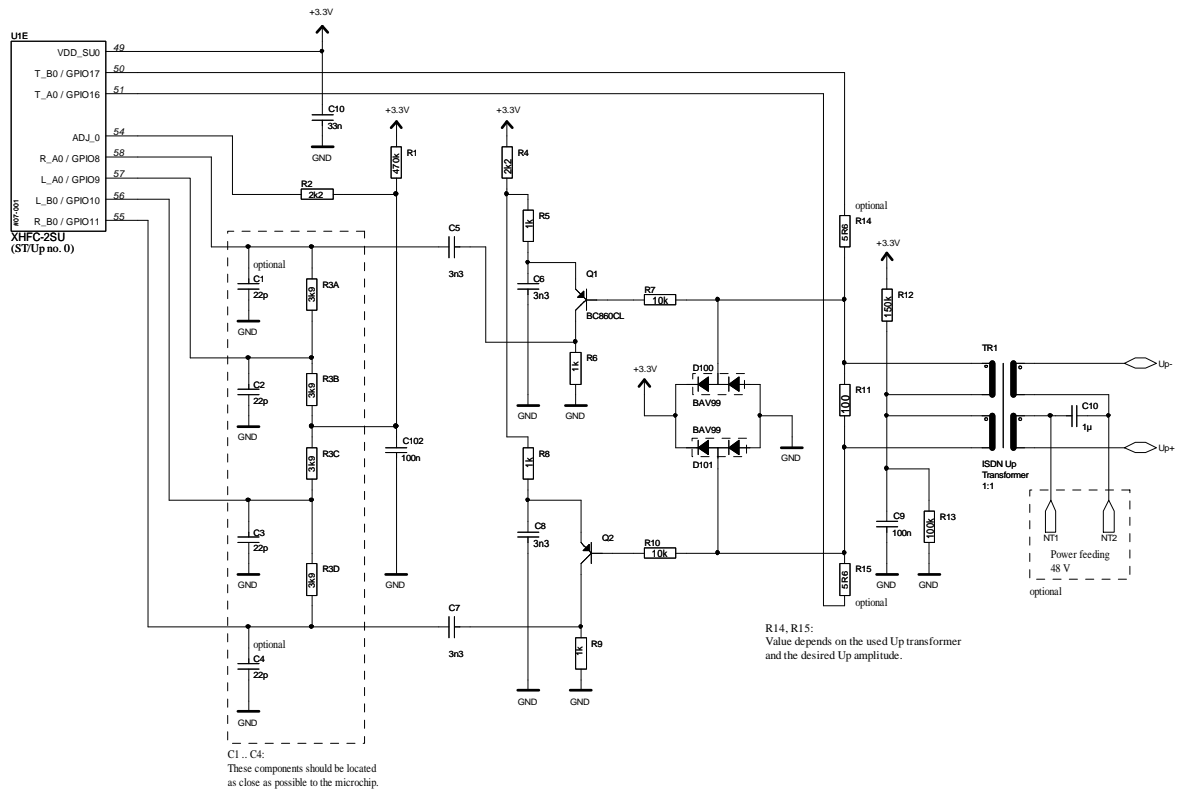


Figure 5.14: External U<sub>p</sub> circuitry for TE and LT mode

### 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*

<b>UMEC GmbH, Germany, Taiwan, United States, <a href="http://www.umec.de">http://www.umec.de</a></b>	
<b>Type</b>	<b>Device</b>
Single transformer:	UT 21434A-TS (SMD)
<b>Vacuumschmelze GmbH &amp; Co. KG, Germany, <a href="http://www.vacuumschmelze.com">http://www.vacuumschmelze.com</a></b>	
<b>Type</b>	<b>Device</b>
Single transformer:	3-M5024-X008 (SMD)
<b>Vogt electronic AG, Germany, <a href="http://www.vogt-electronic.com">http://www.vogt-electronic.com</a></b>	
<b>Type</b>	<b>Device</b>
Single transformer	503 10 903 00 (SMD)
<b>Please note:</b> 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<sub>p</sub> 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.

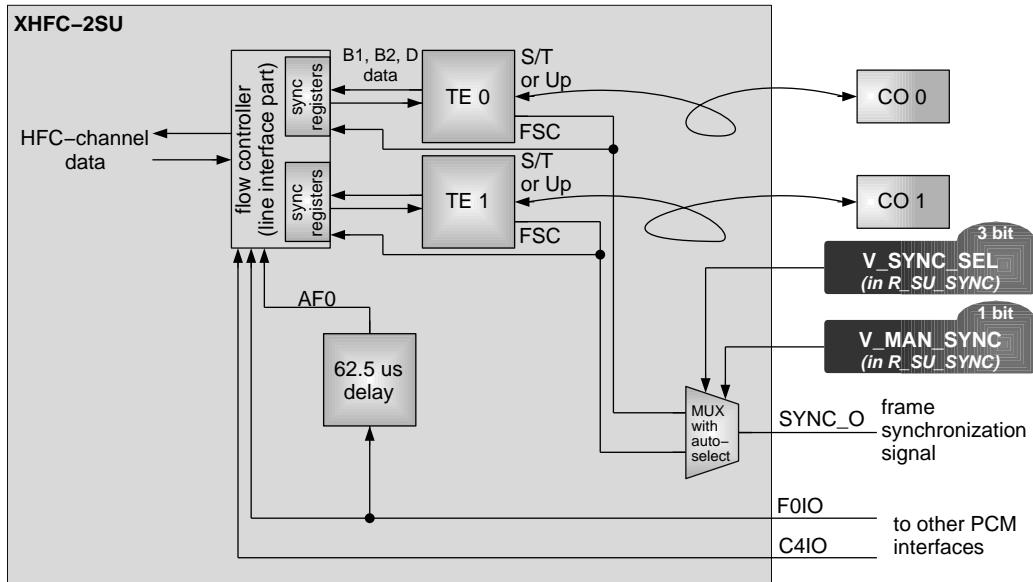
### Interrupt capability

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.



TE: Terminal Equipment (S/T or Up interface in TE mode)  
CO: Central office

**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 and F0 is smaller than approximately 25  $\mu$ s (guard window). So the phase offset between FSC and F0 is always 25  $\mu$ s . . 100  $\mu$ 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<sub>0</sub> and F0IO is 62.5  $\mu$ 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  $\mu$ s.

Figure 5.17 is divided into three parts. The upper and lower part show the line interface oriented

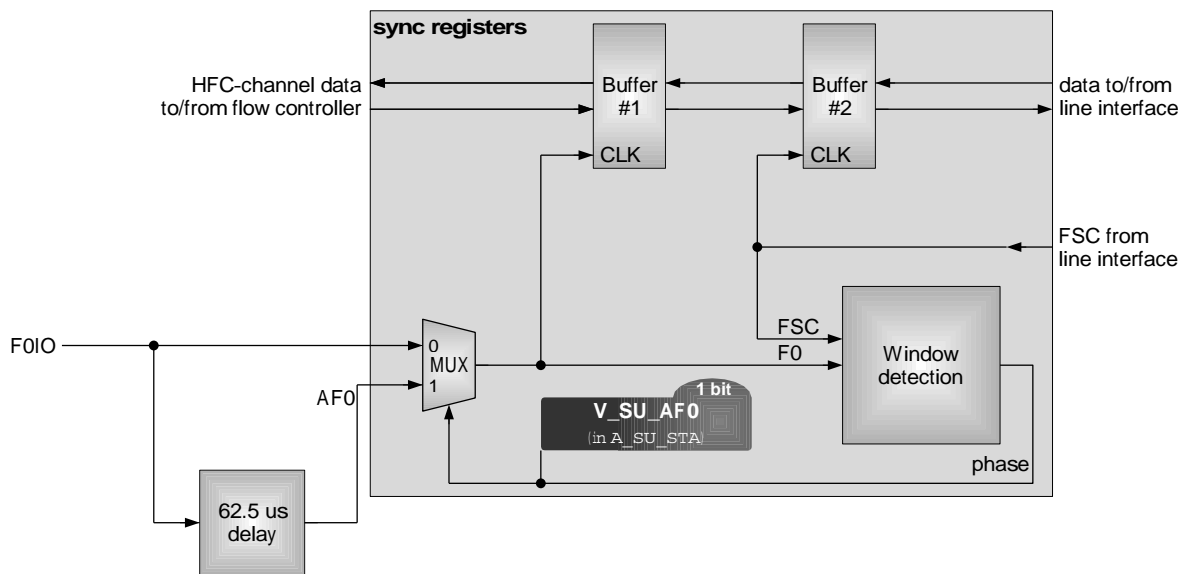
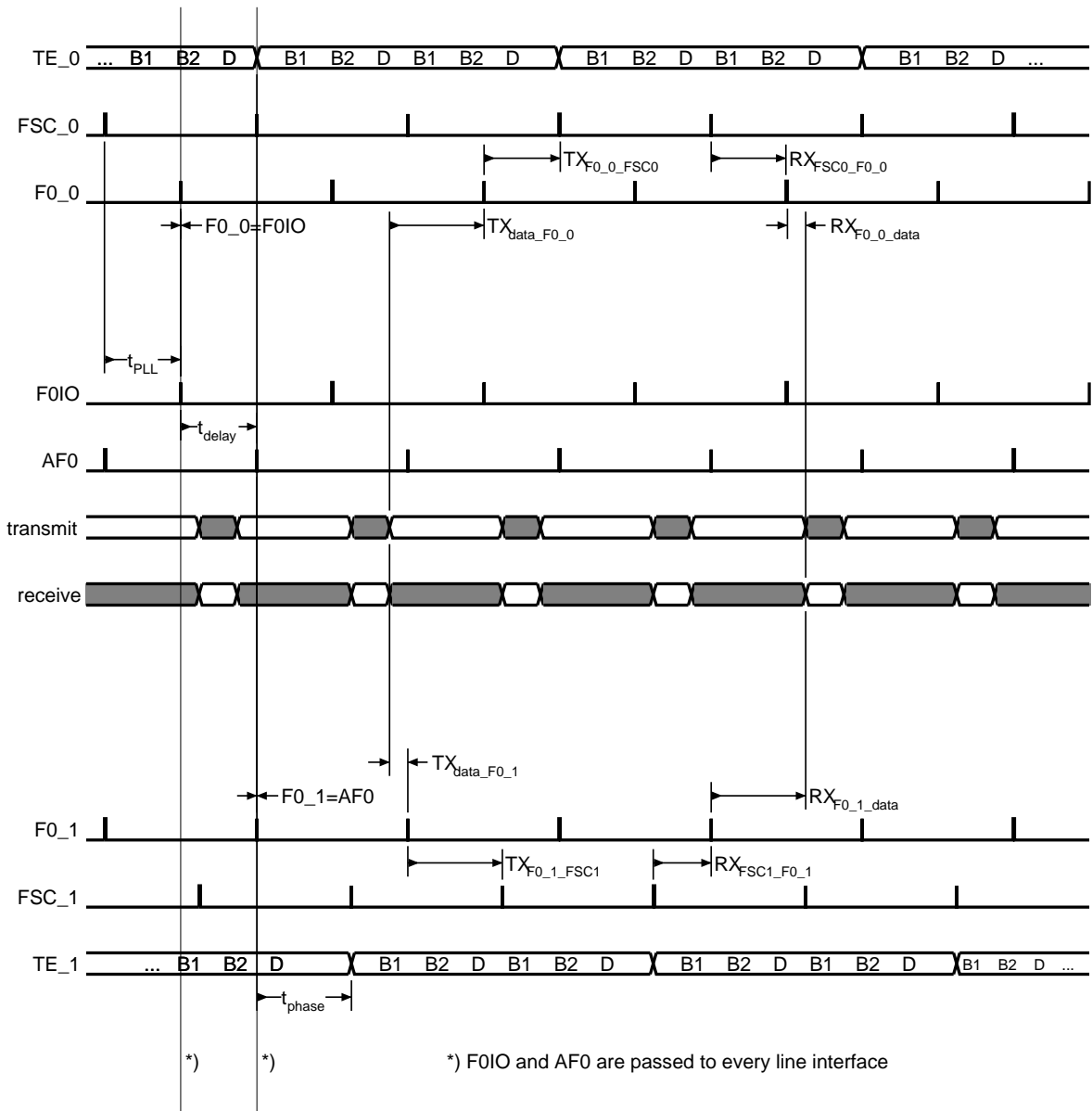


Figure 5.16: Synchronization registers (detail of Figure 5.15)

Table 5.12: Symbols of Figures 5.17

Symbol	Characteristic
$t_{PLL}$	PLL-generated frame pulse offset between FSC and F0IO (62.5 $\mu$ s)
$t_{delay}$	Frame pulse delay between F0IO and AFO (62.5 $\mu$ 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_0 pulse
$RX_{FSC0\_F0\_0}$	Time from FSC_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_1 pulse
$RX_{FSC1\_F0\_1}$	Time from FSC_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)

signals of TE 0 and TE 1 respectively. In the middle part, HFC-channel oriented signals are shown 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 TE 0 is the synchronization source in this example, FSC\_0 is the reference signal for the PLL to generate the F0IO signal. A data transmission from TE 0 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\_0 pulse.

In receive direction, the incoming data byte is stored in buffer #2 first with the FSC\_0 pulse. After  $RX_{FSC_0\_F0_0}$ , buffer #1 takes the data byte where it becomes valid for the HFC-channel.

TE 1 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<sub>p</sub> interface is controlled either by F0IO or by AF0. In this example  $F0_1 = AF0$  is shown.  $TX_{F0_1\_FSC_1}$  after F0\_1, the FSC\_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\_1 pulse. After  $RX_{FSC_1\_F0_1}$  buffer #1 takes the data byte and it receives the HFC-channel.

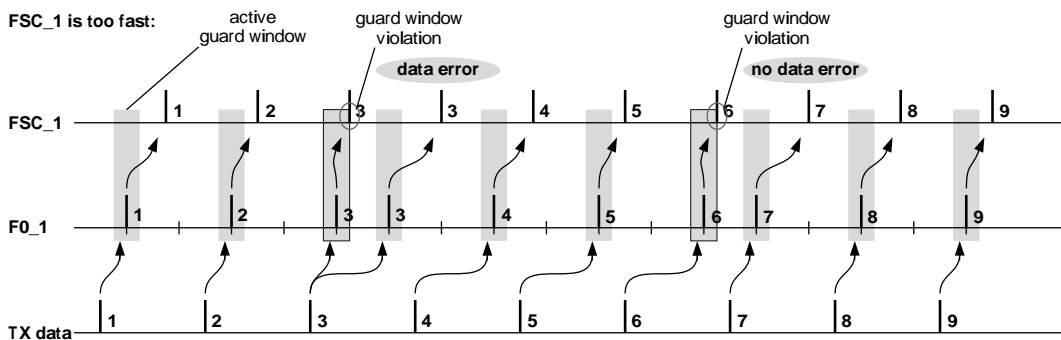
For the TE which acts as synchronization source, the clock pulses of buffer #1 and #2 have always a  $62.5 \mu s$  delay. Unsynchronized ST/U<sub>p</sub> interfaces have clock pulses of buffer #1 and #2 that are delayed  $25 \mu s \dots 100 \mu s$ . The value depends on the phase offset  $t_{phase}$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

**Timing with frequency drift**

When there is a frequency drift between FSC\_0 and FSC\_1, the window detection block changes it's 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  $f_{FSC\_1} > f_{FSC\_0}$  in transmit data direction. FSC\_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\_1 is too fast which leads to a *byte doubling* in case of transmission error.

Every time, when the detection window reaches the FSC\_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  $f_{FSC\_1} > f_{FSC\_0}$  (i.e. too fast FSC from unsynchronized TE)

Figure 5.19 shows the synchronization process for  $f_{FSC\_1} < f_{FSC\_0}$  in transmit data direction. Again,  $FSC\_0$  is assumed to be the synchronization signal. In this case  $FSC\_1$  is too slow which leads to a *byte skip* in case of transmission error.

Every time, when the detection window reaches the  $FSC\_1$  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  $FSC\_1$  leads to *byte skip* errors and a too slow  $FSC\_1$  causes *byte doubling* errors from time to time.

The time between two errors is given by

$$T_{\text{error}} = \frac{1}{f_{FSC\_0}} \cdot \frac{1 + \Delta f_{\text{rel}}}{\Delta f_{\text{rel}}} \approx \frac{125 \mu\text{s}}{\Delta f_{\text{rel}}} \text{ for } \Delta f_{\text{rel}} \ll 1$$

with the precise frame clock  $f_{FSC\_0} = 8 \text{ kHz}$  and the relative frequency error

$$\Delta f_{\text{rel}} = \frac{|f_{FSC\_1} - f_{FSC\_0}|}{f_{FSC\_0}}$$

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

### 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.

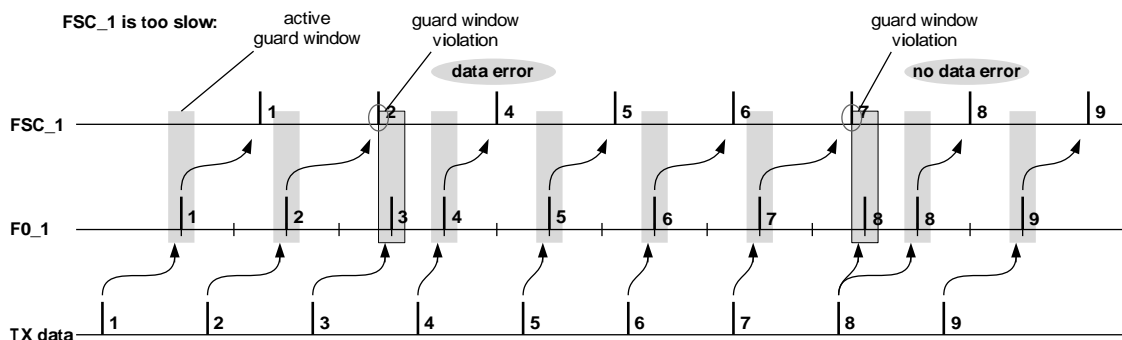


Figure 5.19: Data transmission with  $f_{FSC\_1} < f_{FSC\_0}$  (i.e. too slow  $FSC$  from unsynchronized TE)

### 5.4.3 Combined S/T and U<sub>p</sub> circuitry

An external circuitry of the Universal ISDN Port can be set up which can be used for both S/T and U<sub>p</sub> interface mode. This circuitry requires only one S/T transformer module which is used for U<sub>p</sub> 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<sub>p</sub> interface mode. SU means 'ST/U<sub>p</sub>'.

The name fragments ST or UP are used when a register or bitmap is either valid in S/T or U<sub>p</sub> interface mode.

### 5.5.1 Write only registers

R_SU_SEL		(w)	0x16
<b>S/T or Up interface selection register</b>			
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		<b>V_SU_SEL</b>	<b>Single line interface selection</b> '000' = ST/Up interface 0 '001' = ST/Up interface 1 '010' = ST/Up interface 2 '011' = ST/Up interface 3
2		(reserved)	
3		<b>V_MULT_SU</b>	<b>multiple line interface selection</b> 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)	0x30
<b>ST/Up state machine register</b>			
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	<b>Binary value of the new state</b> (NT/LT: Gx, TE: Fx) V_SU_LD_STA must also be set to load the state.
4	1	V_SU_LD_STA	<b>Load the new state</b> '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 $\mu$ s and must be cleared by software. '0' = enable the automatic state machine (V_SU_SET_STA is ignored). <b>Note:</b> After writing an invalid state, the state machine goes to deactivated state (G1, F2).
6..5	0	V_SU_ACT	<b>Start activation/ deactivation</b> '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	<b>Allow G2 to G3 transition</b> '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
<b>A_SU_CTRL0</b> [ST/Up] (w) 0x31 <b>Control register of the selected line interface, register 0</b> Before writing this array register the line interface must be selected by register R_SU_SEL.			
0	0	V_B1_TX_EN	<b>B1-channel transmit</b> '0' = B1 send data disabled (permanent '1's sent when the line interface is activated) '1' = B1 send data enabled
1	0	V_B2_TX_EN	<b>B2-channel transmit</b> '0' = B2 send data disabled (permanent '1's sent when the line interface is activated) '1' = B2 send data enabled
2	0	V_SU_MD	<b>Line interface mode</b> '0' = TE mode '1' = NT/LT mode
3	0	V_ST_D_LPRI0	<b>D-channel priority</b> '0' = high priority 8/9 '1' = low priority 10/11 <b>Note:</b> 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	<b>S/Q bits transmission</b> '0' = S/Q bits disabled '1' = S/Q bits (multiframe) enabled <b>Note:</b> 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	<b>Send test signal</b> '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: <b>S/T interface mode:</b> Test signal is 96 kHz (alternating '0's) or 2 kHz (one alternating '0' per frame) <b>Up interface mode:</b> 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	<b>End of pulse control</b> 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 <b>Note:</b> This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
7	0	V_SU_STOP	<b>Power down</b> '0' = external receiver activated '1' = power down, external receiver disabled

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

A\_SU\_CTRL2 [ST/Up]

(w)

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	<b>Enable B1-channel receive</b> '0' = B1 receive bits are forced to '1' '1' = normal operation
1	0	V_B2_RX_EN	<b>Enable B2-channel receive</b> '0' = B2 receive bits are forced to '1' '1' = normal operation
2	0	V_MS_SSYNC2	<b>Multiframe / superframe single synchronization pulse</b> The multiframe / superframe synchronization pulse can be configured to occur only once when this bit is set to '1'.  <b>S/T interface mode:</b> '0' = normal multiframe bit M is transmitted (once every 20 S/T frames) '1' = single multiframe bit can be transmitted  <b>Up interface mode:</b> '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')  <b>Note:</b> 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	<b>Multiframe / superframe bit source selection</b> This bit has different meaning for line interfaces in S/T or Up mode: <b>S/T interface mode:</b> Source selection of multiframe S/Q-bits and multiframe bit M <b>Up interface mode:</b> 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  <b>Note:</b> 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	<b>V_SU_2KHZ</b>	<b>Select test signal frequency</b> '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	<b>V_SU_TRI</b>	<b>Line interface output buffer tristated</b> '0' = normal operation '1' = set output buffer into tristate mode (i.e. transmitter is switched off)
7	0	<b>V_SU_EXCHG</b>	<b>Exchange transmit output buffers</b> Change of the polarity of the line interface output pins. This is equal to an external crossing of the two transmit pins.

A_MS_TX [ST/Up]		(w)	0x34
<b>Multiframe/superframe transmit register</b>			
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_MS_TX	<b>Multiframe/superframe bits</b> The meaning of this bitmap depends on the selected line interface mode: <b>S/T interface mode:</b> 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. <b>Up interface mode:</b> 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	<b>S-bit of the Up superframe</b> 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. <b>Note:</b> 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
<b>A_ST_CTRL3</b> [ST/Up] (w) 0x35 <b>Control register of the selected line interface, register 3</b> This register is an array register <i>and</i> 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_ST_SEL = '0' in this register.  <b>Note:</b> This register is only used in S/T interface mode and should be 0xF8. See register A_UP_CTRL3 when the line interface is in Up mode.			
0	0	V_ST_SEL	<b>Line interface mode selection</b> 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 <b>Note:</b> This bit should only be changed after reset.
7..1	0	V_ST_PULSE	<b>End of pulse control</b> The shape of the pulse end can be adjusted with this bitmap. V_ST_PULSE = 0x7C should be used.

A\_UP\_CTRL3 [ST/Up]

(w)

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><b>Line interface mode selection</b></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><b>Note:</b> This bit should only be changed after reset.</p>
1	0	V_UP_VIO	<p><b>Up activation after superframe violation</b></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><b>DC-balancing mode</b></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><b>Note:</b> DC-balancing must be enabled with V_UP_DC_OFF = '0' to generate DC-balancing bits</p>
3	0	V_UP_DC_OFF	<p><b>DC-balancing bit disabled</b></p> <p>'0' = normal operation</p> <p>'1' = The DC-balancing bit is always '0'</p>
4	0	V_UP_RPT_PAT	<p><b>Allow repeated patterns of scrambled data</b></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><b>Scrambler / descrambler mode selection</b></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><b>Note:</b> 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	<b>Transmit scrambler disabled</b> 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	<b>Receive descrambler disabled</b> 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)	0x36
<b>Multiframe/superframe data flow configuration register</b>			
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	<b>No BAC-bit inversion</b> '0' = invert BAC-bit '1' = no inversion of BAC-bit
1	0	V_SG_AB_INV	<b>Invert S/G or A/B bit</b> '0' = normal operation '1' = invert S/G or A/B bit (depending on V_SQ_T_DST)
2	0	V_SQ_T_SRC	<b>Source of S/Q-bits (S/T multiframe) or T-bits (Up superframe)</b> '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	<b>Source of M-bit (S/T multiframing bit) or S-bit (Up superframe)</b> <b>S/T interface mode:</b> 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)  <b>Up interface mode:</b> '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	<b>Destination of S/Q-bits (S/T multiframe) or T-bits (Up superframe)</b> '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	<b>Value of unused bits in received D- and E-channel</b> 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)	0x37
<b>Clock control register of the line interface module</b>			
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		V_SU_CLK_DLY	<p><b>Line interface clock delay</b>  <b>TE mode (S/T or Up):</b> 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.  <b>NT mode (for S/T interface mode only):</b> Data sample point. The lower the value the earlier the input data is sampled. The normal operation value is 0xC.  <b>LT mode (for Up interface mode only):</b> 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		V_ST_SMPL	<p><b>Early edge input data shaping</b>                      (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.  <b>Note:</b> 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)	0x3C	
<b>Transmit register for the B1-channel data</b>			
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	<b>B1-channel data byte</b> Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

A_B2_TX[ST/Up]	(w)	0x3D	
<b>Transmit register for the B2-channel data</b>			
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	<b>B2-channel data byte</b> Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

Bits	Reset value	Name	Description
<b>Transmit register for the D-channel data</b>			
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_D_TX_S	<b>S-bit of the Up superframe</b> 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'.
5		V_D_TX_BAC	<b>BAC-bit</b> 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	<b>D-channel data bits</b>

A_BAC_S_TX [ST/Up]		(w)	0x3F
<b>BAC-bit and S-bit for multiframe / superframe transmission</b>			
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).			
Bits	Reset value	Name	Description
0		V_S_TX	<b>S-bit of the Up superframe</b> 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 0.
5		V_BAC_TX	<b>BAC-bit</b> 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	0	(reserved)	Must be '00'.

**5.5.2 Read only registers**

<b>R_AF0_OVIEW</b>		<b>(r)</b>	<b>0x13</b>
<b>Alternate frame synchronization signal overview register</b>			
<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		<b>V_SU0_AF0</b>	<b>FSC selection for line interface 0</b> This bit is equal to A_SU_STA with line interface 0 being selected.
1		<b>V_SU1_AF0</b>	<b>FSC selection for line interface 1</b> This bit is equal to A_SU_STA with line interface 1 being selected.
7..2	0	(reserved)	

<b>A_SU_RD_STA</b> [ST/Up]		<b>(r)</b>		<b>0x30</b>
<b>ST/Up state machine register</b>				
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.				
<b>Bits</b>	<b>Reset value</b>	<b>Name</b>	<b>Description</b>	
3..0	0	<b>V_SU_STA</b>	<b>ST/Up state</b> Binary value of current state (NT/LT: Gx, TE: Fx)	
4	0	<b>V_SU_FR_SYNC</b>	<b>Frame synchronization</b> '0' = not synchronized '1' = synchronized	
5	0	<b>V_SU_T2_EXP</b>	<b>Timer T2 expired</b> '1' = timer T2 expired (NT/LT mode only)	
6	0	<b>V_SU_INFO0</b>	<b>INFO 0</b> '1' = receiving INFO 0	
7	0	<b>V_G2_G3</b>	<b>G2 to G3 transition allowed</b> '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)	0x31
<b>NT-TE-NT/LT-TE-LT delay</b>		
<p>This register shows the round trip delay and is only valid in NT/LT mode. It is updated once every 250 <math>\mu</math>s.</p> <p><b>Line interface in S/T mode:</b> 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 <math>t_0 = 1/6.144\text{MHz} = 1/32</math> bit length = 162.8ns. The minimum delay is 2 bit times because of the NT/TE frame offset.</p> <p><b>Line interface in Up mode:</b> 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 <math>t_0 = 1/12.288\text{MHz} = 1/32</math> bit length = 81.4ns. 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
4..0	0x00	<b>V_SU_DLYL</b>
<p><b>Lower part of the round trip delay</b>                      The lower part of the delay time  <math>t_{dly,l} = t_0 \cdot V\_SU\_DLYL</math>                      Register A_SU_DLYH must also be read to obtain the whole delay <math>t_{dly} = t_{dly,l} + t_{dly,h}</math>.</p> <p><b>S/T interface mode:</b>                      0 = delay <math>t_{dly,l} = 0</math> ns                      1 = delay <math>t_{dly,l} = 162.8</math> ns                      2 = delay <math>t_{dly,l} = 325.5</math> ns                      3 = delay <math>t_{dly,l} = 488.3</math> ns                      ...                      31 = delay <math>t_{dly,l} = 5045.6</math> ns</p> <p><b>Up interface mode:</b>                      0 = delay <math>t_{dly,l} = 0</math> ns                      1 = delay <math>t_{dly,l} = 81.4</math> ns                      2 = delay <math>t_{dly,l} = 162.8</math> ns                      3 = delay <math>t_{dly,l} = 244.1</math> ns                      ...                      31 = delay <math>t_{dly,l} = 2522.8</math> ns</p>		
7..5	0	(reserved)

<b>A_SU_DLYH</b> [ST/Up]	(r)	0x32
<p><b>NT-TE-NT / LT-TE-LT delay</b></p> <p>This register shows the round trip delay and is only valid in NT/LT mode. It is updated once every 250 <math>\mu</math>s.</p> <p><b>Line interface in S/T mode:</b> 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 <math>t_0 = 1/6.144\text{MHz} = 1/32\text{bit length} = 162.8\text{ns}</math>. The minimum delay is 2 bit times because of the NT/TE frame offset.</p> <p><b>Line interface in Up mode:</b> This register reports the <i>end of transmit frame to begin of receive frame</i> delay of a device in LT mode. The resolution is <math>t_0 = 1/12.288\text{MHz} = 1/32\text{bit length} = 81.4\text{ns}</math>. 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 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 <math>32 \cdot t_0 = 5.208\mu\text{s}</math> for a line interface in S/T mode and <math>32 \cdot t_0 = 2.604\mu\text{s}</math> for a line interface in Up mode, which is the length of one bit in both cases.</p> <p>Before reading this array register the line interface must be selected by register R_SU_SEL.</p>		
<b>Bits</b>	<b>Reset value</b>	<b>Name</b>
<b>Description</b>		
4..0	0x00	<p><b>V_SU_DLYH</b></p> <p><b>Upper part of the round trip delay</b>                      The upper part of the delay time  <math>t_{dly,h} = 32 \cdot t_0 \cdot \text{V\_SU\_DLYH}</math>.                      Register A_SU_DLYL can be read to obtain the fractional part of a bit time delay.</p> <p><b>S/T interface mode:</b>                      0 = delay <math>t_{dly,h} = 0\mu\text{s}</math>                      1 = delay <math>t_{dly,h} = 5.208\mu\text{s}</math>                      2 = delay <math>t_{dly,h} = 10.416\mu\text{s}</math>                      3 = delay <math>t_{dly,h} = 15.625\mu\text{s}</math>                      ...                      31 = delay <math>t_{dly,h} = 161.458\mu\text{s}</math></p> <p><b>Up interface mode:</b>                      0 = delay <math>t_{dly,h} = 0\mu\text{s}</math>                      1 = delay <math>t_{dly,h} = 2.604\mu\text{s}</math>                      2 = delay <math>t_{dly,h} = 5.208\mu\text{s}</math>                      3 = delay <math>t_{dly,h} = 7.812\mu\text{s}</math>                      ...                      31 = delay <math>t_{dly,h} = 80.729\mu\text{s}</math></p>
7..5	0	(reserved)

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

A\_SU\_STA [ST/Up]

(r)

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><b>Current D-channel priority state if high priority is selected</b></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><b>Note:</b> This bit is only valid in S/T interface mode.</p>
1	0	V_ST_D_LPRIO11	<p><b>Current D-channel priority state if low priority is selected</b></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><b>Note:</b> This bit is only valid in S/T interface mode.</p>
2	0	V_ST_D_CONT	<p><b>D-channel contention</b></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><b>Note:</b> This bit is only valid in S/T interface mode.</p>
3	0	V_ST_D_ACT	<p><b>D-channel active</b></p> <p>This bit has the value '1' when the D-channel is just transmitting data.</p> <p><b>Note:</b> This bit is only valid in S/T interface mode.</p>
6..4	0	(reserved)	

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	V_SU_AF0	<p><b>FSC selection</b></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)	0x3C	
<b>Receive register for the B1-channel data</b>			
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	0xFF	V_B1_RX	<p><b>B1-channel data byte</b></p> <p>Data is valid after a F0IO or AF0 pulse.</p>

A_B2_RX [ST/Up]	(r)	0x3D	
<b>Receive register of the B2-channel data</b>			
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	0xFF	V_B2_RX	<p><b>B2-channel data byte</b></p> <p>Data is valid after a F0IO or AF0 pulse.</p>

A_D_RX [ST/Up]		(r)	0x3E
<b>Receive register for the D-channel data</b>			
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 F0IO or AF0 pulse.			
Bits	Reset value	Name	Description
0		V_D_RX_S	<b>S-bit of the Up superframe</b> Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_E_RX and A_MS_RX. <b>Note:</b> 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_D_RX_AB	<b>A/B-bit</b> 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.
5		V_D_RX_SG	<b>S/G-bit</b> 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.
7..6	3	V_D_RX	<b>D-channel data bits</b>

A_E_RX [ST/Up]		(r)	0x3F
<b>Receive register for the E-channel data</b>			
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 F0IO or AF0 pulse.			
Bits	Reset value	Name	Description
0		V_E_RX_S	<b>S-bit of the Up superframe</b> Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_D_RX and A_MS_RX. <b>Note:</b> 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	<b>A/B-bit</b> 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	<b>S/G-bit</b> 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	3	V_E_RX	<b>E-channel data bits</b>





## 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	242	0x18	R_F0_CNTL	259
0x14	R_PCM_MD0	243	0x19	R_F0_CNTH	259
0x15	R_SL_SEL0	244	0x1D	R_SL_MAX	259
0x15	R_SL_SEL1	245	0x28	R_CI_RX	260
0x15	R_SL_SEL7	245	0x29	R_GCI_STA	261
0x15	R_MSS0	246	0x2A	R_MON_RX	261
0x15	R_PCM_MD1	248			
0x15	R_PCM_MD2	249	Read / write registers:		
0x15	R_MSS1	251	<b>Address</b>	<b>Name</b>	<b>Page</b>
0x15	R_SH0L	252	0xD0	A_SL_CFG	262
0x15	R_SH0H	252			
0x15	R_SH1L	252			
0x15	R_SH1H	253			
0x17	R_SU_SYNC	254			
0x28	R_CI_TX	255			
0x29	R_GCI_CFG0	256			
0x2A	R_GCI_CFG1	258			
0x2B	R_MON_TX	258			

## 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

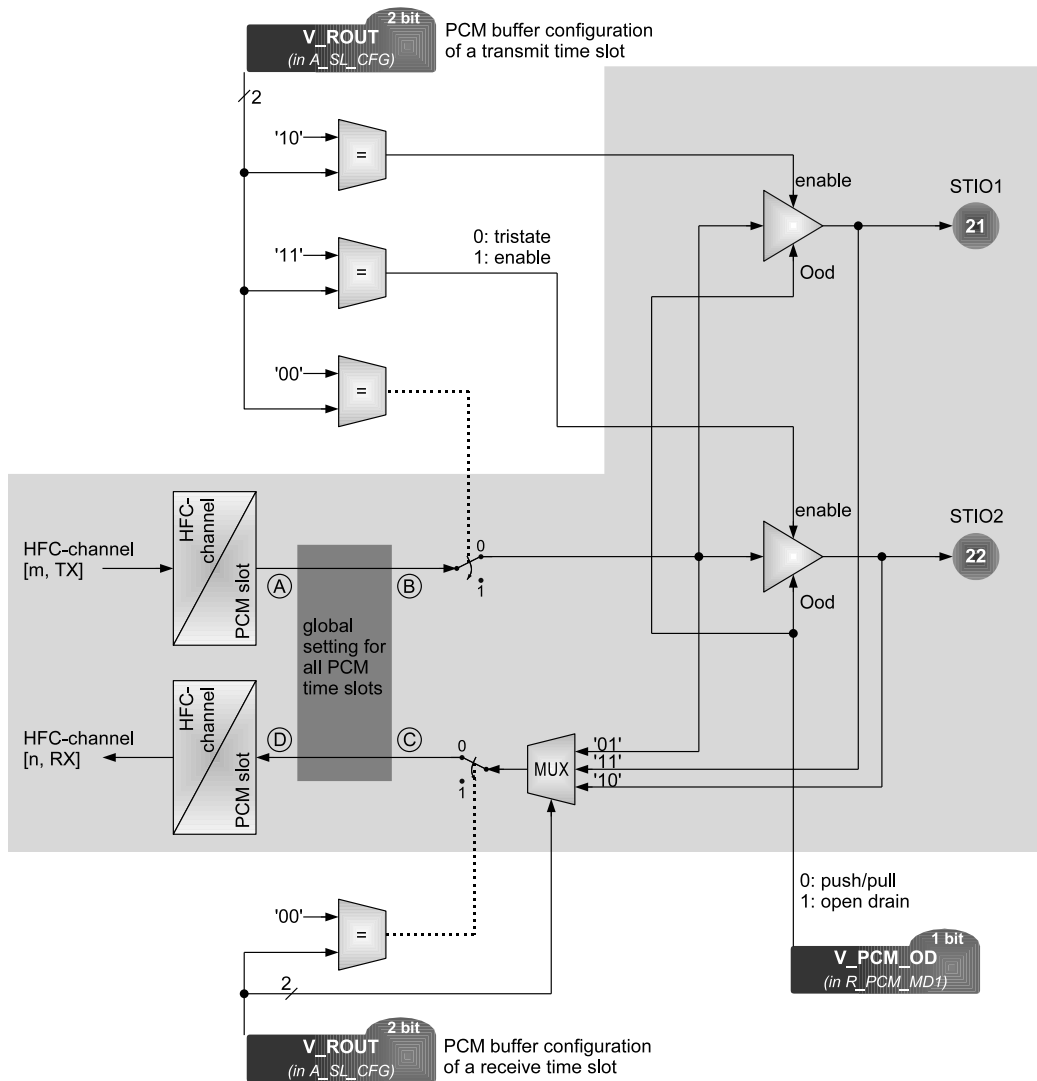
$$f(\text{C4IO}) = n \cdot 128 \text{ kHz} \quad \text{with } n = 1 \dots 128$$

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

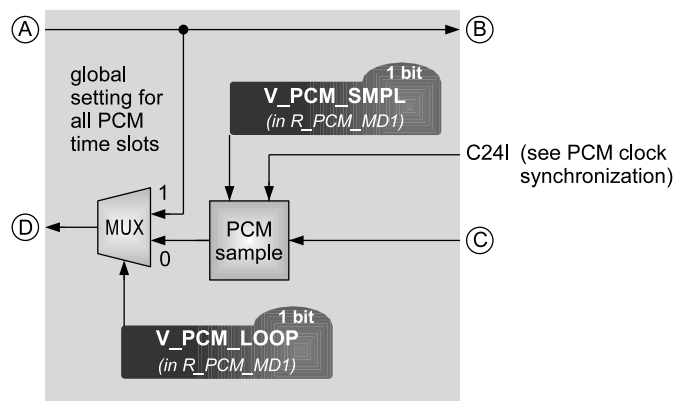
The PCM data flow is shown in Figure 6.1. The input and output behavior has to be programmed with bitmap `V_ROUT` in array register `A_SL_CFG[SLOT]`. Every time slots has its own configuration.



**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.



**Figure 6.2:** Global setting for all PCM time slots to set up an internal PCM loop (detail of Figure 6.1), normally used for test loop setup

PCM input data selects one of three data sources:

- V\_ROUT = '10' receives data from STIO2 .
- V\_ROUT = '11' receives data from STIO1 .
- V\_ROUT = '01' is used for an internal data loop.

The data transfer to the receive HFC-channel can be disabled with V\_ROUT = '00'.

A corresponding transmit /receive pair of PCM time slots is typically programmed with the same value for V\_ROUT in both directions. Table 6.3 summarizes these settings.

**Table 6.3:** PCM data flow programming with the same value of V\_ROUT in corresponding transmit and receive time slots

V_ROUT	Data transmission from/to the HFC-channel *1	STIO1 I/O path *2	STIO2 I/O path *2	Description
'00'	disabled	tristated	tristated	PCM time slot not used
'01'	enabled	tristated	tristated	internal data loop
'10'	enabled	output	input	bidirectional data transfer
'11'	enabled	input	output	bidirectional data transfer

\*1: PCM data flow configuration of a receive time slot

\*2: PCM data flow configuration of a transmit time slot

Figure 6.1 shows the PCM data flow which can be programmed for each PCM time slot individually. Global settings to the PCM data flow are available between (A) .. (D) as shown in Figure 6.2. When V\_PCM\_LOOP = '1' in register R\_PCM\_MD1, the PCM data is looped internally.

## 6.3 PCM initialization

After hard or soft 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<sub>p</sub> interface and the flow controller as well.

### 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 227).

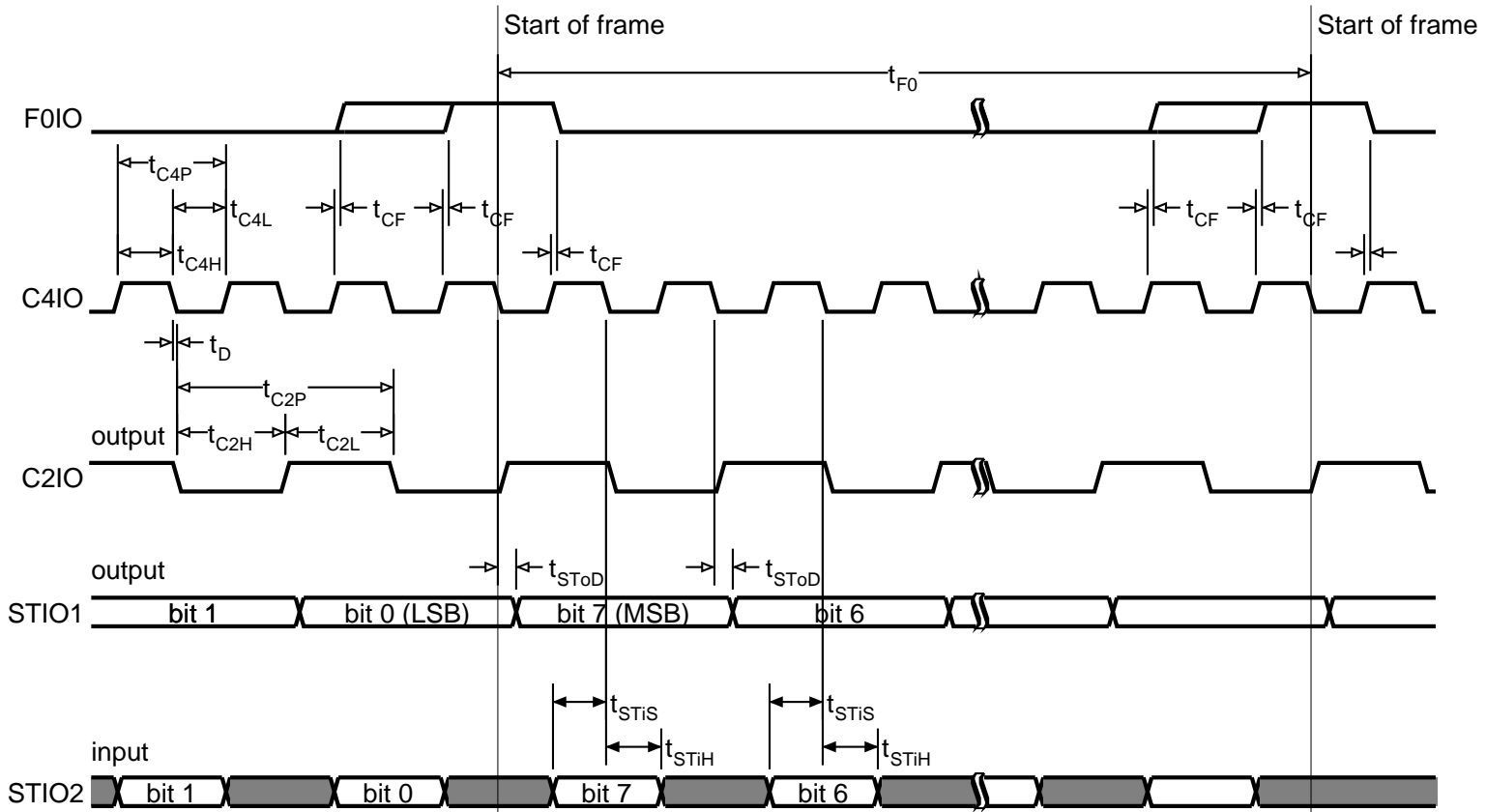
### 6.4.3 Slave mode

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<sub>p</sub> 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

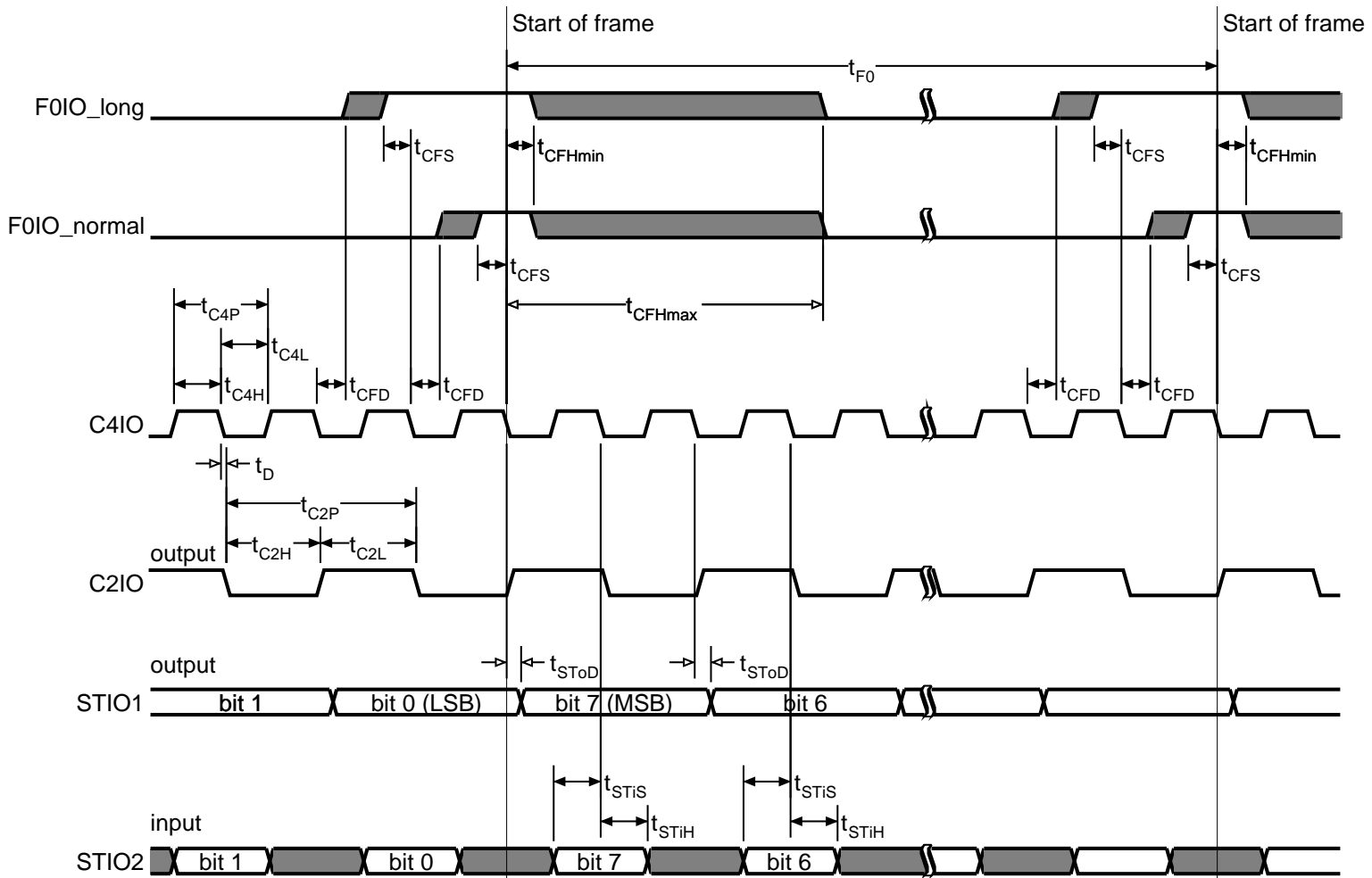


STIO1 is shown as data output and STIO2 as data input. However, both pins can change their I/O characteristic with every PCM time slot.

**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 $\downarrow$ 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 157) and 5.13 (page 173) and with the MSS controller (shown in Figure 6.10 on page 225).

### 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/Up` interface.<sup>1</sup> 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.

---

<sup>1</sup>C4IO adjustment is only in operation when the PCM DPLL receives both 8 kHz reference clocks.

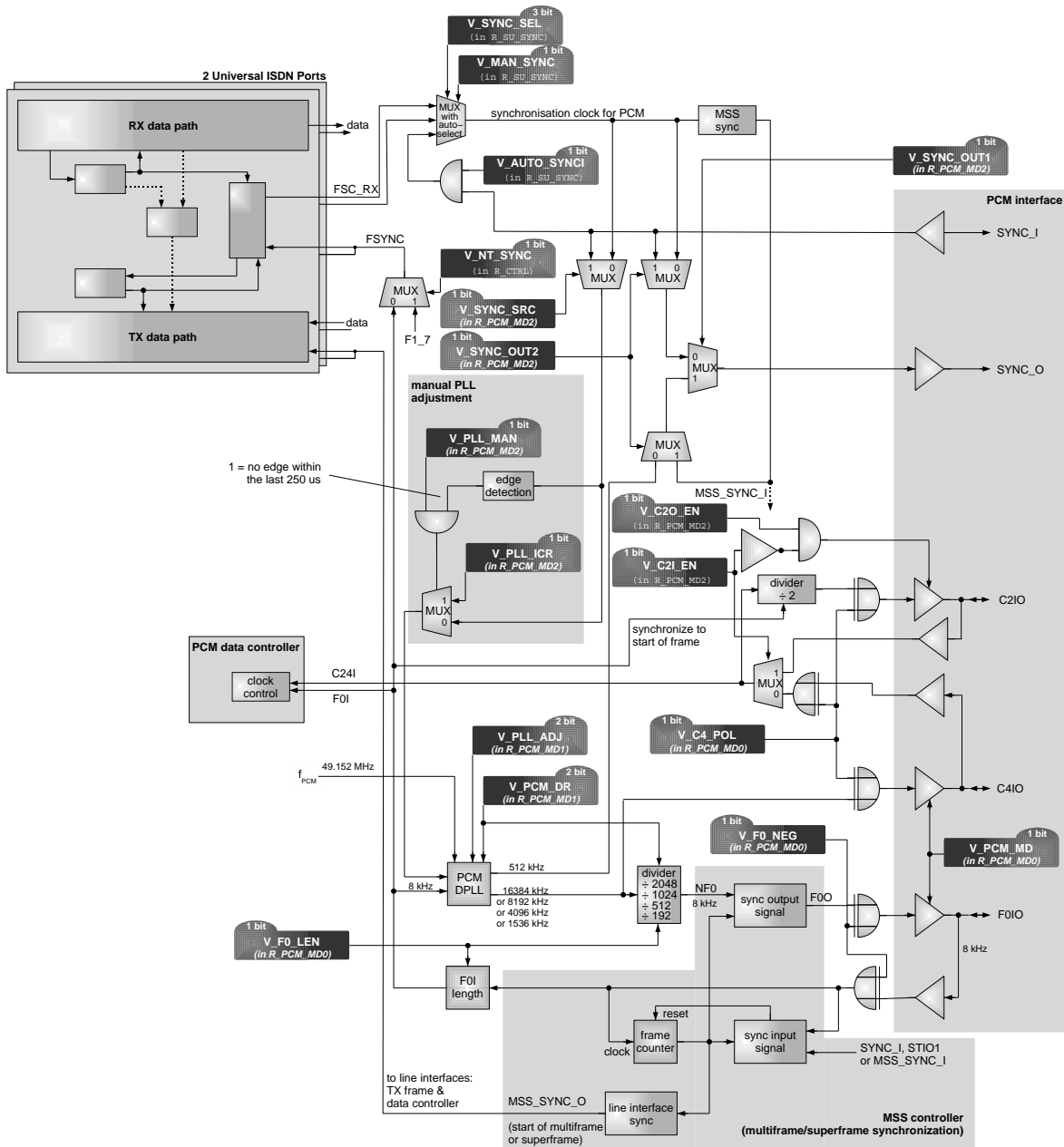


Figure 6.5: PCM clock synchronization (For details on the Universal ISDN Port see Figures 5.5 and 5.13, for details on the MSS controller see Figure 6.10)

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 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.

#### 6.5.6.1 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.

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.

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.

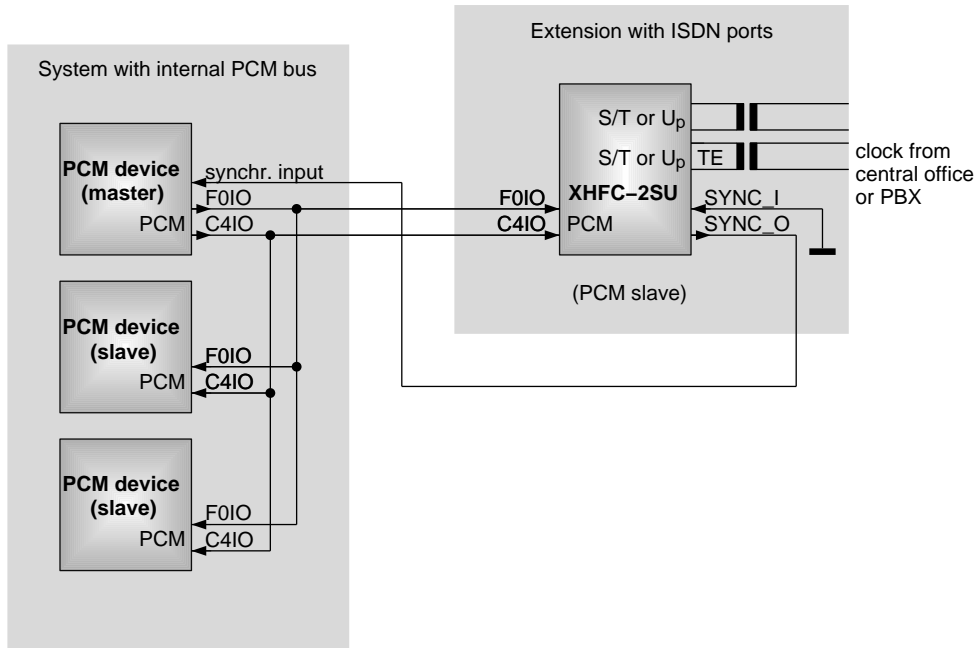


Figure 6.6: Expanding an existing system with ISDN ports (XHFC-2SU as PCM slave)

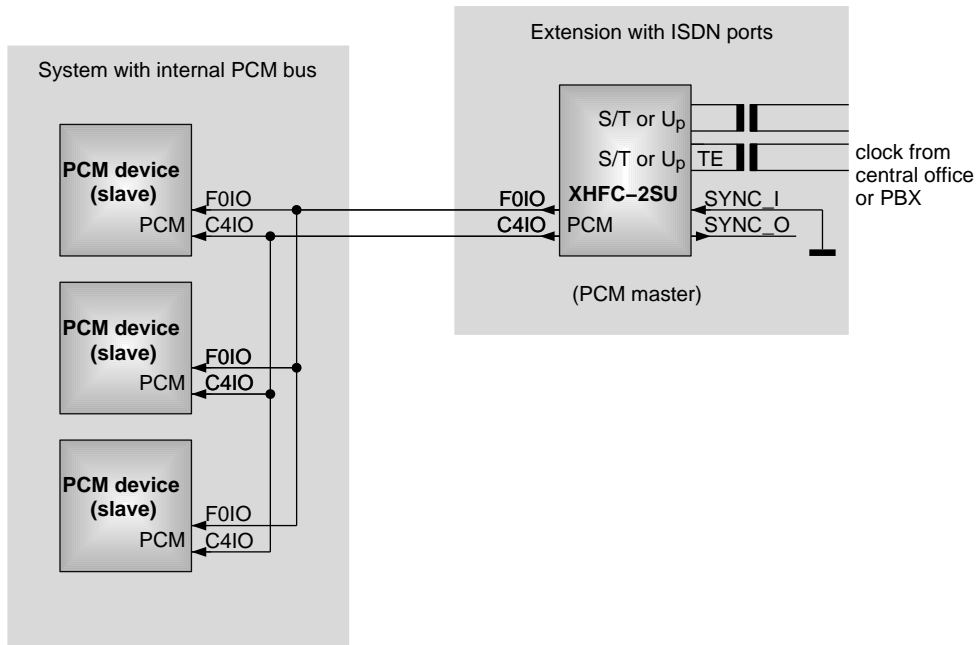


Figure 6.7: Expanding an existing system with ISDN ports (XHFC-2SU as PCM master)

### 6.5.6.2 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.

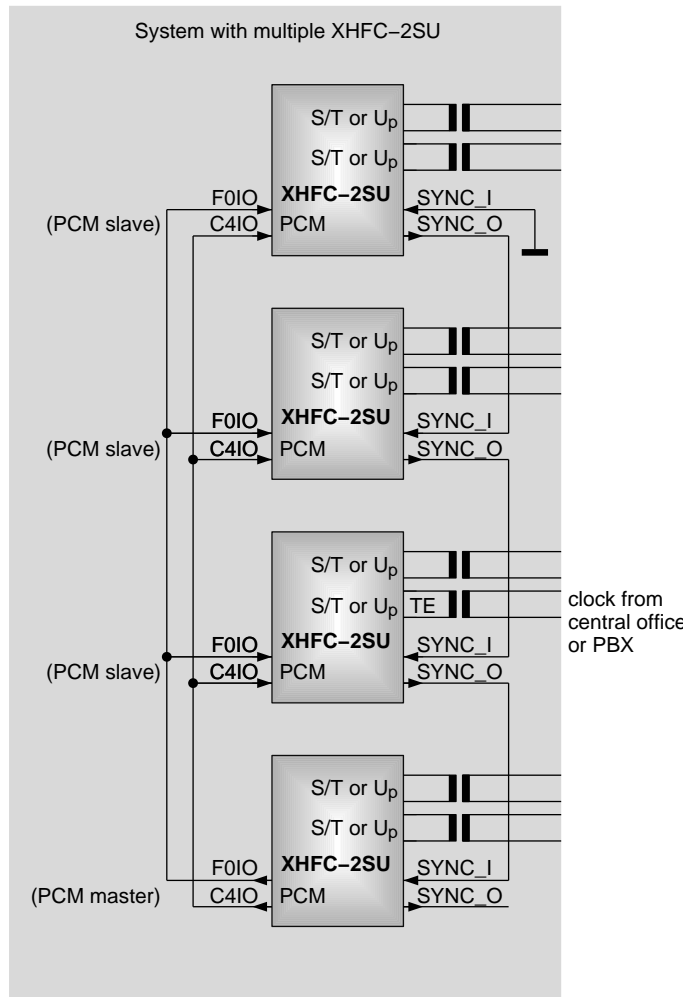


Figure 6.8: Multiple XHFC-2SU synchronized with an open loop of SYNC\_O / SYNC\_I

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.

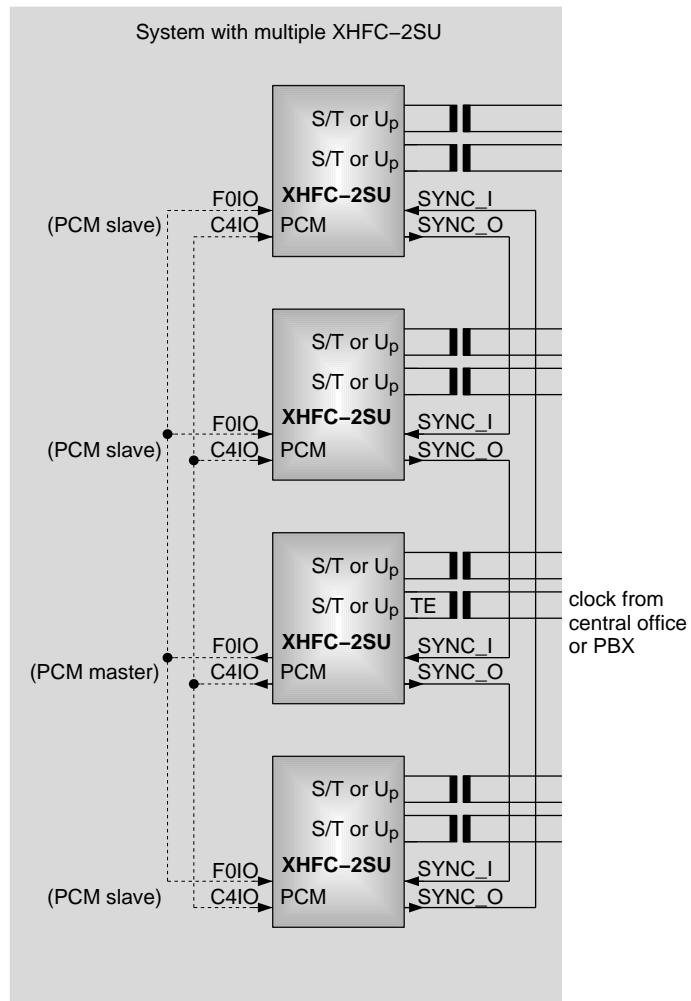


Figure 6.9: Multiple XHFC-2SU synchronized with an closed loop of SYNC\_O / SYNC\_I

## 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 157 and Figure 5.13 on page 173)

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.

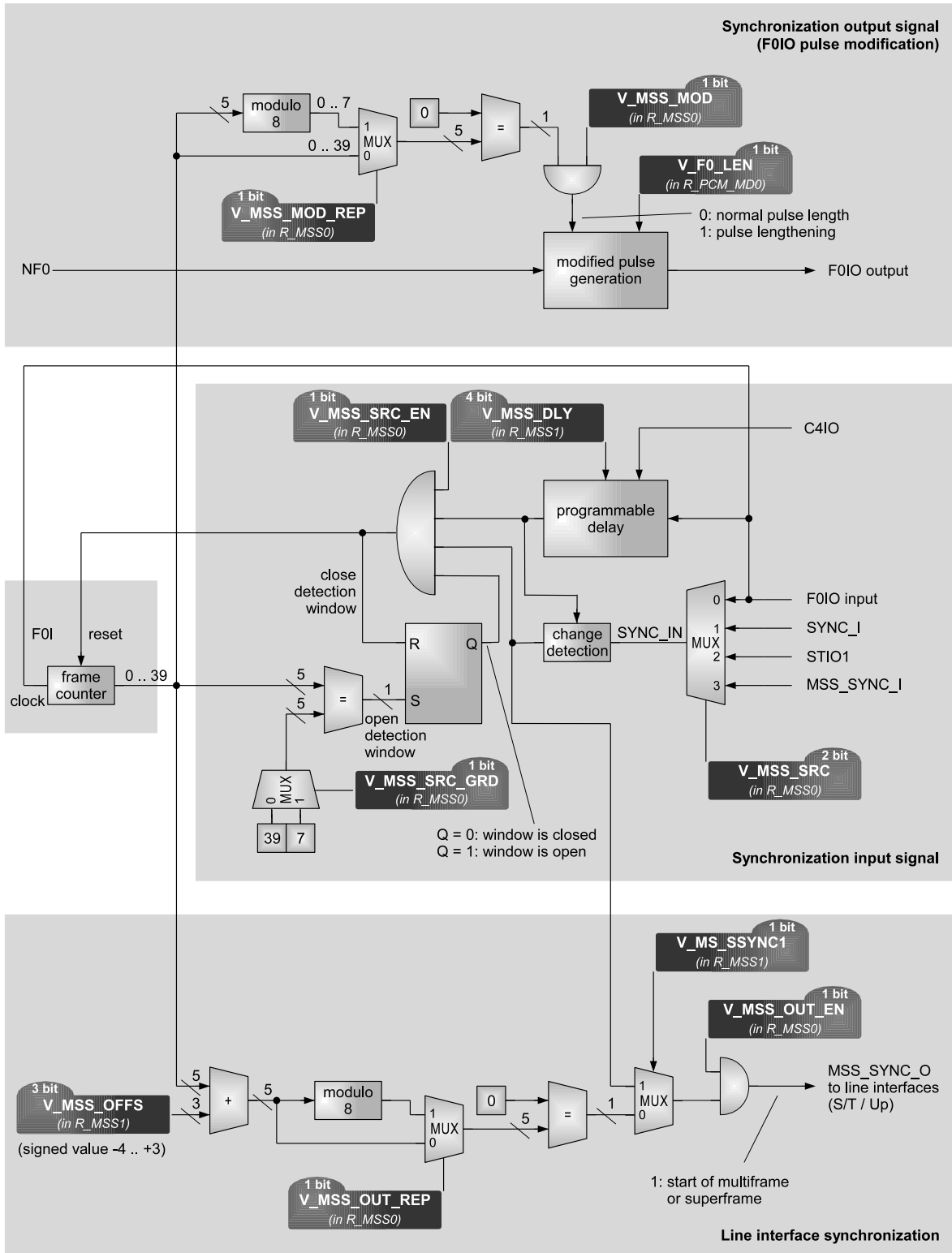


Figure 6.10: Multiframe / superframe synchronization to the PCM interface

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<sub>p</sub> 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.

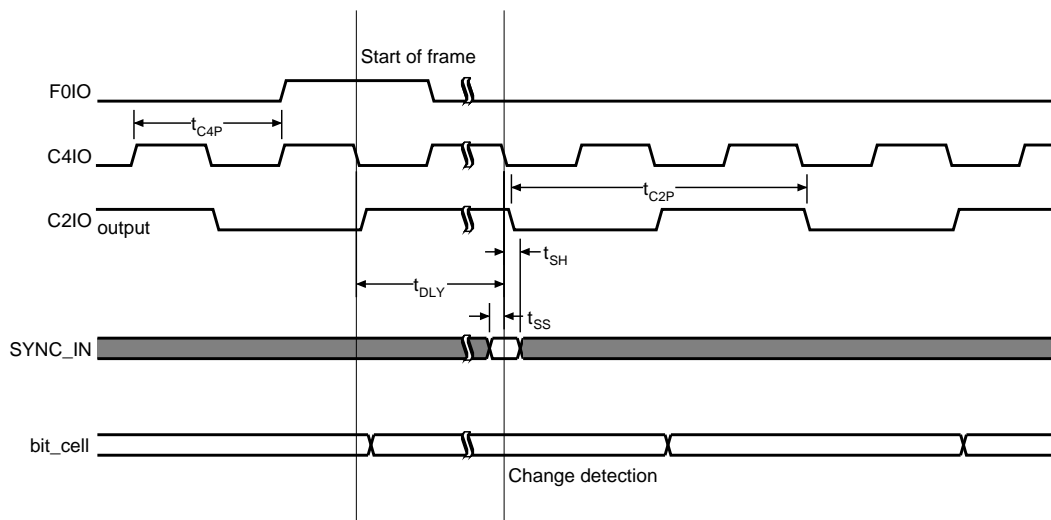


Figure 6.11: Timing specification of the SYNC\_IN change detection

**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

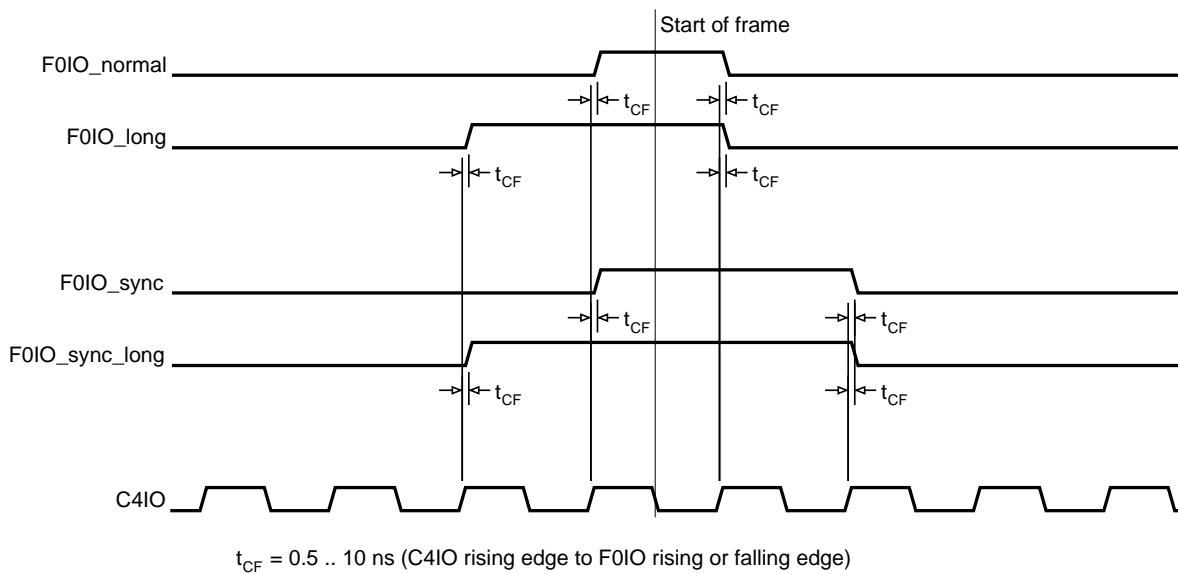
### 6.6.4 Synchronization output signal (F0IO pulse modification)

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 multi-frame/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<sub>p</sub> 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<sub>p</sub> 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<sub>p</sub> 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.

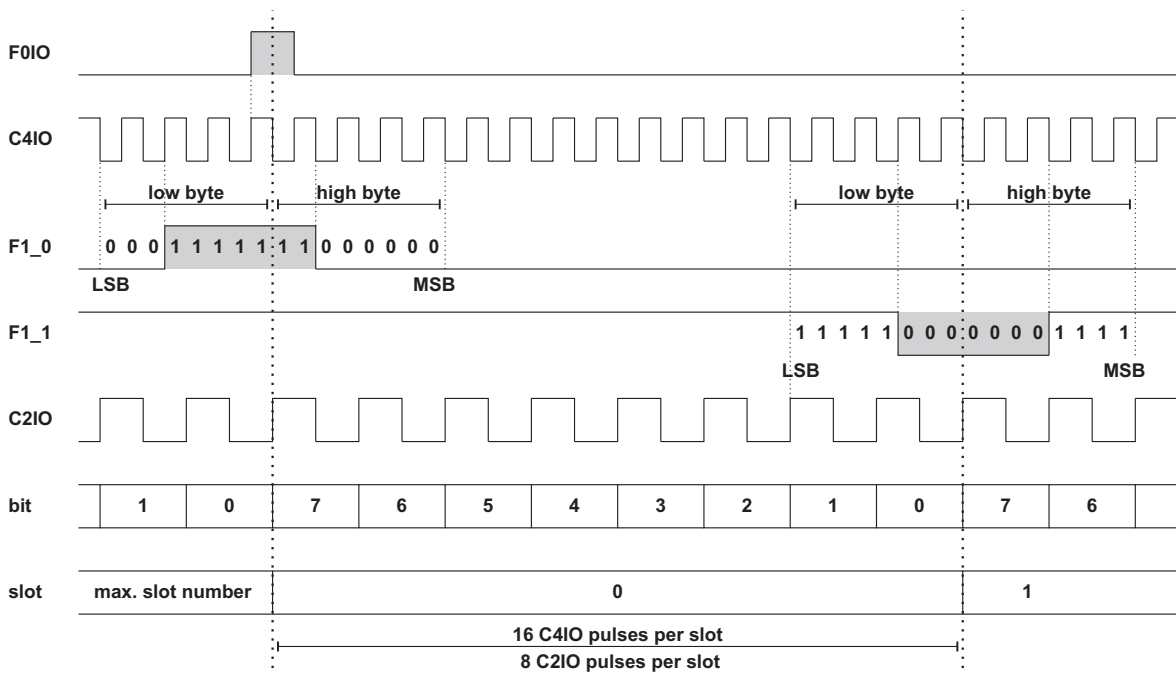


Figure 6.13: Example for two CODEC enable signal shapes

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 = '11111000'  $\xrightarrow{\text{reverse}}$  '00011111')  
 R\_PCM\_MD0 : V\_PCM\_IDX = 0xD      (R\_SH0H register accessible)  
 R\_SH0H    : V\_SH0H    = 0x03      (0x03 = '00000011'  $\xrightarrow{\text{reverse}}$  '11000000')  
 R\_PCM\_MD0 : V\_PCM\_IDX = 0xE      (R\_SH1L register accessible)  
 R\_SH1L    : V\_SH1L    = 0x1F      (0x1F = '00011111'  $\xrightarrow{\text{reverse}}$  '11111000')  
 R\_PCM\_MD0 : V\_PCM\_IDX = 0xF      (R\_SH1H register accessible)  
 R\_SH1H    : V\_SH1H    = 0xF0      (0xF0 = '11110000'  $\xrightarrow{\text{reverse}}$  '00001111')

## 6.8 GCI/IOM-2 mode

### 6.8.1 Overview

XHFC-2SU is equipped with a simple GCI controller<sup>2</sup> (also known as IOM<sup>TM</sup>-2)<sup>3</sup> to support interconnection to U-chips, external CODECs or DSPs.

The IOM<sup>TM</sup>-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 group<sup>4</sup>. The GCI<sup>5</sup> functionality has been implemented in respect to the IOM<sup>TM</sup>-2 specification [6].

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.

---

<sup>2</sup>GCI = General Circuit Interface

<sup>3</sup>IOM<sup>TM</sup>-2= ISDN Oriented Modular revision 2, trademark of Infineon Technologies AG

<sup>4</sup>Alcatel (France), Siemens (Germany), Italtel (Italy) and Plessey (UK)

<sup>5</sup>The term GCI is always used in this document and includes IOM<sup>TM</sup>-2 functionality as well.

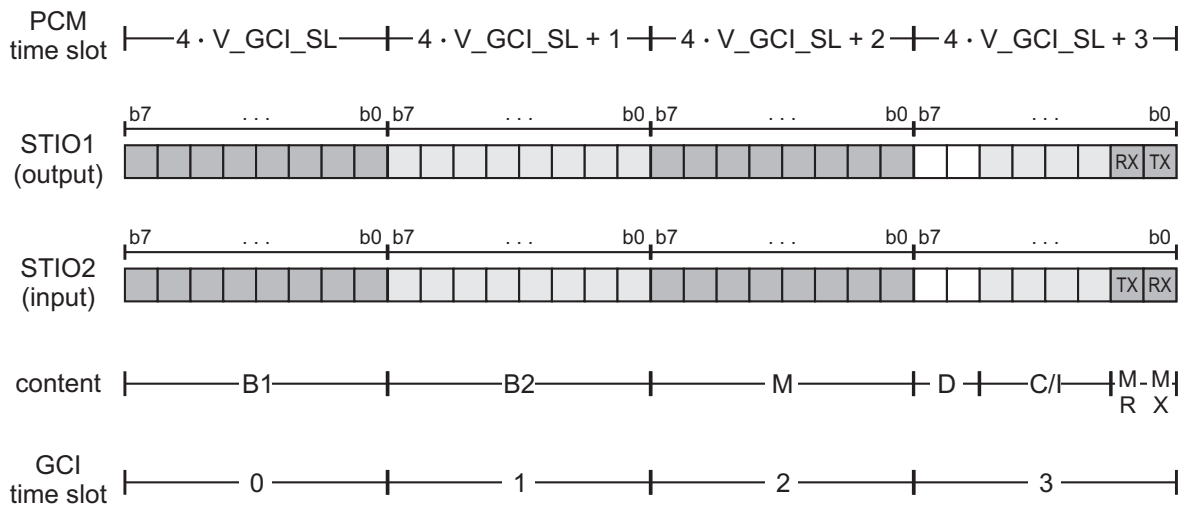


Figure 6.14: Single channel GCI format

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 <sub>RX</sub> on STIO1 output and MR <sub>TX</sub> on STIO2 input)
MX	Handshake bit for the monitor channel (MX <sub>TX</sub> on STIO1 output and MX <sub>RX</sub> 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.

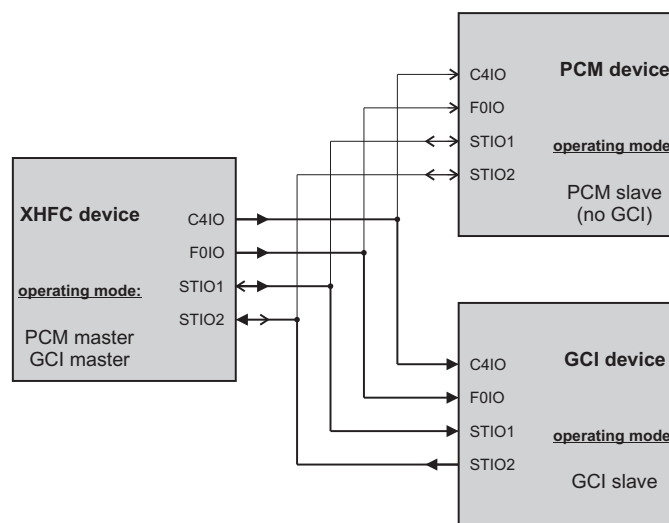
### 6.8.3.4 Examples for GCI frame embedding in the PCM data structure

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

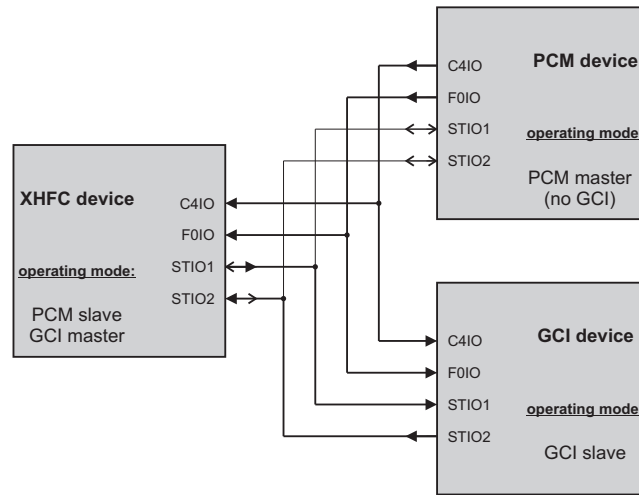


Figure 6.16: GCI application example 2 (XHFC-2SU is PCM slave and GCI master)

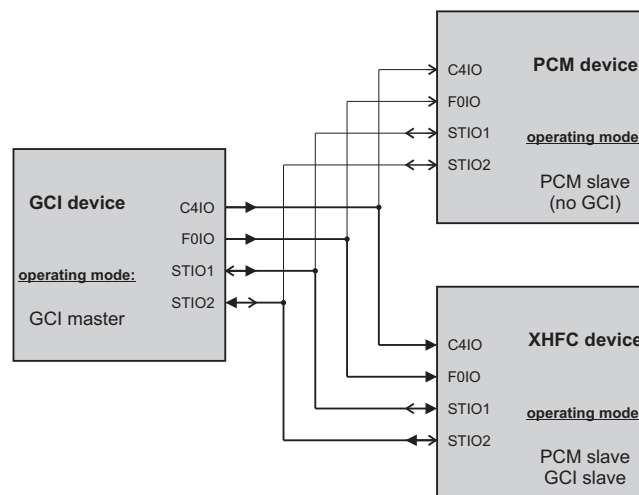


Figure 6.17: GCI application example 3 (XHFC-2SU is PCM slave and GCI slave)

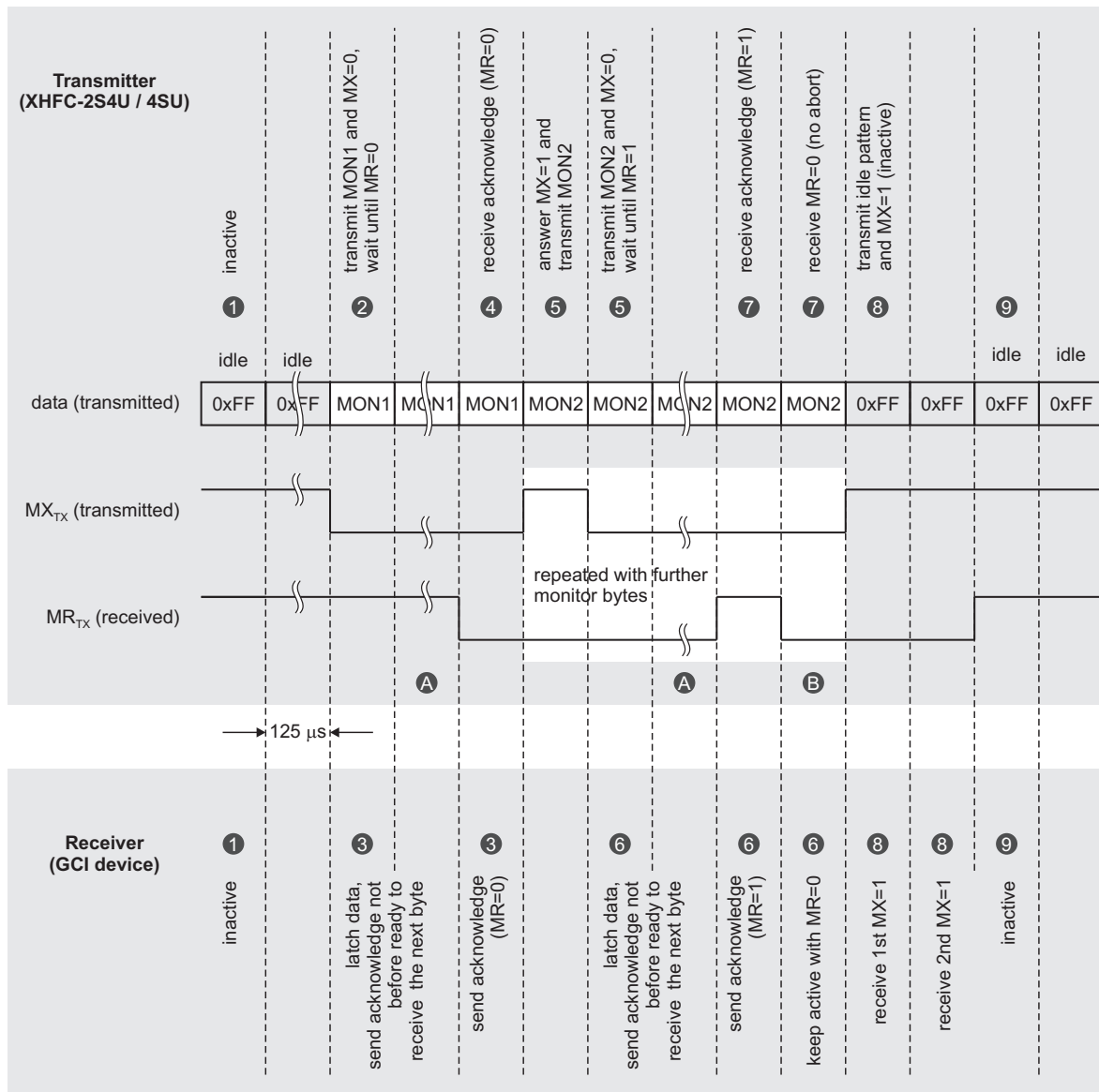
**Table 6.10:** Programming procedure for application example 3 according to Figure 6.17 (XHFC-2SU is PCM slave and GCI slave)

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.



- A omitted, if the receiver sends acknowledge immediately
- B omitted, if  $V\_MON\_SLOW = 0$  (high transmission speed)

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  $\mu$ 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  $\mu$ 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.

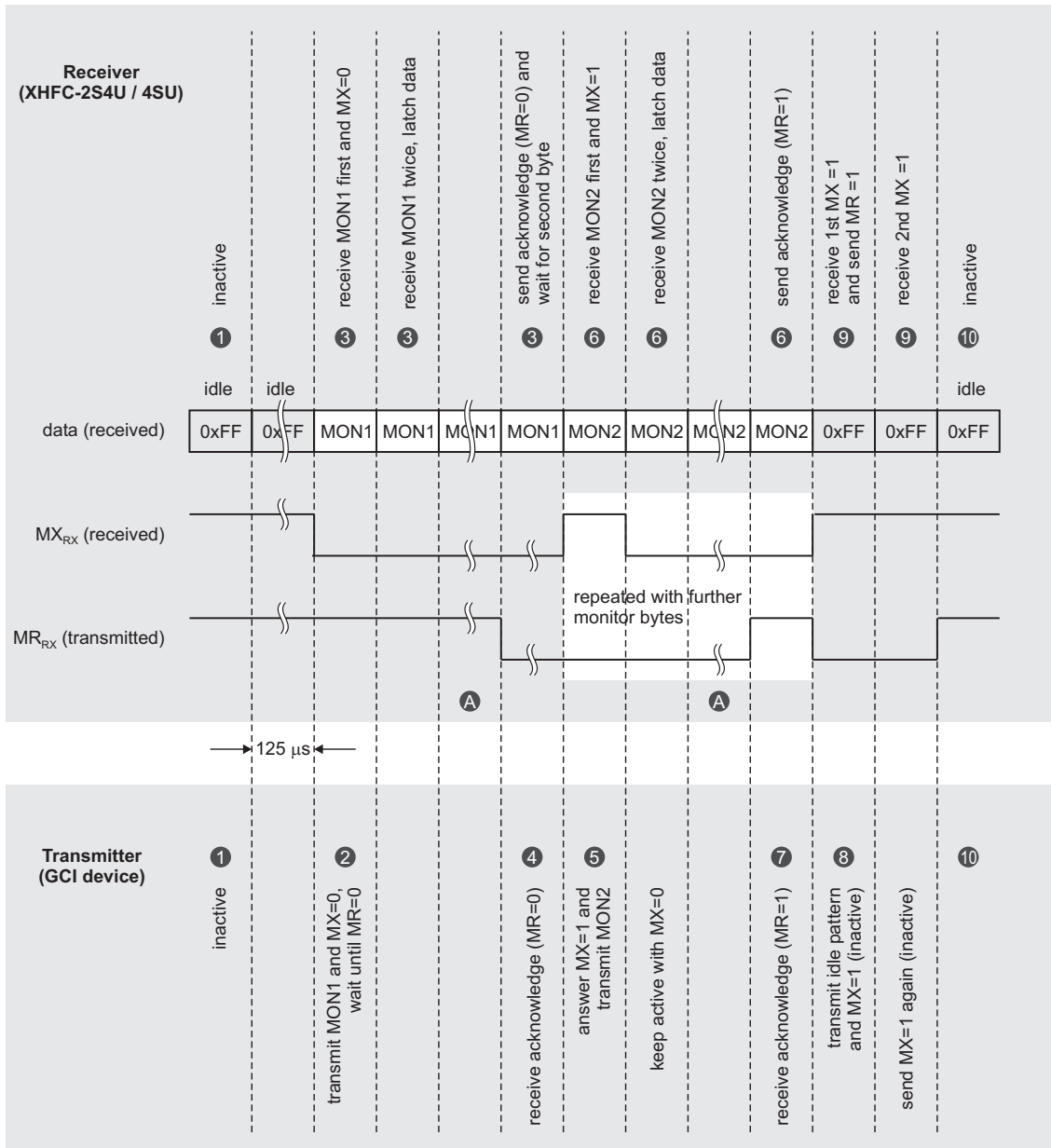
**Table 6.11:** Rules for the handshake signals  $MX_{TX}$  and  $MR_{TX}$  when XHFC-2SU transmits a monitor byte sequence

$MX_{TX}$	$MR_{TX}$	Monitor channel state
'1'	'1'	Idle
'0'	'0'	Transmitter and receiver are both active
'0'	'1' (once)	Acknowledge of the 2 <sup>nd</sup> and following monitor bytes
'0'	'1' (repeated)	Transmitter waits for acknowledge to the 1 <sup>st</sup> 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.



A omitted, if the receiver sends acknowledge immediately

Figure 6.19: GCI protocol for monitor bytes receiving

- 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  $\mu s$  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  $\mu s$  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 <sup>st</sup> 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)	0x10	
<b>PCM time slot selection</b>			
<p>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.</p>			
Bits	Reset value	Name	Description
0	0	V_SL_DIR	<b>PCM time slot data direction</b> '0' = transmit PCM data '1' = receive PCM data
7..1	0x00	V_SL_NUM	<b>PCM time slot number</b>

R_PCM_MD0	(w)	0x14	
<b>PCM mode, register 0</b>			
Bits	Reset value	Name	Description
0	0	<b>V_PCM_MD</b>	<b>PCM bus mode</b> '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	<b>V_C4_POL</b>	<b>Polarity of C4IO clock</b> '0' = pin F0IO is sampled on negative clock transition of C4IO '1' = pin F0IO is sampled on positive clock transition of C4IO
2	0	<b>V_F0_NEG</b>	<b>Polarity of F0IO signal</b> '0' = positive pulse '1' = negative pulse
3	0	<b>V_F0_LEN</b>	<b>Duration of F0IO signal</b> '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	<b>V_PCM_IDX</b>	<b>Index value to select the register at address 15</b> 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)	0x15	
<b>Slot selection register for pin F1_0</b>			
This multi-register is selected with bitmap V_PCM_IDX = 0 in register R_PCM_MD0.			
<b>Note:</b> By setting all 8 bits to '1' pin F1_0 is disabled.			
Bits	Reset value	Name	Description
6..0	0x7F	V_SL_SELO	<b>PCM time slot selection</b> 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.
7	1	V_SH_SELO	<b>Shape selection</b> '0' = use shape 0 set by registers R_SH0L and R_SH0H '1' = use shape 1 set by registers R_SH1L and R_SH1H



### 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.

<b>R_SL_SEL1</b>	(w)	0x15	
<p><b>Slot selection register for pin F1_1</b></p> <p>This multi-register is selected with bitmap V_PCM_IDX = 1 in register R_PCM_MD0.</p> <p><b>Note:</b> By setting all 8 bits to '1' pin F1_1 is disabled.</p>			
Bits	Reset value	Name	Description
6..0	0x7F	<b>V_SL_SEL1</b>	<p><b>PCM time slot selection</b></p> <p>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.</p>
7	1	<b>V_SH_SEL1</b>	<p><b>Shape selection</b></p> <p>'0' = use shape 0 set by registers R_SH0L and R_SH0H</p> <p>'1' = use shape 1 set by registers R_SH1L and R_SH1H</p>

<b>R_SL_SEL7</b>	(w)	0x15	
<p><b>Slot selection register for signal F1_7</b></p> <p>This multi-register is selected with bitmap V_PCM_IDX = 7 in register R_PCM_MD0.</p> <p>The F1_7 signal is only internally available. It can be used to shift the synchronization signal for Universal ISDN Ports in NT mode.</p>			
Bits	Reset value	Name	Description
6..0	0x7F	<b>V_SL_SEL7</b>	<p><b>PCM time slot selection</b></p> <p>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.</p>
7	0	(reserved)	Must be '0'.

R_MSS0	(w)	0x15	
<b>PCM multiframe/superframe synchronization mode, register 0</b>			
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	<b>F0IO pulse modification</b> 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	<b>F0IO modification repetition rate</b> 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	<b>Enable external multiframe / superframe synchronization signal</b> '0' = disabled '1' = enabled
3	0	V_MSS_SRC_GRD	<b>Synchronization guard count</b> 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	<b>Enable line interface synchronization signal for the multiframe / superframe</b> '0' = the synchronization signal is disabled '1' = the synchronization signal is passed to the line interfaces
5	0	V_MSS_OUT_REP	<b>Repetition rate of the line interface synchronization signal for the multiframe / superframe</b> 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	<b>Multiframe / superframe synchronization source</b> One of the following signals can be chosen as synchronization source. '00' = pin FIO '01' = SYNC_I '10' = STIO1 '11' = internal signal MSS_SYNC_I

R_PCM_MD1	(w)	0x15	
<b>PCM mode, register 1</b>			
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		<b>V_PCM_OD</b>	<b>Characteristic of the PCM output lines</b> '0' = STIO1 and STIO2 have push / pull characteristic '1' = STIO1 and STIO2 have open drain characteristic (pull up resistor required)
3..2	0	<b>V_PLL_ADJ</b>	<b>DPLL adjust speed</b> '00' = C4IO clock is adjusted in the last time slot of the PCM frame 4 times by one half clock cycle of the system clock $f_{SYS}$ '01' = C4IO clock is adjusted in the last time slot of the PCM frame 3 times by one half clock cycle of the system clock $f_{SYS}$ '10' = C4IO clock is adjusted in the last time slot of the PCM frame twice by one half clock cycle of the system clock $f_{SYS}$ '11' = C4IO clock is adjusted in the last time slot of the PCM frame once by one half clock cycle of the system clock $f_{SYS}$
5..4	0	<b>V_PCM_DR</b>	<b>PCM data rate</b> '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	<b>V_PCM_LOOP</b>	<b>PCM test loop</b> When this bit is set, the PCM output data is looped to the PCM input data internally for all PCM time slots.
7	0	<b>V_PCM_SMPL</b>	<b>PCM receive sample point</b> '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)	0x15	
<b>PCM mode, register 2</b>			
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	<b>V_SYNC_OUT1</b>	<b>SYNC_O output signal selection</b> 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	<b>V_SYNC_SRC</b>	<b>PCM PLL synchronization source selection</b> '0' = line interface (see R_SU_SYNC for further synchronization configuration) '1' = SYNC_I input (8 kHz)
3	0	<b>V_SYNC_OUT2</b>	<b>SYNC_O output signal selection</b> 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	<b>V_C2O_EN</b>	<b>Enable C2IO output signal</b> '0' = C2IO output is disabled '1' = C2IO output is enabled when also V_C2I_EN = '0'
5		<b>V_C2I_EN</b>	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	<b>V_PLL_ICR</b>	<b>Increase PCM frame time</b> 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	<b>Manual PLL adjustment</b> '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. <b>Note:</b> Manual PLL adjustment is automatically disabled when a synchronization pulse is available.

R_MSS1	(w)	0x15	
<b>PCM multiframe/superframe synchronization mode, register 1</b>			
This multi-register is selected with bitmap V_PCM_IDX = 0xB in register R_PCM_MD0.			
Bits	Reset value	Name	Description
2..0	0	<b>V_MSS_OFFS</b>	<b>Synchronization source offset</b> 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	<b>V_MS_SSYNC1</b>	<b>Multiframe/superframe single synchronization pulse</b> '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) <b>Note:</b> When this bit is set to '1', usually also V_MS_SSYNC2 has to be set.
7..4	0	<b>V_MSS_DLY</b>	<b>Multiframe/superframe delay</b> 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

<b>R_SH0L</b>	(w)	<b>0x15</b>	
<p><b>CODEC enable signal F1_0 , low byte</b></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	<b>V_SH0L</b>	<b>Shape bits 7..0</b> Every bit is used for 1/2 C4IO clock cycle.

<b>R_SH0H</b>	(w)	<b>0x15</b>	
<p><b>CODEC enable signal F1_0 , high byte</b></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	<b>V_SH0H</b>	<b>Shape bits 15..8</b> Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH0H defines the value for the rest of the period.

<b>R_SH1L</b>	(w)	<b>0x15</b>	
<p><b>CODEC enable signal F1_1 , low byte</b></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	<b>V_SH1L</b>	<b>Shape bits 7..0</b> Every bit is used for 1/2 C4IO clock cycle.

<b>R_SH1H</b>	(w)	<b>0x15</b>	
<p><b>CODEC enable signal F1_1 , high byte</b></p> <p>This multi-register is selected with bitmap V_PCM_IDX = 0xF in register R_PCM_MD0.</p>			
<b>Bits</b>	<b>Reset value</b>	<b>Name</b>	<b>Description</b>
7..0	0x00	<b>V_SH1H</b>	<p><b>Shape bits 15..8</b></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)	0x17	
<b>ST/Up synchronization source</b>			
This register selects the synchronization source for the internal or external PCM clock PLL.			
Bits	Reset value	Name	Description
2..0	0	<b>V_SYNC_SEL</b>	<b>Synchronization source selection</b> 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	<b>V_MAN_SYNC</b>	<b>Automatically synchronization source selection</b> '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	<b>V_AUTO_SYNCI</b>	<b>Enable SYNC_I as synchronization signal</b> 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	<b>V_D_MERGE_TX</b>	Both D-channels are taken from one byte in TX direction.
6	0	<b>V_E_MERGE_RX</b>	Both E-channels are combined into one Byte in RX direction.
7	0	<b>V_D_MERGE_RX</b>	Both D-channels are combined into one Byte in RX direction.

<b>R_CI_TX</b>		(w)	0x28
<b>C/I-channel of the GCI interface</b>			
Bits	Reset value	Name	Description
5..0	0	<b>V_GCI_C</b>	<b>Command bits of the C/I-channel</b> 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)	0x29	
<b>GCI interface configuration, register 0</b>			
Bits	Reset value	Name	Description
0	0	<b>V_MON_END</b>	<b>End of command flag for the monitor channel</b> The transmitted monitor command ends after the next monitor byte.
1	0	<b>V_MON_SLOW</b>	<b>Transmission speed of the monitor channel</b> '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	<b>V_MON_DLL</b>	<b>Enable double last look criterion</b> '0' = a received monitor byte is accepted at once '1' = a monitor byte is accepted only if it is received twice
3	0	<b>V_MON_C16</b>	<b>Expand C/I-channel width to 6 bits</b> '0' = 4 bit C/I-channel '1' = 6 bit C/I-channel (no D-channel transmission)
4	0	<b>V_GCI_SWAP_TXHS</b>	<b>Swap handshake bits for transmitted monitor bytes</b> '0' = normal operation (XHFC-2SU is GCI master) '1' = handshake bits MR <sub>TX</sub> and MX <sub>TX</sub> are swapped (XHFC-2SU is GCI slave) <b>Note:</b> 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	<b>V_GCI_SWAP_RXHS</b>	<b>Swap handshake bits for received monitor bytes</b> '0' = normal operation (XHFC-2SU is GCI master) '1' = handshake bits MR <sub>RX</sub> and MX <sub>RX</sub> are swapped (XHFC-2SU is GCI slave) <b>Note:</b> 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	<b>V_GCI_SWAP_STIO</b>	<p><b>Swap STIO pins for monitor bytes and C/I-channel</b></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><b>Note:</b> 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	<b>V_GCI_EN</b>	<p><b>Enable GCI interface function</b></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)	0x2A
<b>GCI interface configuration, register 1</b>			
Bits	Reset value	Name	Description
4..0	0	V_GCI_SL	<p><b>Slot group of the GCI interface</b>                      The GCI interface allocates four PCM time slots <math>4 \cdot V\_GCI\_SL \dots 4 \cdot V\_GCI\_SL + 3</math>. 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><b>Note:</b> 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)	0x2B
<b>Monitor data byte</b>			
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)	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 $\mu$ s time counter This register should be read first to latch the value of register R_F0_CNTH.

R_F0_CNTH		(r)	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 $\mu$ s time counter The low byte should be read first (see register R_F0_CNTL)

R_SL_MAX		(r)	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

<b>R_CI_RX</b>		(r)	0x28
<b>CI-channel of the GCI interface</b>			
Bits	Reset value	Name	Description
5..0	0	<b>V_GCI_I</b>	<p><b>Indication bits of the C/I-channel</b>                  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_CI6 setting. Bits [5,4] are always '00' when the C/I-channel length is 4 bit.</p>
7..6	0	(reserved)	Returns '00'.

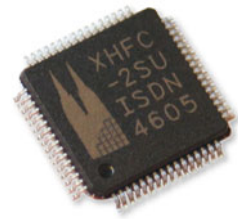
R_GCI_STA		(r)	0x29
<b>Status register of the GCI interface</b>			
Bits	Reset value	Name	Description
0	0	V_MON_RXR	<b>Monitor receiver ready</b> A monitor byte has been received and can be read from register R_MON_RX.
1	0	V_MON_TXR	<b>Monitor transmitter ready</b> A monitor byte has been send and the next byte can be written into register R_MON_TX.
2	0	V_GCI_MX	<b>Status of MX handshake bit</b> This bit shows the current status of the MX <sub>RX</sub> handshake bit (normally not used, for test purposes only).
3	0	V_GCI_MR	<b>Status of MR handshake bit</b> This bit shows the current status of the MR <sub>RX</sub> handshake bit (normally not used, for test purposes only).
4	0	V_GCI_RX	<b>Receiving monitor byte</b> '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	<b>Receiver abort</b> '0' = normal operation '1' = receiver aborted the monitor byte transmission
7..6	0	(reserved)	

R_MON_RX		(r)	0x2A
<b>Monitor data byte</b>			
Bits	Reset value	Name	Description
7..0	0	V_MON_RX	<b>Received monitor data byte</b>

### 6.9.3 Read/write register

A_SL_CFG [SLOT]		(r*/w)		0xD0
<b>HFC-channel assignment for the selected PCM time slot and PCM output buffer configuration</b>				
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	<b>HFC-channel data direction</b> '0' = HFC-channel for transmit data '1' = HFC-channel for receive data	
5..1	0x00	V_CH_SNUM	<b>HFC-channel number</b> (0..31)	
7..6	0	V_ROUT	<b>PCM output buffer configuration</b> 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	

(See Section 2.2.3.2 on page 40 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	265
0x38	R_PWM0	265
0x39	R_PWM1	266
0x46	R_PWM_MD	266

## 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.

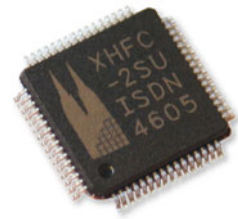
## 7.4 Register description

R_PWM_CFG		(w)	0x1E
Bits	Reset value	Name	Description
3..0	0	(reserved)	Must be '0000'.
4	0	V_PWM0_16KHZ	<b>16 kHz signal on PWM0</b> '0' = normal PWM output due to the R_PWM0 register setting '1' = 16 kHz output on PWM0
5	0	V_PWM1_16KHZ	<b>16 kHz signal on PWM1</b> '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)	0x38
<b>Modulator register for pin PWM0</b>			
Bits	Reset value	Name	Description
7..0	0x00	V_PWM0	<b>PWM duty cycle</b> 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)	0x39	
<b>Modulator register for pin PWM1</b>			
Bits	Reset value	Name	Description
7..0	0x00	<b>V_PWM1</b>	<b>PWM duty cycle</b> 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)	0x46	
<b>PWM output mode register</b>			
Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	<b>V_WAK_EN</b>	<b>Enable WAKEUP pin</b> '0' = disable WAKEUP pin '1' = enable WAKEUP pin
3..2	0	(reserved)	Must be '00'.
5..4	0	<b>V_PWM0_MD</b>	<b>Output buffer configuration for pin PWM0</b> '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	<b>V_PWM1_MD</b>	<b>Output buffer configuration for pin PWM1</b> '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	272	0x17	R_BERT_STA	273
			0x1A	R_BERT_ECL	273
			0x1B	R_BERT_ECH	274

## 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 [10] and O.151 [9] 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 70).

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

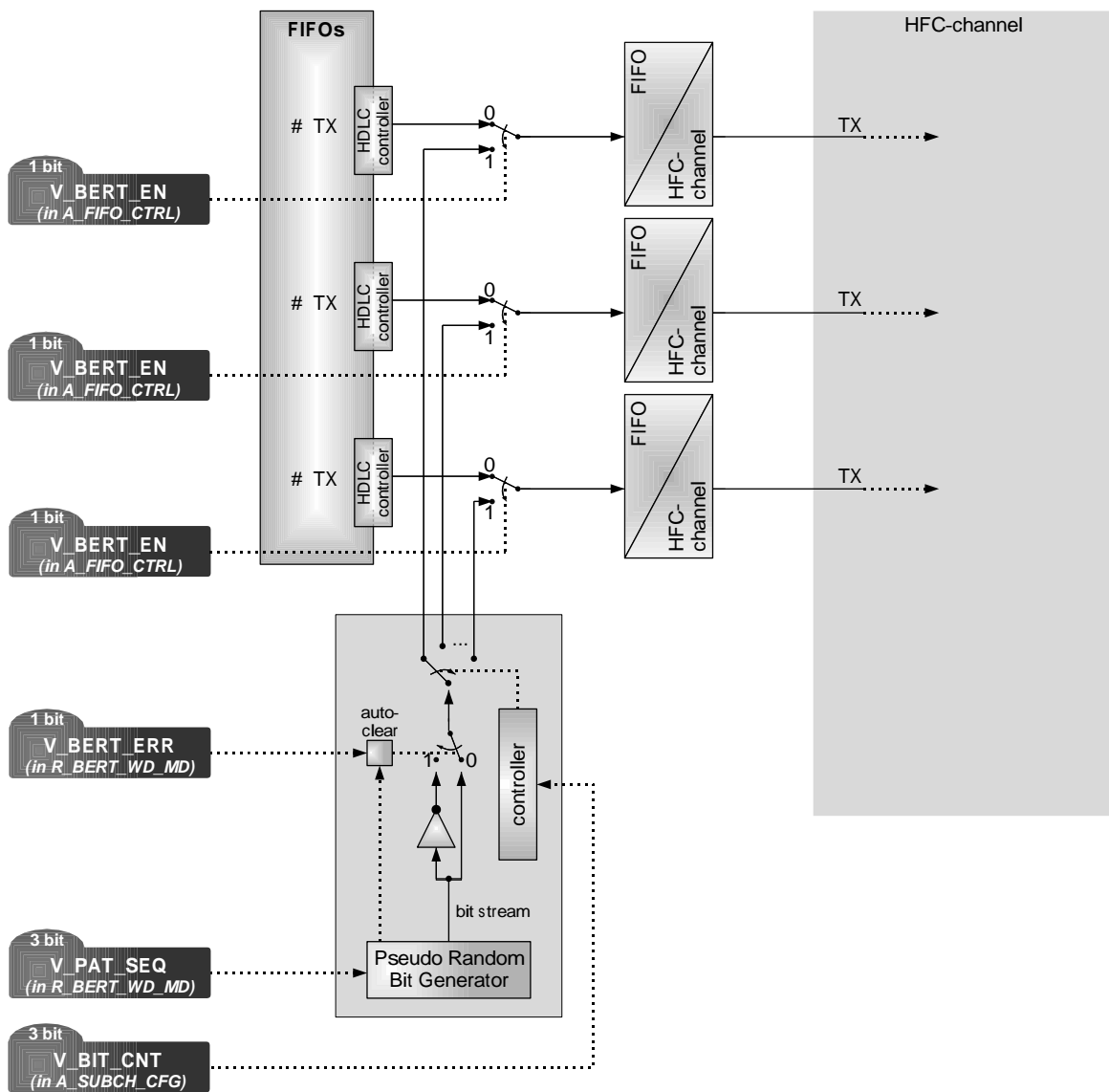


Figure 8.1: BERT transmitter block diagram

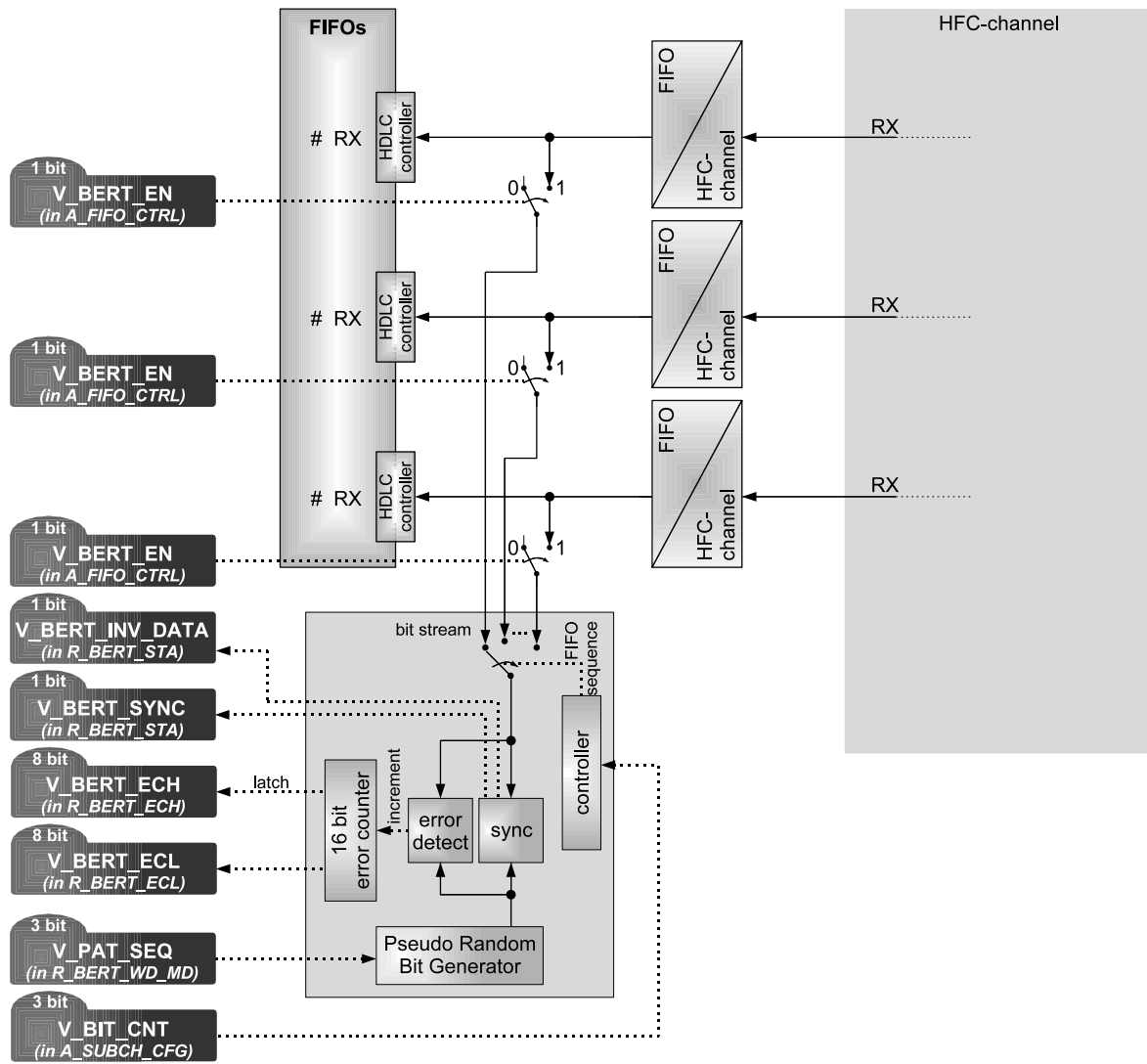


Figure 8.2: BERT receiver block diagram

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<sub>p</sub> 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 70).

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

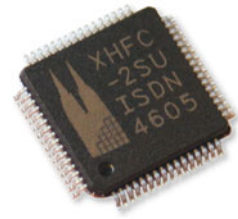
R_BERT_WD_MD		(w)	0x1B
<b>Bit error rate test (BERT) and watchdog mode</b>			
Bits	Reset value	Name	Description
2..0	0	<b>V_PAT_SEQ</b>	<b>Continuous '0' / '1' or pseudo random pattern sequence for BERT</b> '000' = continuous '0' pattern '001' = continuous '1' pattern '010' = sequence length $2^9 - 1$ 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. $2^{23} - 1$ <b>Note:</b> These sequences are defined in ITU-T O.150 and O.151 specifications.
3	0	<b>V_BERT_ERR</b>	<b>BERT error</b> 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	<b>V_AUTO_WD_RES</b>	<b>Automatic watchdog timer reset</b> '0' = watchdog is only reset by V_WD_RES '1' = watchdog is reset after every access to the chip
6	0	(reserved)	Must be '0'.
7	0	<b>V_WD_RES</b>	<b>Watchdog timer reset</b> '0' = no action '1' = manual watchdog timer reset This bit is automatically cleared.

8.4.2 Read only registers

R_BERT_STA		(r)	0x17
<b>Bit error rate test status</b>			
Bits	Reset value	Name	Description
2..0	0	V_RD_SYNC_SRC	<b>Synchronization source selection</b> 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	0	(reserved)	
4	0	V_BERT_SYNC	<b>BERT synchronization status</b> '0' = BERT not synchronized to input data '1' = BERT synchronized to input data
5	0	V_BERT_INV_DATA	<b>BERT data inversion</b> '0' = BERT receives normal data '1' = BERT receives inverted data
7..6	0	(reserved)	

R_BERT_ECL		(r)	0x1A
<b>BERT error counter, low byte</b>			
Bits	Reset value	Name	Description
7..0	0x00	V_BERT_ECL	<b>Bits 7..0 of the BERT error counter</b> This register should be read first to latch the value of register R_BERT_ECH. <b>Note:</b> The BERT counter is cleared after reading this register.

<b>R_BERT_ECH</b>	(r)	<b>0x1B</b>	
<b>BERT error counter, high byte</b>			
<b>Bits</b>	<b>Reset value</b>	<b>Name</b>	<b>Description</b>
7..0	0x00	<b>V_BERT_ECH</b>	<b>Bits 15..8 of the BERT error counter</b> <b>Note:</b> 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	293	0x10	R_IRQ_OVIEW	300	0x51	R_PLL_P	309
0x02	R_CLK_CFG	295	0x11	R_MISC_IRQ	302	0x52	R_PLL_N	309
0x11	R_MISC_IRQMSK	296	0x12	R_SU_IRQ	303	0x53	R_PLL_S	309
0x12	R_SU_IRQMSK	296	0x1C	R_STATUS	304			
0x13	R_IRQ_CTRL	297	0x20	R_FIFO_BL0_IRQ	305			
0x1A	R_TI_WD	298	0x21	R_FIFO_BL1_IRQ	306			
0x50	R_PLL_CTRL	299	0x22	R_FIFO_BL2_IRQ	307			
			0x23	R_FIFO_BL3_IRQ	308			
			0x50	R_PLL_STA	308			

## 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  $f_{\text{SRC}}$ .  $V\_CLK\_F1 = '0'$  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  $f_{\text{SYS}}$  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  $f_{\text{SYS}}$  and  $V\_CLK\_PLL$  must be set to '1' in register R\_CLK\_CFG.

Both the line interface clock  $f_{\text{SU}}$  and the PCM clock  $f_{\text{PCM}}$  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  $f_{\text{SYS}}$ ,  $f_{\text{SU}}$  and  $f_{\text{PCM}}$  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.

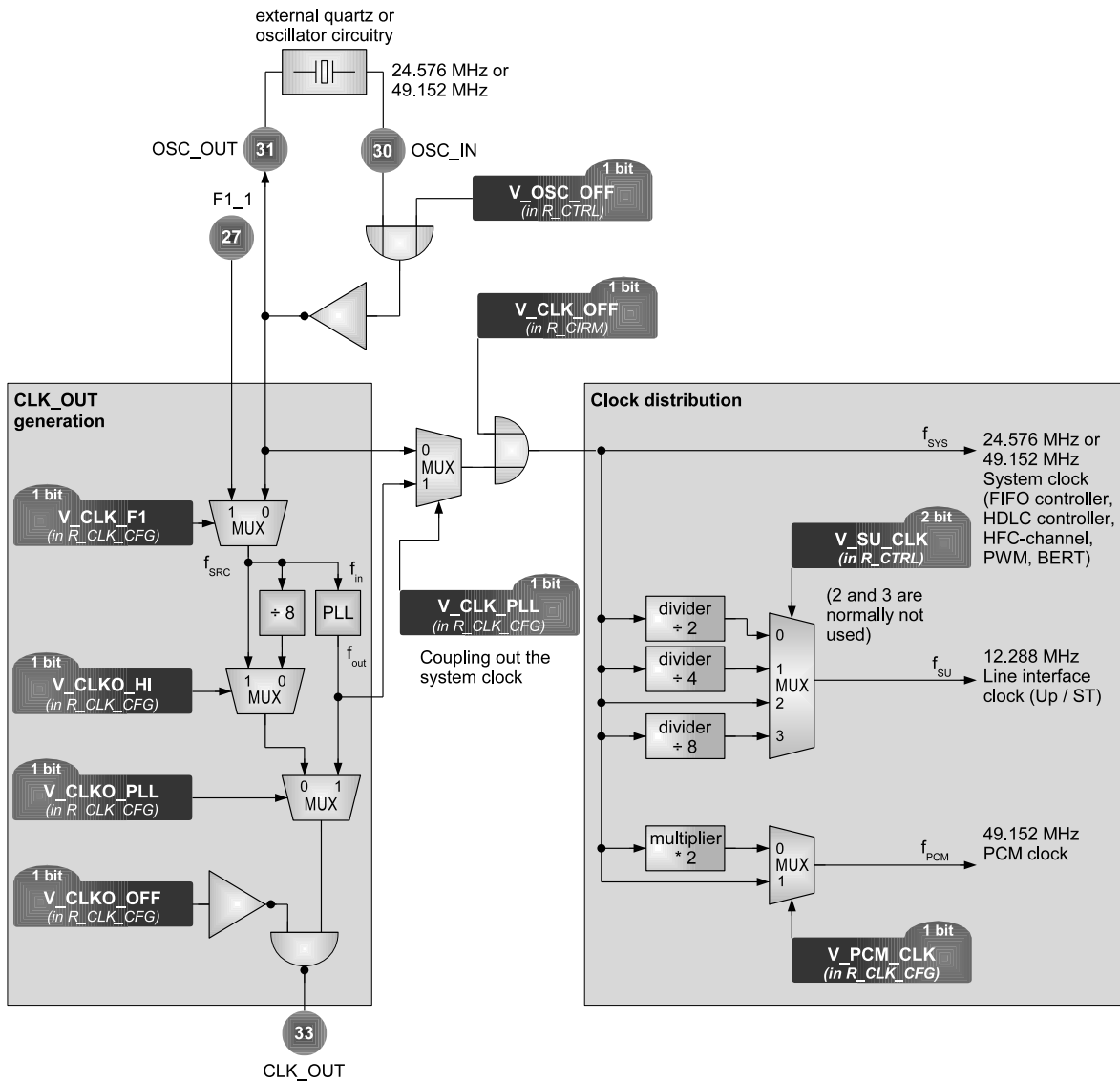


Figure 9.1: CLK\_OUT generation and clock distribution



### 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

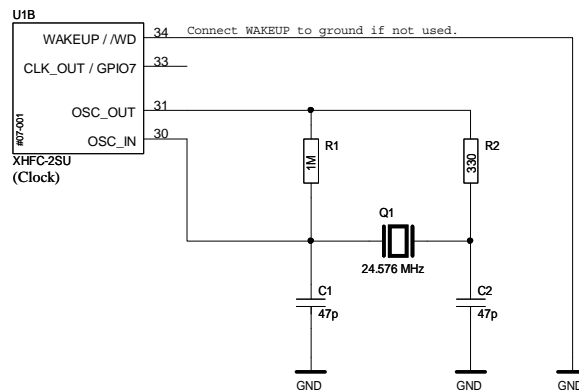
There are different ways to provide the internal clocks of XHFC-2SU. This section describes the Pierce oscillator circuitry, gives a hint to 3rd overtone oscillator and the usage of crystal oscillator circuitries.

### 9.1.3.1 Frequency accuracy

ISDN applications need an exact clock frequency. By the ISDN specification a precision of  $\pm 100$  ppm is minimum requirement for passing the ISDN type approval. In respect to temperature dependence and ageing behavior a crystal with  $\pm 50$  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,  $C_1$  and  $C_2$  are chosen to be equal.

Finally, the resistor  $R_2$  is chosen to be roughly equal to the capacitive reactance of  $C_2$  at the frequency of oscillation.

$$R_2 \sim \frac{1}{2\pi f_{Q1} \cdot C_2}$$

$R_2$  and  $C_2$  provide a low-pass filter that prevents the circuit from oscillating at a higher harmonic of the crystal frequency.

The minimum value of  $R_2$  depends on the recommended power consumption of the crystal. A too small value may damage the crystal or shorten the lifetime. When  $R_2$  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  $R_2$  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  $C_L = 30$  pF and a stray capacitance of  $C_S = 5$  pF. This leads to

$$C_1 = C_2 = 2 \cdot (C_L - C_S) = 59 \text{ pF} \sim 47 \text{ pF}$$

and

$$R_2 = \frac{3}{2\pi f_{Q1} \cdot C_2} = 413 \Omega \sim 330 \Omega .$$

### 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.

## 9.2 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<sup>1</sup>.

#### Technical features:

- Oscillator frequency range  $54\text{MHz} \leq f_{OSC} \leq 108\text{MHz}$
- Programmable loop multiplication  $5 \leq N \leq 255$
- Programmable predivider  $P$  and post-scaler  $S$  with range 1..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 9.3.

The PLL loop with input frequency  $f_{ref}$  and output frequency  $f_{PLL}$  has the ratio

$$\frac{f_{PLL}}{f_{ref}} = N$$

with  $N = V\_PLL\_N = 5..255$  (0..4 are not allowed).

The overall frequency ratio

$$\frac{f_{out}}{f_{in}} = \frac{N}{P \cdot S}$$

can be adjusted with the predivider and the post-scaler. Both dividers operate in the range 1..256.

<sup>1</sup>Detailed 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.

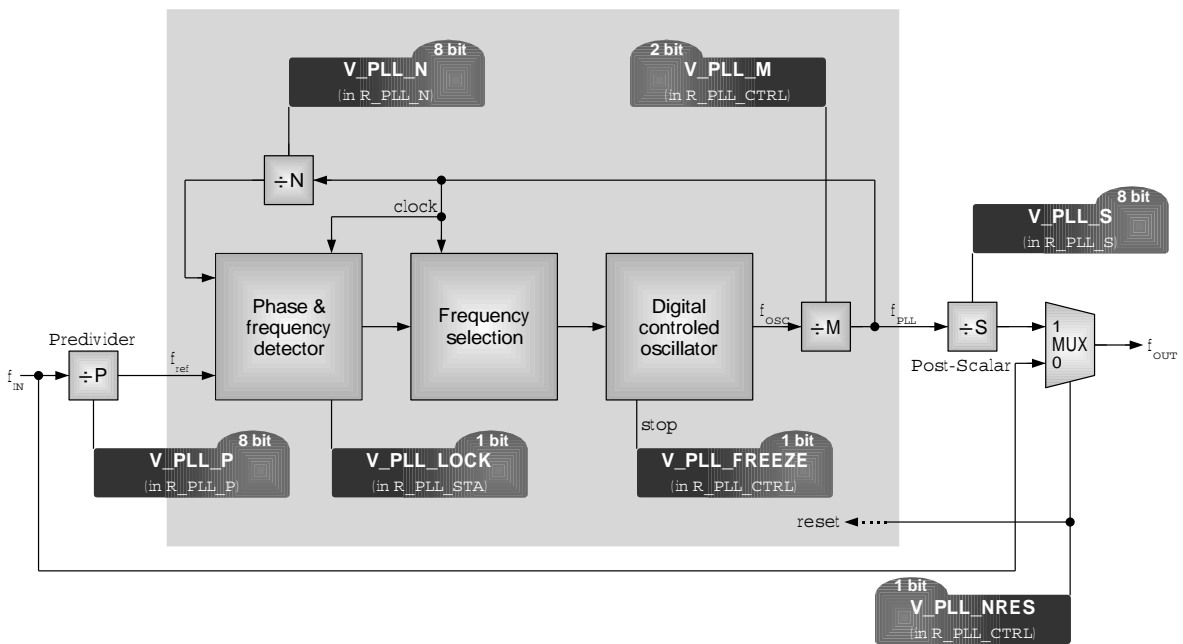


Figure 9.3: PLL block diagram

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 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  $\pm 100$  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) <b>+33 ppm</b> , +117 Hz	(10, 1, 67, 23) <b>+1 ppm</b> , +3 Hz	(20, 1, 67, 23) <b>+1 ppm</b> , +3 Hz	(40, 1, 67, 23) <b>+1 ppm</b> , +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) <b>+22 ppm</b> , +225 Hz	(89, 4, 155, 2) <b>+22 ppm</b> , +225 Hz	(89, 2, 155, 4) <b>+22 ppm</b> , +225 Hz	(89, 1, 155, 8) <b>+22 ppm</b> , +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) <b>+33 ppm</b> , +465 Hz	(23, 1, 134, 5) <b>+1 ppm</b> , +12 Hz	(23, 1, 67, 5) <b>+1 ppm</b> , +12 Hz	(46, 1, 67, 5) <b>+1 ppm</b> , +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) <b>+35 ppm</b> , +852 Hz	(29, 4, 59, 1) <b>-12 ppm</b> , -276 Hz	(29, 2, 59, 2) <b>-12 ppm</b> , -276 Hz	(29, 1, 59, 4) <b>-12 ppm</b> , -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) <b>+99 ppm</b> , +3244 Hz	(89, 3, 239, 1) <b>-58 ppm</b> , -1888 Hz	(89, 1, 239, 2) <b>-58 ppm</b> , -1888 Hz	(71, 1, 143, 3) <b>-38 ppm</b> , -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) <b>-135 ppm</b> , -8889 Hz	(35, 1, 188, 1) <b>+63 ppm</b> , +4115 Hz	(89, 1, 239, 1) <b>-58 ppm</b> , -3776 Hz	(178, 1, 239, 1) <b>-58 ppm</b> , -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  $\mu$ s with  $f_{SYS} = 24.576$  MHz. When the initialization process is finished, bit V\_BUSY changes from '1' to '0'.

PCM initialization takes 149  $\mu$ 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 soft resets. The FIFO registers, PCM registers and ST/U<sub>p</sub> registers can be reset independently with the bits of register R\_CIRM which are listed in Table 9.4. The reset bits must be set and cleared by software.

A hardware reset implies the soft reset, of course.

**Table 9.4:** XHFC-2SU reset groups

Reset name	Reset group	Register bit	Description
Soft reset	0	V_SRES	Reset for FIFO, PCM and ST/U <sub>p</sub> registers of XHFC-2SU. Soft reset is the same as reset of all partial reset registers.
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 <sub>p</sub> reset	3	V_SU_RES	Reset for all ST/U <sub>p</sub> registers of XHFC-2SU.
Hardware reset	H	–	Hardware reset initiated by /RES input pin.

Information about the allocation of the registers to the different reset groups can be found in the register list on pages 18 and 20. Many registers are allocated to more than one reset group, and some are not resettable by software.

## 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.

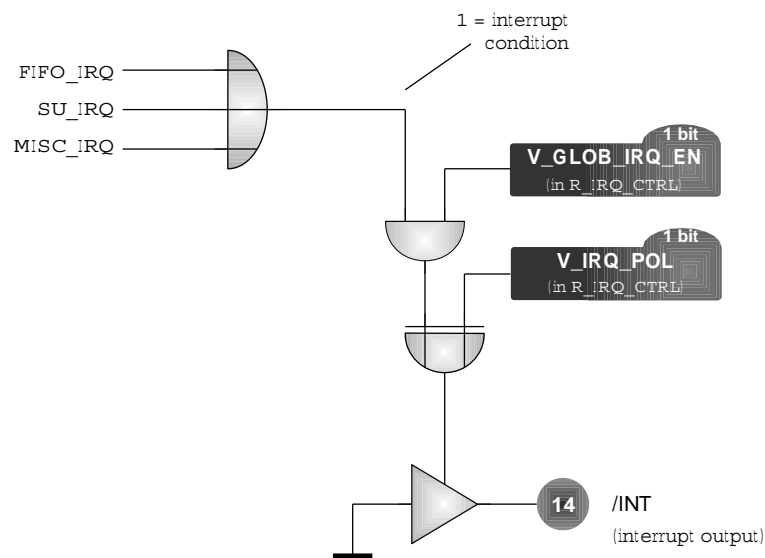


Figure 9.4: Interrupt output

### 9.4.2 ST/U<sub>p</sub> 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<sub>p</sub> 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.

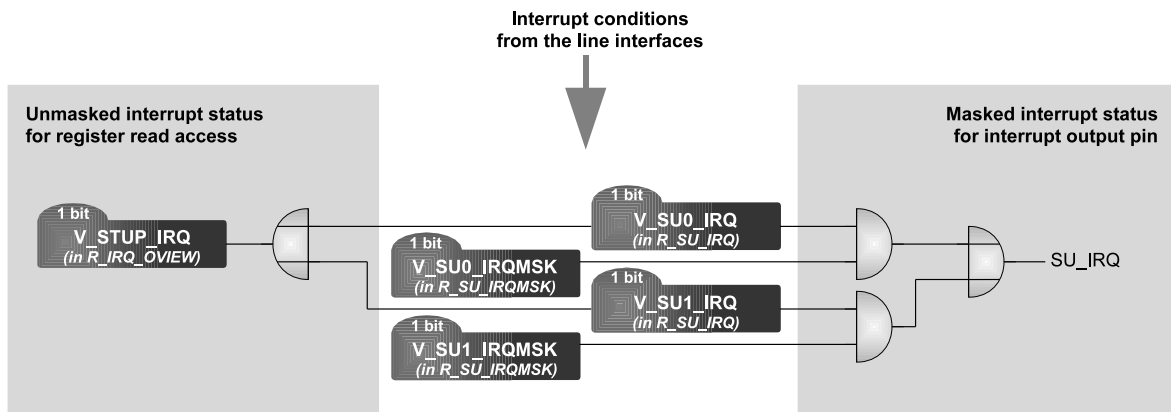


Figure 9.5: ST/U<sub>p</sub> interface interrupt

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 9.6.

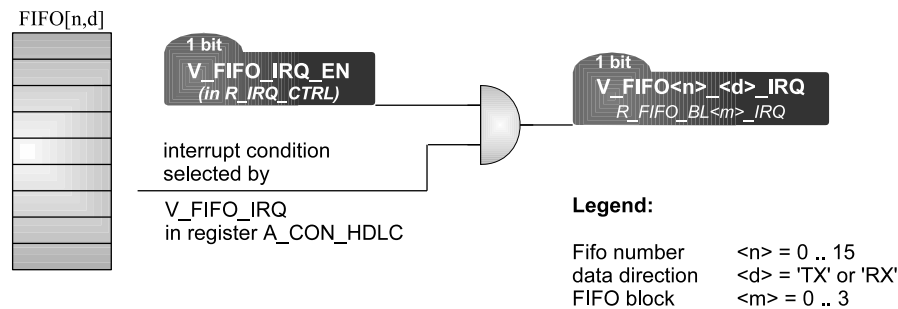


Figure 9.6: Enable FIFO interrupt condition with V\_FIFO\_IRQ\_EN

**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.

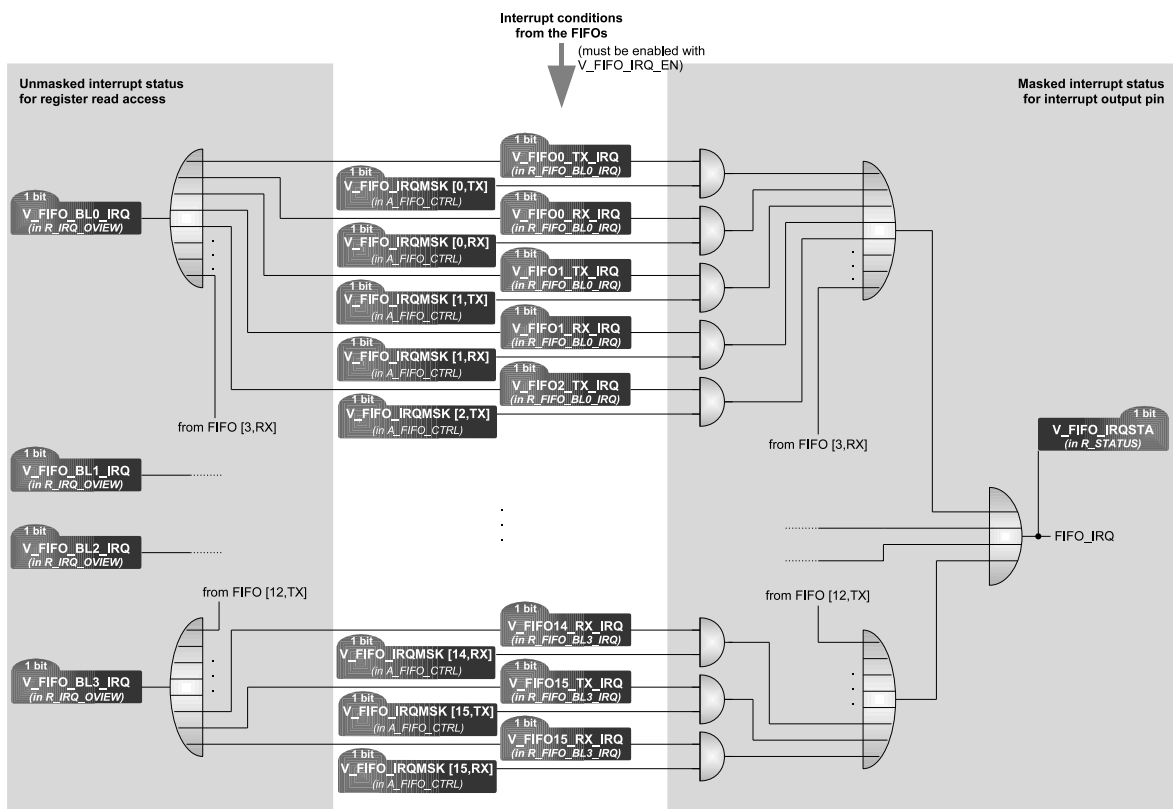


Figure 9.7: FIFO interrupt

## 9.4.4 Miscellaneous interrupts

Seven miscellaneous interrupts are available to report important XHFC-2SU status. Figure 9.8 shows the block diagram of these interrupt capabilities.

An overview bit `V_MISC_IRQ` can be read from register `R_IRQ_OVIEW`.

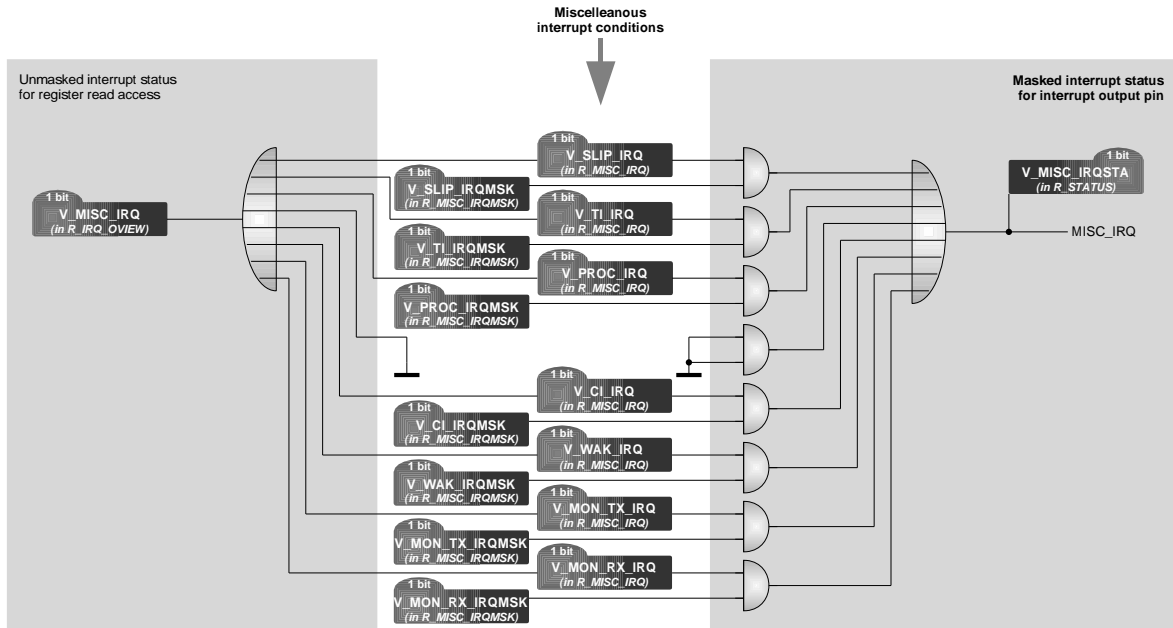


Figure 9.8: Miscellaneous interrupts

### 9.4.4.1 Line interface frequency slip interrupt

The frame synchronization signal FSC can either be the F0IO or the AF0 signal as shown in Figure 5.16 on page 178. 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 177.

An unused line interface might trigger interrupts because of the free running 8 kHz frame signal. 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`.

#### 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  $\mu$ 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  $2^{V\_EV\_TS} \cdot 250 \mu\text{s}$  where  $V\_EV\_TS = 0..15$  in register R\_TI\_WD. 16 timer event frequencies are available in the range 250  $\mu$ s .. 8.192 s.

#### 9.4.4.3 Processing / non-processing interrupt

XHFC-2SU changes every 125  $\mu$ 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  $\mu$ 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 9.9. 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.

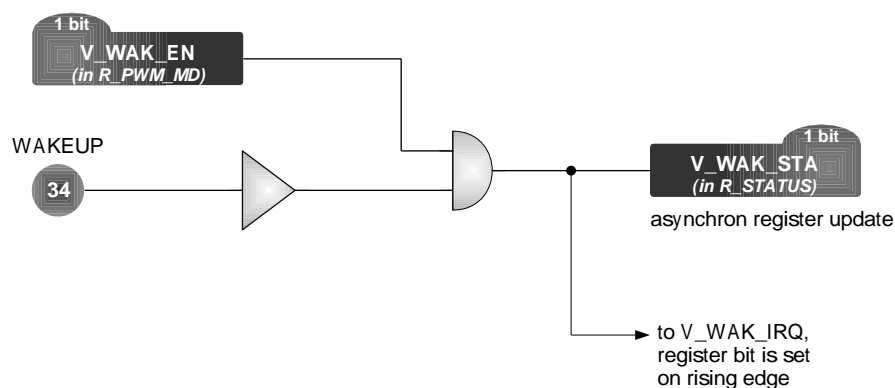


Figure 9.9: Wakeup interrupt

#### **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)	0x00	
<b>Interrupt and reset register</b>			
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	<b>Global clock enable / disable</b> '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	<b>Additional /WAIT signal after write access</b> 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
2	0	V_WAIT_REG	<b>Additional /WAIT signal during internal busy phase</b> 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
3	0	V_SRES	<b>Soft reset (reset group 0)</b> This reset is similar to the hardware reset. The selected I/O address (CIP) remains unchanged. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
4	0	V_HFC_RES	<b>HFC-reset (reset group 1)</b> 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

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
5	0	V_PCM_RES	<b>PCM reset (reset group 2)</b> 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	<b>Line interface reset (reset group 3)</b> 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'.

Bits	Reset value	Name	Description
<b>Clock configuration register</b>			
0	0	<b>V_CLK_PLL</b>	<b>System clock selection</b> '0' = system clock is pin OSC_IN '1' = system clock is PLL output
1	0	<b>V_CLKO_HI</b>	<b>High/low frequency selection for clock output</b> The clock output at pin CLK_OUT is either the PLL input clock (high frequency) or the PLL input clock divided by 8 (low frequency) '0' = low frequency, divided by 8 '1' = high frequency, no divider
2	0	<b>V_CLKO_PLL</b>	<b>Source selection for clock output</b> '0' = divider output clock or PLL input clock is used '1' = PLL output clock is used
4..3	0	(reserved)	Must be '000'.
5	0	<b>V_PCM_CLK</b>	<b>Clock of the PCM module</b> '0' = $f_{PCM} = 2 \cdot f_{SYS}$ '1' = $f_{PCM} = f_{SYS}$ PCM clock must be set up to 49.152 MHz.
6	0	<b>V_CLKO_OFF</b>	<b>clock output enable/disable</b> '0' = clock output pin CLK_OUT is enabled '1' = clock output pin CLK_OUT is tristated
7	0	<b>V_CLK_F1</b>	<b>Pin selection for PLL input frequency</b> '0' = oscillator input OSC_IN is used '1' = F1_1 input pin is used

<b>R_MISC_IRQMSK</b>	(w)	0x11	
<b>Miscellaneous interrupt mask register</b>			
<p>Any miscellaneous interrupt can be enabled with bit value '1' individually. '0' means that the interrupt is not used for generating a signal on the interrupt pin 14. The interrupt status can be read from register R_MISC_IRQ nevertheless.</p>			
Bits	Reset value	Name	Description
0		<b>V_SLIP_IRQMSK</b>	<b>Interrupt mask of the line interface frequency slip</b>
1	0	<b>V_TI_IRQMSK</b>	<b>Timer interrupt mask</b>
2	0	<b>V_PROC_IRQMSK</b>	<b>Processing / nonprocessing transition interrupt mask (every 125 µs)</b>
3	0	(reserved)	Must be '0'.
4	0	<b>V_CI_IRQMSK</b>	<b>Command / indication interrupt mask</b>
5		<b>V_WAK_IRQMSK</b>	<b>Wakeup interrupt mask</b>
6		<b>V_MON_TX_IRQMSK</b>	<b>Transmit monitor byte interrupt mask</b>
7		<b>V_MON_RX_IRQMSK</b>	<b>Receive monitor byte interrupt mask</b>

<b>R_SU_IRQMSK</b>	(w)	0x12	
<b>State change interrupt mask register of the line interfaces</b>			
<p>The line interface interrupt can be enabled with bit value '1' for every interface individually. '0' means that the interrupt is not used for generating a signal on the interrupt pin 14. The interrupt status can be read from register R_SU_IRQ nevertheless.</p>			
Bits	Reset value	Name	Description
0	0	<b>V_SU0_IRQMSK</b>	<b>Interrupt mask of line interface 0</b> '1' = V_SU0_IRQ in register R_SU_IRQ is used for interrupt generation
1	0	<b>V_SU1_IRQMSK</b>	<b>Interrupt mask of line interface 1</b> '1' = V_SU1_IRQ in register R_SU_IRQ is used for interrupt generation
7..2		(reserved)	

R_IRQ_CTRL		(w)	0x13
<b>Interrupt control register</b>			
Bits	Reset value	Name	Description
0	0	<b>V_FIFO_IRQ_EN</b>	<b>FIFO interrupt</b> '0' = all FIFO interrupts disabled '1' = all FIFO interrupts enabled
2..1	0	(reserved)	Must be '00'.
3	0	<b>V_GLOB_IRQ_EN</b>	<b>Global interrupt signal enable</b> The interrupt line is pin 14. '0' = disable '1' = enable
4	0	<b>V_IRQ_POL</b>	<b>Polarity of interrupt signal</b> '0' = active low signal '1' = active high signal
7..5	0	(reserved)	Must be '000'.

R_TI_WD	(w)	0x1A	
<b>Timer and watchdog control register</b>			
Bits	Reset value	Name	Description
3..0	0	<b>V_EV_TS</b>	<b>Timer event after <math>2^n \cdot 250 \mu\text{s}</math></b> 0 = 250 $\mu\text{s}$ 1 = 500 $\mu\text{s}$ 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	<b>V_WD_TS</b>	<b>Watchdog event after <math>2^n \cdot 2 \text{ ms}</math></b> 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)		0x50
<b>PLL control register</b>				
Bits	Reset value	Name	Description	
0	0	<b>V_PLL_NRES</b>	<b>PLL reset (active low)</b> '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	<b>V_PLL_TST</b>	<b>PLL test input</b> '0' = test state disabled, normal operation '1' = test state enabled	
4..2	0	(reserved)	Must be '000'.	
5	0	<b>V_PLL_FREEZE</b>	<b>PLL standby mode</b> '0' = normal PLL operation '1' = PLL is in standby mode, no output clock	
7..6	0	<b>V_PLL_M</b>	<b>Oscillator divider programming value</b> The oscillator output signal is divided by $V\_PLL\_M + 1$ .	

## 9.6.2 Read only registers

R_IRQ_OVIEW	(r)	0x10	
<b>FIFO interrupt overview register</b>			
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		<b>V_FIFO_BL0_IRQ</b>	<b>Interrupt overview of FIFO block 0</b> 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		<b>V_FIFO_BL1_IRQ</b>	<b>Interrupt overview of FIFO block 1</b> 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		<b>V_FIFO_BL2_IRQ</b>	<b>Interrupt overview of FIFO block 2</b> 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		<b>V_FIFO_BL3_IRQ</b>	<b>Interrupt overview of FIFO block 3</b> 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		<b>V_MISC_IRQ</b>	<b>Miscellaneous interrupt overview</b> '0' = No miscellaneous interrupt occurred '1' = At least one miscellaneous interrupt is pending. Reading register R_MISC_IRQ clears the miscellaneous interrupt bits.
5		<b>V_STUP_IRQ</b>	<b>Line interface interrupt overview</b> '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.
7..6		(reserved)	

R\_MISC\_IRQ

(r)

0x11

**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.

Bits	Reset value	Name	Description
0		V_SLIP_IRQ	<b>Interrupt status of the line interface frequency slip</b> 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	<b>Timer interrupt status</b> '1' = timer elapsed
2	0	V_PROC_IRQ	<b>Processing / non processing transition interrupt status</b> '1' = XHFC-2SU has changed from processing to non processing phase (every 125 µs). <b>Note:</b> The current processing / non-processing status can be read from V_PROC in register R_STATUS.
3	0	(reserved)	
4	0	V_CI_IRQ	<b>Command / indication interrupt status</b> '1' = received indication bits changed
5	0	V_WAK_IRQ	<b>Wakeup interrupt status</b> '1' = a wakeup signal at pin WAKEUP occurred
6		V_MON_TX_IRQ	<b>Transmit monitor byte interrupt status</b> '1' = the next monitor byte can be written
7		V_MON_RX_IRQ	<b>Receive monitor byte interrupt status</b> '1' = monitor byte received

Bits	Reset value	Name	Description
0	0	V_SU0_IRQ	<b>Interrupt status of line interface 0</b> '1' = a state change occurred in line interface 0
1	0	V_SU1_IRQ	<b>Interrupt status of line interface 1</b> '1' = a state change occurred in line interface 1
7..2	0	(reserved)	

R_STATUS	(r)	0x1C	
<b>XHFC-2SU status register</b>			
Bits	Reset value	Name	Description
0	0	<b>V_BUSY</b>	<b>BUSY / NOBUSY status</b> '0' = XHFC-2SU is not busy, all accesses are allowed '1' = XHFC-2SU is BUSY after initialising Reset FIFO, increment <i>F</i> -counter or change FIFO
1	1	<b>V_PROC</b>	<b>Processing / non processing status</b> '0' = XHFC-2SU has finished the processing phase during the 125 $\mu$ s cycle '1' = XHFC-2SU is in processing phase (once every 125 $\mu$ s cycle) <b>Note:</b> The processing / non-processing transition can be notified with an interrupt (see V_PROC_IRQ in register R_MISC_IRQ).
2	0	(reserved)	
3	0	<b>V_LOST_STA</b>	<b>LOST error (frames have been lost)</b> This means that XHFC-2SU did not process all data in 125 $\mu$ s. So data may be corrupted. Bit V_RES_LOST of register A_INC_RES_FIFO must be set to reset this bit.
4		<b>V_PCM_INIT</b>	<b>PCM module initialization</b> '0' = initialization sequence is finished '1' = initialization sequence is in progress (after reset) <b>Note:</b> The PCM clocks F0IO and C4IO are ignored during initialization.
5	0	<b>V_WAK_STA</b>	<b>Wakeup status</b> This bit contains the current value of pin WAKEUP when V_WAK_EN = '1' in register R_PWM_MD.
6	0	<b>V_MISC_IRQSTA</b>	<b>Any miscellaneous interrupt</b> All enabled miscellaneous interrupts of register R_MISC_IRQ are 'ored'. '0' = No miscellaneous interrupt is pending '1' = At least one miscellaneous interrupt is pending.
7	0	<b>V_FIFO_IRQSTA</b>	<b>Any FIFO interrupt</b> 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.

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

R\_FIFO\_BL1\_IRQ

(r)

0x21

**Interrupt status register for FIFO block 1**

This register reports the interrupt status of FIFO block 1. 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.

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_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

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

<b>R_FIFO_BL3_IRQ</b>	(r)	0x23	
<b>Interrupt status register for FIFO block 3</b>			
<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.</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

<b>R_PLL_STA</b>	(r)	0x50	
<b>PLL status register</b>			
Bits	Reset value	Name	Description
6..0		(reserved)	
7	0	V_PLL_LOCK	<b>PLL lock status</b> '0' = PLL is unlocked or in standby mode '1' = PLL is locked

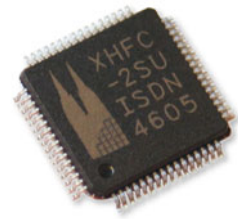
**9.6.3 Read/write registers**

<b>R_PLL_P</b>	<b>(r/w)</b>	<b>0x51</b>	
<b>PLL predivider programming value</b>			
Bits	Reset value	Name	Description
7..0	0	<b>V_PLL_P</b>	<b>Predivider programming value</b> The divisor of the predivider is V_PLL_P. The bitmap value 0 has the meaning of 256.

<b>R_PLL_N</b>	<b>(r/w)</b>	<b>0x52</b>	
<b>PLL loop factor</b>			
Bits	Reset value	Name	Description
7..0	0	<b>V_PLL_N</b>	<b>Loop factor programming value</b> The loop factor is V_PLL_N. <b>Note:</b> 0..4 are not allowed.

<b>R_PLL_S</b>	<b>(r/w)</b>	<b>0x53</b>	
<b>PLL post-scaler programming value</b>			
Bits	Reset value	Name	Description
7..0	0	<b>V_PLL_S</b>	<b>Post-scaler programming value</b> The divisor of the post-scaler is V_PLL_S. The bitmap value 0 has the meaning of 256.





## 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	315	0x40	R_GPIO_IN1	320
0x42	R_GPIO_EN1	316	0x45	R_GPIO_IN2	320
0x44	R_GPIO_SEL_BL	316	0x48	R_GPIO_IN0	321
0x45	R_GPIO_OUT2	317			
0x47	R_GPIO_EN2	317			
0x48	R_GPIO_OUT0	318			
0x4A	R_GPIO_EN0	318			
0x4C	R_GPIO_SEL	319			

## 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<sub>p</sub> 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 (1<sup>st</sup> pin function) is selected for interface #0. GPIO8 output is disabled within a group of six pins (2<sup>nd</sup> 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 (1<sup>st</sup> pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2<sup>nd</sup> 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 (1<sup>st</sup> pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2<sup>nd</sup> 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<sub>p</sub> 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<sub>p</sub> 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.

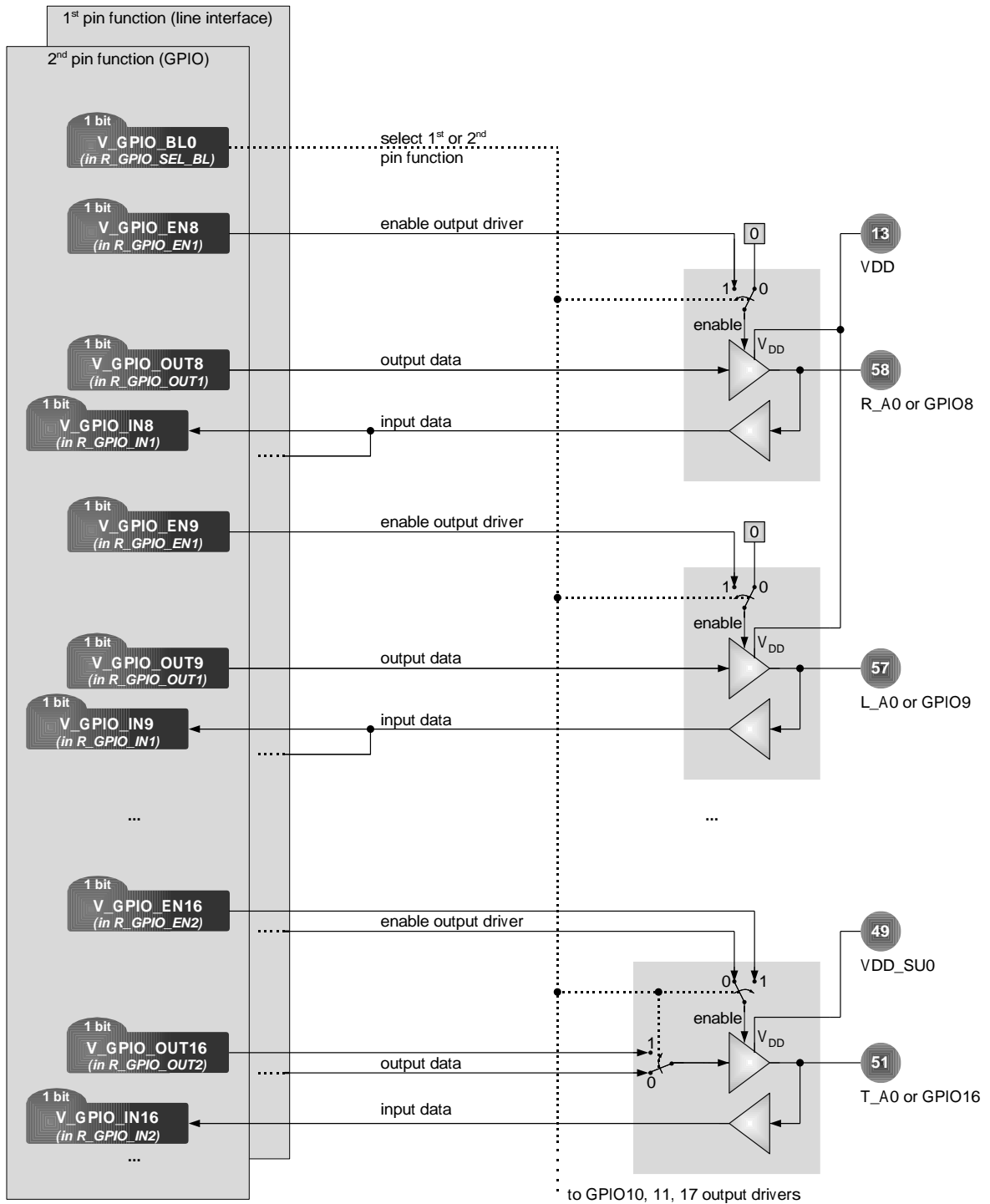


Figure 10.1: GPIO block diagram (GPIO8 , GPIO9 and GPIO16 exemplarily)

Table 10.2: GPIO pins

Pin number	GPIO pin	GPIO byte	Shared with (1 <sup>st</sup> 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 Register description

### 10.3.1 Write only registers

R_GPIO_OUT1		(w)	0x40
<b>GPIO output data bits 15..8</b>			
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)	0x42	
<b>GPIO output enable bits 15..8</b>			
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)	0x44	
<b>Selection register for GPIO block</b>			
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	<b>GPIO function on line interface no. 0</b> '0' = line interface no. 0 used (first pin function) '1' = GPIO8 .. GPIO11 , GPIO16 and GPIO17 used (second pin function)
1	0	V_GPIO_BL1	<b>GPIO function on line interface no. 1</b> '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)	0x45
<b>GPIO output data bits 19..16</b>			
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)	0x47
<b>GPIO output enable bits 19..16</b>			
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)	0x48
<b>GPIO output data bits 7..0</b>			
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

R_GPIO_EN0		(w)	0x4A
<b>GPIO output enable bits 7..0</b>			
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)	0x4C	
<b>GPIO selection register</b>			
This register controls only the output driver, whereas the input functionality needs no programming.			
Bits	Reset value	Name	Description
0	0	V_GPIO_SEL0	<b>Selection of first or second pin function</b> '0' = PWM0 (first pin function) '1' = GPIO0 (second pin function)
1	0	V_GPIO_SEL1	<b>Selection of first or second pin function</b> '0' = PWM1 (first pin function) '1' = GPIO1 (second pin function)
2	0	V_GPIO_SEL2	<b>Selection of first or second pin function</b> '0' = F1_0 (first pin function) '1' = GPIO2 (second pin function)
3	0	V_GPIO_SEL3	<b>Selection of first or second pin function</b> '0' = F1_1 (first pin function) '1' = GPIO3 (second pin function)
4	0	V_GPIO_SEL4	<b>Selection of first or second pin function</b> '0' = STIO1 (first pin function) '1' = GPIO4 (second pin function)
5	0	V_GPIO_SEL5	<b>Selection of first or second pin function</b> '0' = STIO2 (first pin function) '1' = GPIO5 (second pin function)
6	0	V_GPIO_SEL6	<b>Selection of first or second pin function</b> '0' = SYNC_O (first pin function) '1' = GPIO6 (second pin function)
7	0	V_GPIO_SEL7	<b>Selection of first or second pin function</b> '0' = CLK_OUT (first pin function) '1' = GPIO7 (second pin function)

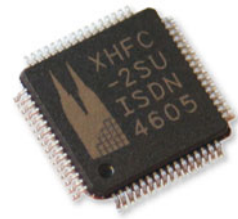
### 10.3.2 Read only registers

R_GPIO_IN1		(r)	0x40
<b>GPIO input data bits 15..8</b>			
<b>Note:</b> Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0	0	V_GPIO_IN8	Input data bit from pin GPIO8
1	0	V_GPIO_IN9	Input data bit from pin GPIO9
2	0	V_GPIO_IN10	Input data bit from pin GPIO10
3	0	V_GPIO_IN11	Input data bit from pin GPIO11
4	0	V_GPIO_IN12	Input data bit from pin GPIO12
5	0	V_GPIO_IN13	Input data bit from pin GPIO13
6	0	V_GPIO_IN14	Input data bit from pin GPIO14
7	0	V_GPIO_IN15	Input data bit from pin GPIO15

R_GPIO_IN2		(r)	0x45
<b>GPI input data bits 19..16</b>			
<b>Note:</b> Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0	0	V_GPIO_IN16	Input data bit from pin GPIO16
1	0	V_GPIO_IN17	Input data bit from pin GPIO17
2	0	V_GPIO_IN18	Input data bit from pin GPIO18
3	0	V_GPIO_IN19	Input data bit from pin GPIO19
7..4	0	(reserved)	

R_GPIO_IN0		(r)	0x48
<b>GPIO input data bits 7..0</b>			
<b>Note:</b> Unused GPIO pins should be configured as output pins.			
Bits	Reset value	Name	Description
0	0	V_GPIO_IN0	Input data bit from pin GPIO0
1	0	V_GPIO_IN1	Input data bit from pin GPIO1
2	0	V_GPIO_IN2	Input data bit from pin GPIO2
3	0	V_GPIO_IN3	Input data bit from pin GPIO3
4	0	V_GPIO_IN4	Input data bit from pin GPIO4
5	0	V_GPIO_IN5	Input data bit from pin GPIO5
6	0	V_GPIO_IN6	Input data bit from pin GPIO6
7	0	V_GPIO_IN7	Input data bit from pin GPIO7





## Chapter 11

# Electrical characteristics

**Absolute maximum ratings** <sup>\*1</sup>

Parameter	Symbol	Min.	Max.
Power supply	$V_{DD}$	-0.3 V	+4.6 V
Input voltage	$V_I$	-0.3 V	6.0 V <sup>*2</sup>
Operating temperature	$T_{opr}$	-30 °C	+70 °C
Junction temperature	$T_{jnc}$	0 °C	+100 °C
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	$T_{opr}$	0 °C		+70 °C	

**Electrical characteristics for 3.3 V power supply**

Parameter	Symbol	Min.	Typ.	Max	Conditions
Low input voltage <sup>*3</sup>	$V_{IL}$	-0.5 V		0.8 V	
High input voltage <sup>*4</sup>	$V_{IH}$	2.0 V		$V_{DD}$	
High input voltage <sup>*5</sup>	$V_{IH}$	2.0 V		5.5 V	
Low output voltage <sup>*3</sup>	$V_{OL}$			0.4 V	
High output voltage <sup>*3</sup>	$V_{OH}$	2.4 V			

<sup>\*1</sup>: 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.

<sup>\*2</sup>: Maximum voltage for oscillator pins is  $V_{DD}$ .

<sup>\*3</sup>: All pins except oscillator and S/T/ $U_p$  type.

<sup>\*4</sup>: All pins except oscillator, S/T/ $U_p$  type and 5 V tolerant pins.

<sup>\*5</sup>: Only 5 V tolerant pins.

**Power consumption for 3.3 V power supply**

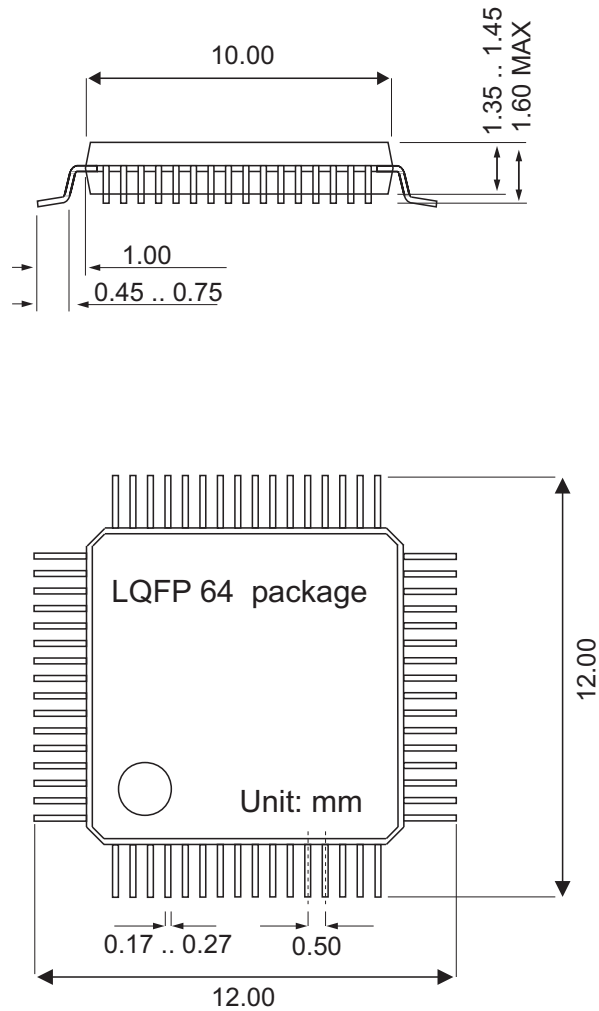
Parameter	Symbol	Min.	Typ.	Max	Conditions
Oscillator running with 24.576 MHz, internal clock disabled	$I_{opr}$		4 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running	$I_{opr}$		19 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running, both S/T interfaces sending 96 kHz test signal on 50 $\Omega$ 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





## Chapter 12

# XHFC-2SU package dimensions



**Figure 12.1:** XHFC-2SU package dimensions

# References

- [1] Cologne Chip AG. *DIGICC™ 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] Infineon Technologies. *PEB 2096 Data Sheet; ICs for Communications: Octal Transceiver for Upn Interfaces (OCTAT-P)*, 2.1 edition, April 1999.
- [6] Siemens AG. *ICs for Communication. IOM®-2 Interface Reference Guide*, 3 1991.
- [7] 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.
- [8] 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.
- [9] 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.
- [10] 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.

<b>16KHZ</b>	16 kHz	<b>CHANNEL</b>	HFC-channel	<b>E</b>	E-channel
<b>2KHZ</b>	2 kHz	<b>CHIP</b>	microchip	<b>ECH</b>	error counter, high byte
<b>AB</b>	A/B-bit	<b>CI</b>	Command/Indication (C/I channel of the GCI interface)	<b>ECL</b>	error counter, low byte
<b>ABO</b>	abort	<b>CI6</b>	6 bit C/I-channel length	<b>EN</b>	enable
<b>ACT</b>	active, activation	<b>CIRM</b>	configuration, interrupt and reset	<b>END</b>	end
<b>ADDR</b>	address	<b>CLK</b>	clock	<b>ERR</b>	error
<b>ADJ</b>	adjust	<b>CLKO</b>	clock output	<b>EV</b>	event
<b>AF0</b>	alternative frame synchronization signal	<b>CNT</b>	counter	<b>EXCHG</b>	exchange
<b>AUTO</b>	automatic	<b>CNTH</b>	counter, high byte	<b>EXP</b>	expired
<b>B1</b>	B1-channel	<b>CNTL</b>	counter, low byte	<b>F</b>	<i>F</i> -counter
<b>B12</b>	B1- and B2-channels	<b>CON</b>	connection settings	<b>F0</b>	frame synchronization signal
<b>B2</b>	B2-channel	<b>CONT</b>	contention	<b>F1</b>	F1_1 pin
<b>BAC</b>	BAC-bit	<b>CRC</b>	cyclic redundancy check	<b>F1</b>	<i>F1</i> -counter
<b>BERT</b>	bit error rate test	<b>CTRL</b>	control	<b>F2</b>	<i>F2</i> -counter
<b>BIT</b>	bit	<b>D</b>	D-channel	<b>FDIR</b>	direction (FIFO-related)
<b>BL</b>	block	<b>DATA</b>	data	<b>FIFO</b>	FIFO
<b>BUSY</b>	busy	<b>DC</b>	DC-balancing bit	<b>FILL</b>	fill level
<b>C</b>	command bits of the C/I-channel	<b>DF</b>	data flow	<b>FIRST</b>	first
<b>C2I</b>	C2 clock input (PCM bit clock)	<b>DIR</b>	direction	<b>FLOW</b>	flow
<b>C2O</b>	C2 clock output (PCM bit clock)	<b>DLL</b>	double last look criterion	<b>FNUM</b>	number (FIFO-related)
<b>C4</b>	C4IO clock (PCM double bit clock)	<b>DLY</b>	delay	<b>FR</b>	frame
<b>CFG</b>	configuration	<b>DLYH</b>	delay, high byte	<b>FREEZE</b>	freeze
<b>CH</b>	HFC-channel	<b>DLYL</b>	delay, low byte	<b>FRQ</b>	frequency
		<b>DONE</b>	done	<b>FSM</b>	FIFO sequence mode
		<b>DR</b>	data rate		
		<b>DST</b>	destination	<b>G2</b>	G2 state

<b>G3</b>	G3 state	<b>MISC</b>	miscellaneous	<b>REP</b>	repetition
<b>GCI</b>	GCI interface	<b>MIX</b>	mixed	<b>RES</b>	reset
<b>GLOB</b>	global	<b>MOD</b>	modification	<b>REV</b>	reverse
<b>GPIO</b>	general purpose input / output	<b>MON</b>	monitor channel of the GCI interface	<b>ROUT</b>	routing
<b>GRD</b>	guard	<b>MR</b>	handshake bit MR	<b>RPT</b>	repeat
<b>HDLC</b>	high-level data link control	<b>MS</b>	multiframe / superframe	<b>RV</b>	revision
<b>HFC</b>	HDLC FIFO controller	<b>MSK</b>	mask	<b>RX</b>	receive
<b>HI</b>	high	<b>MSS</b>	multiframe / superframe synchronization	<b>RXHS</b>	receiver handshake
<b>HPRIO9</b>	high priority, 9 bits	<b>MULT</b>	multiple	<b>RXR</b>	receiver ready
<b>I</b>	indication bits of the C/I-channel	<b>MX</b>	handshake bit MX	<b>S</b>	S-bit
<b>ICR</b>	increase	<b>N</b>	PLL loop factor	<b>S</b>	PLL post-scaler
<b>ID</b>	identifier	<b>NEG</b>	negative	<b>SCRM</b>	scrambler
<b>IDX</b>	index	<b>NEXT</b>	next	<b>SDIR</b>	direction (slot-related)
<b>IFF</b>	inter frame fill	<b>NINV</b>	no inversion	<b>SEL</b>	select, selection
<b>IGNO</b>	ignore	<b>NO</b>	no	<b>SEQ</b>	sequence
<b>IN</b>	input	<b>NOINC</b>	no increment	<b>SET</b>	setup
<b>INC</b>	increment	<b>NRES</b>	active low reset	<b>SG</b>	S/G-bit
<b>INFO0</b>	INFO 0 line condition (no signal)	<b>NT</b>	NT mode	<b>SH</b>	shape
<b>INIT</b>	initialization	<b>NUM</b>	number	<b>SH0H</b>	shape 0, high byte
<b>INT</b>	internal	<b>OD</b>	open drain output	<b>SH0L</b>	shape 0, low byte
<b>INV</b>	invert, inversion	<b>OFF</b>	off	<b>SH1H</b>	shape 1, high byte
<b>IRQ</b>	interrupt	<b>OFFS</b>	offset	<b>SH1L</b>	shape 1, low byte
<b>IRQMSK</b>	interrupt mask	<b>OSC</b>	oscillator	<b>SIG</b>	signal
<b>IRQSTA</b>	interrupt status	<b>OUT</b>	output	<b>SL</b>	time slot
<b>LD</b>	load	<b>OVIEW</b>	overview	<b>SLIP</b>	frequency slip
<b>LEN</b>	length	<b>P</b>	PLL predivider	<b>SLOT</b>	time slot
<b>LO</b>	low	<b>PAT</b>	pattern	<b>SLOW</b>	slow
<b>LOCK</b>	locked	<b>PCM</b>	pulse code modulation	<b>SMPL</b>	sample
<b>LOOP</b>	loop	<b>PLL</b>	phase locked loop	<b>SNUM</b>	number (slot-related)
<b>LOST</b>	frame data lost	<b>POL</b>	polarity	<b>SQ</b>	S/Q-bits
<b>LPRIO</b>	low priority	<b>PROC</b>	processing	<b>SRAM</b>	SRAM
<b>LPRIO11</b>	low priority, 11 bits	<b>PU</b>	pulse	<b>SRC</b>	source
<b>M</b>	M-bit	<b>PULSE</b>	pulse	<b>SRES</b>	soft reset
<b>M</b>	divider value M	<b>PWM</b>	pulse width modulation	<b>SSYNC</b>	single synchronization pulse
<b>MAN</b>	manual	<b>RAM</b>	RAM	<b>ST</b>	S/T interface
<b>MAX</b>	maximum	<b>RD</b>	read	<b>STA</b>	state, status
<b>MD</b>	mode	<b>RDY</b>	ready	<b>START</b>	start
<b>MERGE</b>	merge	<b>REG</b>	register	<b>STATUS</b>	status
				<b>STIO</b>	STIO pins
				<b>STOP</b>	stop
				<b>STR</b>	strict
				<b>STUP</b>	S/T / U <sub>p</sub> interface)

<b>SU</b>	Universal ISDN Port (combined S/T and U <sub>p</sub> interface)	<b>TI</b>	timer	<b>USE</b>	use, usage
<b>SUBCH</b>	subchannel	<b>TRI</b>	tristate	<b>VAL</b>	value
<b>SWAP</b>	swap	<b>TRP</b>	transparent	<b>VIO</b>	code violation
<b>SYNC</b>	synchronize, synchronization	<b>TS</b>	time step	<b>WAIT</b>	wait
<b>SYNCI</b>	synchronization input signal	<b>TST</b>	test	<b>WAK</b>	wakeup
<b>T</b>	T-bits	<b>TX</b>	transmit	<b>WD</b>	watchdog timer
<b>T2</b>	S/T timer T2	<b>TXHS</b>	transmitter handshake	<b>WR</b>	write
<b>THRES</b>	threshold	<b>TXR</b>	transmitter ready	<b>Z1</b>	Z1-counter
		<b>UNIDIR</b>	unidirectional	<b>Z2</b>	Z2-counter
		<b>UP</b>	U <sub>p</sub> interface		
		<b>USAGE</b>	usage		

# Index of register and bitmap names

Index entries are sorted by name. Pages of the register tables are printed in bold type.

<b>A_B1_RX</b>	<b>A_D_RX</b>	Ch. 3: 78
Ch. 5: 147, 149, 152, 167, 168, 176, <b>205</b>	Ch. 5: 147, 149, 152, 167, 168, 176, 194, 203, <b>206</b> , 207	Ch. 4: 118, 121, 122, <b>126</b>
<b>A_B1_TX</b>	<b>A_D_TX</b>	Ch. 9: 304
Ch. 5: 147, 149, 167, 176, <b>196</b>	Ch. 5: 147, 149, 167, 176, 188, 190, <b>197</b>	<b>A_MS_DF</b>
<b>A_B2_RX</b>	<b>A_E_RX</b>	Ch. 5: 188, <b>194</b> , 206, 207
Ch. 5: 147, 149, 152, 167, 168, 176, <b>205</b>	Ch. 5: 147, 149, 151, 152, 167, 168, 176, 194, 203, 206, <b>207</b>	<b>A_MS_RX</b>
<b>A_B2_TX</b>	<b>A_F1</b>	Ch. 5: 147, 151, 157, 166, 167, 173, <b>203</b> , 206, 207
Ch. 5: 147, 149, 167, 176, <b>196</b>	Ch. 3: 78	<b>A_MS_TX</b>
<b>A_BAC_S_TX</b>	Ch. 4: <b>128</b>	Ch. 5: 147, 151, 152, 157, 166–168, 173, <b>190</b> , 194, 197, 198, 203
Ch. 5: 147, 149, 167, 176, 188, 190, <b>198</b>	<b>A_F2</b>	<b>A_SL_CFG</b>
<b>A_CHANNEL</b>	Ch. 3: 78	Ch. 2: 40
Ch. 2: 40	Ch. 4: <b>129</b>	Ch. 3: 74–76, 81–84, 88, 89, 97, 108, 109, 114, 115
Ch. 3: 75, 76, 78, 85, 87–89, 91, 93, 95, 96, 98, 99, 111, 112, 114, 115	<b>A_FIFO_CTRL</b>	Ch. 6: 211, 213, 235, 237, 258, <b>262</b>
Ch. 4: <b>140</b>	Ch. 2: 40	<b>A_ST_CTRL3</b>
<b>A_CH_MSK</b>	Ch. 3: 78, 93	Ch. 5: 146, 186, <b>191</b> , 192
Ch. 2: 40	Ch. 4: 130, 137, <b>142</b>	<b>A_SUBCH_CFG</b>
Ch. 3: 78, 79, 86, 93, 94, 100, 101, 105–108, 110, 111, 113, 114	Ch. 8: 268, 271	Ch. 2: 40
Ch. 4: 126, <b>136</b>	Ch. 9: 305–308	Ch. 3: 78, 79, 85, 86, 93, 94, 100, 101, 105, 106, 108, 109, 111–115
<b>A_CON_HDLC</b>	<b>A_FIFO_DATA</b>	Ch. 4: 123, 136, <b>139</b>
Ch. 2: 40	Ch. 3: 78	<b>A_SU_CLK_DLY</b>
Ch. 3: 69, 71, 78, 79, 81–83, 86–89, 93, 94, 96–98, 106, 108, 109, 111, 112, 114, 115	Ch. 4: 120, <b>135</b>	Ch. 5: 148, 153, 157, 169, 173, <b>195</b>
Ch. 4: 118, 120, 122, <b>137</b> , 142	<b>A_FIFO_DATA_NOINC</b>	<b>A_SU_CTRL0</b>
Ch. 9: 287	Ch. 3: 78	Ch. 5: 149, 157, 166, 173, <b>185</b> , 189, 204
	Ch. 4: 120, 123, <b>135</b>	Ch. 9: 289
	<b>A_FIFO_SEQ</b>	<b>A_SU_CTRL1</b>
	Ch. 2: 40	Ch. 5: 153, 154, 157, 170, 172, 173, <b>187</b>
	Ch. 3: 78, 93, 95, 96, 98, 99	
	Ch. 4: <b>141</b>	
	<b>A_FIFO_STA</b>	
	Ch. 4: 126, <b>130</b> , 143	
	<b>A_INC_RES_FIFO</b>	

<b>A_SU_CTRL2</b>	<b>R_CI_TX</b>	<b>R_GCI_CFG1</b>
Ch. 5: 149, 157, 159, 166, 173, 185, <b>188</b>	Ch. 6: 233, <b>255</b>	Ch. 6: 235, 237, 257, <b>258</b>
<b>A_SU_DLYH</b>	<b>R_CLK_CFG</b>	<b>R_GCI_STA</b>
Ch. 5: 157, 173, 201, <b>202</b>	Ch. 9: 276, <b>295</b>	Ch. 6: 233, 241, <b>261</b>
<b>A_SU_DLYL</b>	<b>R_CTRL</b>	<b>R_GPIO_EN0</b>
Ch. 5: 157, 173, <b>201</b> , 202	Ch. 2: <b>62</b>	Ch. 10: <b>318</b>
<b>A_SU_RD_STA</b>	Ch. 9: 276	<b>R_GPIO_EN1</b>
Ch. 5: 153, 157, 169, 172, 173, 184, <b>200</b>	<b>R_F0_CNTH</b>	Ch. 10: 312, <b>316</b>
<b>A_SU_STA</b>	Ch. 6: <b>259</b>	<b>R_GPIO_EN2</b>
Ch. 5: 199, <b>204</b>	<b>R_F0_CNTL</b>	Ch. 10: <b>317</b>
Ch. 9: 289, 302	Ch. 6: <b>259</b>	<b>R_GPIO_IN0</b>
<b>A_SU_WR_STA</b>	<b>R_FIFO</b>	Ch. 10: 312, <b>321</b>
Ch. 5: 153, 154, 157, 169, 170, 172, 173, <b>184</b> , 187, 200	Ch. 3: 71, 76–78, 81–83, 85, 87–89, 91, 93, 94, 96–98, 101, 105–115	<b>R_GPIO_IN1</b>
<b>A_UP_CTRL3</b>	Ch. 4: 118, 121–123, 126, <b>127</b> , 128–131, 135–142	Ch. 10: 312, <b>320</b>
Ch. 5: 146, 172, 191, <b>192</b>	<b>R_FIFO_BL0_IRQ</b>	<b>R_GPIO_IN2</b>
<b>A_USAGE</b>	Ch. 4: 142	Ch. 10: <b>320</b>
Ch. 4: <b>131</b>	Ch. 9: 287, 300, 304, <b>305</b>	<b>R_GPIO_OUT0</b>
<b>A_Z1</b>	<b>R_FIFO_BL1_IRQ</b>	Ch. 10: <b>318</b>
Ch. 3: 78	Ch. 9: 300, <b>306</b>	<b>R_GPIO_OUT1</b>
Ch. 4: <b>128</b>	<b>R_FIFO_BL2_IRQ</b>	Ch. 10: <b>315</b>
<b>A_Z2</b>	Ch. 9: 300, <b>307</b>	<b>R_GPIO_OUT2</b>
Ch. 3: 78	<b>R_FIFO_BL3_IRQ</b>	Ch. 10: <b>317</b>
Ch. 4: <b>128</b>	Ch. 4: 142	<b>R_GPIO_SEL</b>
<b>R_AF0_OVIEW</b>	Ch. 9: 287, 300, 304, <b>308</b>	Ch. 10: 312, <b>319</b>
Ch. 5: <b>199</b> , 205	<b>R_FIFO_MD</b>	<b>R_GPIO_SEL_BL</b>
Ch. 9: 289, 302	Ch. 3: 77, 85, 91	Ch. 10: <b>316</b>
<b>R_BERT_ECH</b>	Ch. 4: 119, 124, <b>125</b> , 127	<b>R_INT_DATA</b>
Ch. 8: 268, 271, 273, <b>274</b>	<b>R_FIFO_THRES</b>	Ch. 2: 40, <b>65</b>
<b>R_BERT_ECL</b>	Ch. 4: <b>124</b> , 131–134	<b>R_IRQ_CTRL</b>
Ch. 8: 268, 271, <b>273</b> , 274	<b>R_FILL_BL0</b>	Ch. 9: 286, 287, <b>297</b>
<b>R_BERT_STA</b>	Ch. 4: <b>131</b>	<b>R_IRQ_OVIEW</b>
Ch. 6: 218, 254	<b>R_FILL_BL1</b>	Ch. 9: 287, 289, <b>300</b>
Ch. 8: 268, 271, <b>273</b>	Ch. 4: <b>132</b>	<b>R_MISC_IRQ</b>
<b>R_BERT_WD_MD</b>	<b>R_FILL_BL2</b>	Ch. 6: 233
Ch. 8: 268, <b>272</b>	Ch. 4: <b>133</b>	Ch. 9: 289–291, 296, 301, <b>302</b> , 304
Ch. 9: 292	<b>R_FILL_BL3</b>	<b>R_MISC_IRQMSK</b>
<b>R_CHIP_ID</b>	Ch. 4: <b>134</b>	Ch. 5: 199, 205
Ch. 2: <b>64</b>	<b>R_FIRST_FIFO</b>	Ch. 6: 232, 233
<b>R_CHIP_RV</b>	Ch. 3: 91–95	Ch. 9: 289–291, <b>296</b> , 302
Ch. 2: <b>64</b>	Ch. 4: <b>124</b>	<b>R_MON_RX</b>
<b>R_CIRM</b>	<b>R_FSM_IDX</b>	Ch. 6: 232, 233, 241, <b>261</b>
Ch. 4: 118	Ch. 3: 78, 91, 93, 95, 96, 98, 99	Ch. 9: 291
Ch. 9: 276, 285, <b>293</b>	Ch. 4: <b>127</b>	<b>R_MON_TX</b>
<b>R_CI_RX</b>	<b>R_GCI_CFG0</b>	Ch. 6: 233, 238, 239, 241, <b>258</b> , 261
Ch. 6: 232, <b>260</b>	Ch. 6: 232–235, 237, <b>256</b>	Ch. 9: 291
		<b>R_MSS0</b>
		Ch. 6: 224, 226–228, 243,

<b>246</b>	<b>R_SH1L</b>	Ch. 5: 151, 152, 167, 168, <b>187</b>
<b>R_MSS1</b>	Ch. 6: 230, 243–245, <b>252</b>	<b>V_B1_RX</b>
Ch. 5: 188	<b>R_SLOT</b>	Ch. 5: <b>205</b>
Ch. 6: 226, 243, <b>251</b>	Ch. 3: 74–76, 81–84, 88, 89, 97, 108, 109, 114, 115	<b>V_B1_RX_EN</b>
<b>R_PCM_MD0</b>	Ch. 6: 235, 237, <b>242</b> , 258, 262	Ch. 5: 149, 152, 166, 168, <b>188</b>
Ch. 6: 210, 213, 228–230, 235, 237, <b>243</b> , 244–246, 248, 249, 251–253	<b>R_SL_MAX</b>	<b>V_B1_TX</b>
<b>R_PCM_MD1</b>	Ch. 6: <b>259</b>	Ch. 5: <b>196</b>
Ch. 6: 210, 212, 218, 220, 242, 243, <b>248</b> , 249, 259	<b>R_SL_SEL0</b>	<b>V_B1_TX_EN</b>
<b>R_PCM_MD2</b>	Ch. 6: 229, 243, <b>244</b>	Ch. 5: 149, 151, 166, 167, <b>185</b>
Ch. 6: 213, 218, 220, 243, <b>249</b>	<b>R_SL_SEL1</b>	<b>V_B2_RX</b>
<b>R_PLL_CTRL</b>	Ch. 6: 229, 230, 243, 244, <b>245</b>	Ch. 5: <b>205</b>
Ch. 9: 281, <b>299</b>	<b>R_SL_SEL7</b>	<b>V_B2_RX_EN</b>
<b>R_PLL_N</b>	Ch. 2: 62	Ch. 5: 149, 152, 166, 168, <b>188</b>
Ch. 9: <b>309</b>	Ch. 5: 156, 172	<b>V_B2_TX</b>
<b>R_PLL_P</b>	Ch. 6: 243, <b>245</b>	Ch. 5: <b>196</b>
Ch. 9: <b>309</b>	<b>R_STATUS</b>	<b>V_B2_TX_EN</b>
<b>R_PLL_S</b>	Ch. 4: 119	Ch. 5: 149, 151, 166, 167, <b>185</b>
Ch. 9: <b>309</b>	Ch. 5: 176, 196–198	<b>V_BAC_D</b>
<b>R_PLL_STA</b>	Ch. 6: 213	Ch. 5: 151, <b>187</b>
Ch. 9: <b>308</b>	Ch. 9: 285, 302, <b>304</b>	<b>V_BAC_NINV</b>
<b>R_PWM0</b>	<b>R_SU_IRQ</b>	Ch. 5: 151, 167, <b>194</b>
Ch. 7: 264, <b>265</b>	Ch. 5: 157, 173, 176	<b>V_BAC_S_SEL</b>
<b>R_PWM1</b>	Ch. 9: 286, 296, 301, <b>303</b>	Ch. 5: 151, 167, <b>188</b> , 194
Ch. 7: 264, 265, <b>266</b>	<b>R_SU_IRQMSK</b>	<b>V_BAC_TX</b>
<b>R_PWM_CFG</b>	Ch. 5: 176	Ch. 5: <b>198</b>
Ch. 7: 264, <b>265</b>	Ch. 9: 286, <b>296</b> , 303	<b>V_BERT_ECH</b>
<b>R_PWM_MD</b>	<b>R_SU_SEL</b>	Ch. 8: 270, <b>274</b>
Ch. 7: 264, <b>266</b>	Ch. 5: 146, <b>183</b> , 184, 185, 187, 188, 190–192, 194–198, 200–207	<b>V_BERT_ECL</b>
Ch. 9: 290, 304	<b>R_SU_SYNC</b>	Ch. 8: 270, <b>273</b>
<b>R_RAM_ADDR</b>	Ch. 6: 218, 249, <b>254</b>	<b>V_BERT_EN</b>
Ch. 2: <b>63</b> , 66	<b>R_TI_WD</b>	Ch. 4: <b>142</b>
<b>R_RAM_CTRL</b>	Ch. 9: 290, 292, <b>298</b>	Ch. 8: 268–271
Ch. 2: <b>63</b> , 66	<b>V_ABO_DONE</b>	<b>V_BERT_ERR</b>
<b>R_RAM_DATA</b>	Ch. 4: 126, <b>130</b>	Ch. 8: 268, 269, <b>272</b>
Ch. 2: 40, 63, <b>66</b>	<b>V_ADDR_INC</b>	<b>V_BERT_INV_DATA</b>
<b>R_RAM_USE</b>	Ch. 2: <b>63</b>	Ch. 8: 270, 271, <b>273</b>
Ch. 2: <b>64</b>	<b>V_ADDR_RES</b>	<b>V_BERT_SYNC</b>
<b>R_SH0H</b>	Ch. 2: <b>63</b>	Ch. 8: 268, 270, <b>273</b>
Ch. 6: 229, 230, 243–245, <b>252</b>	<b>V_AUTO_SYNCI</b>	<b>V_BIT_CNT</b>
<b>R_SH0L</b>	Ch. 6: 218, 219, <b>254</b>	Ch. 3: 100–103, 106, 108, 109, 111, 112, 114, 115
Ch. 6: 229, 230, 243–245, <b>252</b>	<b>V_AUTO_WD_RES</b>	Ch. 4: 136, <b>139</b>
<b>R_SH1H</b>	Ch. 8: <b>272</b>	Ch. 8: 268–271
Ch. 6: 230, 243–245, <b>253</b>	Ch. 9: 292	<b>V_BUSY</b>
	<b>V_B12_SWAP</b>	

Ch. 4: 119	Ch. 9: 276, 277, <b>293</b>	228, <b>243</b>
Ch. 9: 285, <b>304</b>	<b>V_CLK_PLL</b>	<b>V_F0_NEG</b>
<b>V_C2I_EN</b>	Ch. 9: 276, 277, <b>295</b>	Ch. 6: 219, <b>243</b>
Ch. 6: 213, 220, <b>249</b>	<b>V_DATA_FLOW</b>	<b>V_F1</b>
<b>V_C2O_EN</b>	Ch. 3: 69, 71–73, 81–83,	Ch. 4: <b>128</b>
Ch. 6: 220, <b>249</b>	87–89, 96–98, 106, 108,	<b>V_F2</b>
<b>V_C4_POL</b>	109, 111, 112, 114, 115	Ch. 4: <b>129</b>
Ch. 6: 219, <b>243</b>	Ch. 4: <b>138</b>	<b>V_FIFO0_RX_IRQ</b>
<b>V_CHIP_ID</b>	<b>V_DF_MD</b>	Ch. 9: 288, <b>305</b>
Ch. 2: <b>64</b>	Ch. 3: 77, 85, 91	<b>V_FIFO0_TX_IRQ</b>
<b>V_CHIP_RV</b>	Ch. 4: <b>125</b> , 127	Ch. 9: 288, <b>305</b>
Ch. 2: <b>64</b>	<b>V_D_MERGE_RX</b>	<b>V_FIFO10_RX_IRQ</b>
<b>V_CH_FDIR</b>	Ch. 6: <b>254</b>	Ch. 9: <b>307</b>
Ch. 3: 76, 85, 87–89, 95, 96,	<b>V_D_MERGE_TX</b>	<b>V_FIFO10_TX_IRQ</b>
98, 99, 111, 112, 114,	Ch. 6: <b>254</b>	Ch. 9: <b>307</b>
115	<b>V_D_RES</b>	<b>V_FIFO11_RX_IRQ</b>
Ch. 4: <b>140</b>	Ch. 5: 151, 167, <b>187</b>	Ch. 9: <b>307</b>
<b>V_CH_FNUM</b>	<b>V_D_RX</b>	<b>V_FIFO11_TX_IRQ</b>
Ch. 3: 76, 85, 87–89, 95, 96,	Ch. 5: <b>206</b>	Ch. 9: <b>307</b>
98, 99, 111, 112, 114,	<b>V_D_RX_AB</b>	<b>V_FIFO12_RX_IRQ</b>
115	Ch. 5: <b>206</b>	Ch. 9: <b>308</b>
Ch. 4: <b>140</b>	<b>V_D_RX_S</b>	<b>V_FIFO12_TX_IRQ</b>
<b>V_CH_MSK</b>	Ch. 5: <b>206</b>	Ch. 9: <b>308</b>
Ch. 3: 104–106, 108, 111,	<b>V_D_RX_SG</b>	<b>V_FIFO13_RX_IRQ</b>
114	Ch. 5: <b>206</b>	Ch. 9: <b>308</b>
Ch. 4: <b>136</b>	<b>V_D_TX</b>	<b>V_FIFO13_TX_IRQ</b>
<b>V_CH_SDIR</b>	Ch. 5: <b>197</b>	Ch. 9: <b>308</b>
Ch. 3: 74, 76, 81–84, 88, 89,	<b>V_D_TX_BAC</b>	<b>V_FIFO14_RX_IRQ</b>
97, 108, 109, 114, 115	Ch. 5: <b>197</b>	Ch. 9: 288, <b>308</b>
Ch. 6: 235, 237, <b>262</b>	<b>V_D_TX_S</b>	<b>V_FIFO14_TX_IRQ</b>
<b>V_CH_SNUM</b>	Ch. 5: <b>197</b> , 198	Ch. 9: <b>308</b>
Ch. 3: 74, 76, 81–84, 88, 89,	<b>V_EV_TS</b>	<b>V_FIFO15_RX_IRQ</b>
97, 108, 109, 114, 115	Ch. 9: 290, <b>298</b>	Ch. 9: 288, <b>308</b>
Ch. 6: 235, 237, <b>262</b>	<b>V_E_MERGE_RX</b>	<b>V_FIFO15_TX_IRQ</b>
<b>V_CI_IRQ</b>	Ch. 6: <b>254</b>	Ch. 9: 288, <b>308</b>
Ch. 6: 233	<b>V_E_RX</b>	<b>V_FIFO1_RX_IRQ</b>
Ch. 9: 289, <b>302</b>	Ch. 5: <b>207</b>	Ch. 9: 288, <b>305</b>
<b>V_CI_IRQMSK</b>	<b>V_E_RX_AB</b>	<b>V_FIFO1_TX_IRQ</b>
Ch. 6: 232, 233	Ch. 5: <b>207</b>	Ch. 9: 288, <b>305</b>
Ch. 9: 289, <b>296</b>	<b>V_E_RX_S</b>	<b>V_FIFO2_RX_IRQ</b>
<b>V_CLKO_HI</b>	Ch. 5: <b>207</b>	Ch. 9: <b>305</b>
Ch. 9: 276, 277, <b>295</b>	<b>V_E_RX_SG</b>	<b>V_FIFO2_TX_IRQ</b>
<b>V_CLKO_OFF</b>	Ch. 5: <b>207</b>	Ch. 9: 288, <b>305</b>
Ch. 9: 276, 277, <b>295</b>	<b>V_F0_CNTH</b>	<b>V_FIFO3_RX_IRQ</b>
<b>V_CLKO_PLL</b>	Ch. 6: <b>259</b>	Ch. 9: <b>305</b>
Ch. 9: 276, 277, <b>295</b>	<b>V_F0_CNTL</b>	<b>V_FIFO3_TX_IRQ</b>
<b>V_CLK_F1</b>	Ch. 6: <b>259</b>	Ch. 9: <b>305</b>
Ch. 9: 276, 277, <b>295</b>	<b>V_F0_LEN</b>	<b>V_FIFO4_RX_IRQ</b>
<b>V_CLK_OFF</b>	Ch. 6: 213, 219, 225, 227,	Ch. 9: <b>306</b>

<b>V_FIFO4_TX_IRQ</b>	Ch. 9: 305–308	Ch. 4: <b>131</b>
Ch. 9: <b>306</b>	<b>V_FIFO_IRQSTA</b>	<b>V_FILL_FIFO3_RX</b>
<b>V_FIFO5_RX_IRQ</b>	Ch. 9: <b>304</b>	Ch. 4: <b>131</b>
Ch. 9: <b>306</b>	<b>V_FIFO_IRQ_EN</b>	<b>V_FILL_FIFO3_TX</b>
<b>V_FIFO5_TX_IRQ</b>	Ch. 9: 287, 288, <b>297</b>	Ch. 4: <b>131</b>
Ch. 9: <b>306</b>	<b>V_FIFO_LPRIO</b>	<b>V_FILL_FIFO4_RX</b>
<b>V_FIFO6_RX_IRQ</b>	Ch. 2: <b>62</b>	Ch. 4: <b>132</b>
Ch. 9: <b>306</b>	<b>V_FIFO_MD</b>	<b>V_FILL_FIFO4_TX</b>
<b>V_FIFO6_TX_IRQ</b>	Ch. 4: 119, <b>125</b>	Ch. 4: <b>132</b>
Ch. 9: <b>306</b>	<b>V_FIFO_NUM</b>	<b>V_FILL_FIFO5_RX</b>
<b>V_FIFO7_RX_IRQ</b>	Ch. 3: 76, 81–83, 87–89,	Ch. 4: <b>132</b>
Ch. 9: <b>306</b>	96–98, 106, 108, 109,	<b>V_FILL_FIFO5_TX</b>
<b>V_FIFO7_TX_IRQ</b>	111, 112, 114, 115	Ch. 4: <b>132</b>
Ch. 9: <b>306</b>	Ch. 4: <b>127</b>	<b>V_FILL_FIFO6_RX</b>
<b>V_FIFO8_RX_IRQ</b>	<b>V_FILL_FIFO0_RX</b>	Ch. 4: <b>132</b>
Ch. 9: <b>307</b>	Ch. 4: <b>131</b>	<b>V_FILL_FIFO6_TX</b>
<b>V_FIFO8_TX_IRQ</b>	<b>V_FILL_FIFO0_TX</b>	Ch. 4: <b>132</b>
Ch. 9: <b>307</b>	Ch. 4: <b>131</b>	<b>V_FILL_FIFO7_RX</b>
<b>V_FIFO9_RX_IRQ</b>	<b>V_FILL_FIFO10_RX</b>	Ch. 4: <b>132</b>
Ch. 9: <b>307</b>	Ch. 4: <b>133</b>	<b>V_FILL_FIFO7_TX</b>
<b>V_FIFO9_TX_IRQ</b>	<b>V_FILL_FIFO10_TX</b>	Ch. 4: <b>132</b>
Ch. 9: <b>307</b>	Ch. 4: <b>133</b>	<b>V_FILL_FIFO8_RX</b>
<b>V_FIFO_BL0_IRQ</b>	<b>V_FILL_FIFO11_RX</b>	Ch. 4: <b>133</b>
Ch. 9: 287, 288, <b>300</b>	Ch. 4: <b>133</b>	<b>V_FILL_FIFO8_TX</b>
<b>V_FIFO_BL1_IRQ</b>	<b>V_FILL_FIFO11_TX</b>	Ch. 4: <b>133</b>
Ch. 9: 288, <b>300</b>	Ch. 4: <b>133</b>	<b>V_FILL_FIFO9_RX</b>
<b>V_FIFO_BL2_IRQ</b>	<b>V_FILL_FIFO12_RX</b>	Ch. 4: <b>133</b>
Ch. 9: 288, <b>300</b>	Ch. 4: <b>134</b>	<b>V_FILL_FIFO9_TX</b>
<b>V_FIFO_BL3_IRQ</b>	<b>V_FILL_FIFO12_TX</b>	Ch. 4: <b>133</b>
Ch. 9: 287, 288, <b>300</b>	Ch. 4: <b>134</b>	<b>V_FIRST_FIFO_DIR</b>
<b>V_FIFO_DATA</b>	<b>V_FILL_FIFO13_RX</b>	Ch. 3: 93, 95
Ch. 4: <b>135</b>	Ch. 4: <b>134</b>	Ch. 4: <b>124</b>
<b>V_FIFO_DATA_NOINC</b>	<b>V_FILL_FIFO13_TX</b>	<b>V_FIRST_FIFO_NUM</b>
Ch. 4: <b>135</b>	Ch. 4: <b>134</b>	Ch. 3: 93, 95
<b>V_FIFO_DIR</b>	<b>V_FILL_FIFO14_RX</b>	Ch. 4: <b>124</b>
Ch. 3: 71, 73, 76, 81–83, 85,	Ch. 4: <b>134</b>	<b>V_FR_ABO</b>
87–89, 96–98, 106, 108,	<b>V_FILL_FIFO14_TX</b>	Ch. 4: 130, <b>143</b>
109, 111, 112, 114, 115	Ch. 4: <b>134</b>	<b>V_G2_G3</b>
Ch. 4: <b>127</b> , 138	<b>V_FILL_FIFO15_RX</b>	Ch. 5: 153, 172, <b>200</b>
<b>V_FIFO_ERR</b>	Ch. 4: <b>134</b>	<b>V_G2_G3_EN</b>
Ch. 4: 126, <b>130</b> , 143	<b>V_FILL_FIFO15_TX</b>	Ch. 5: 153, 154, 170, 172,
<b>V_FIFO_IRQ</b>	Ch. 4: <b>134</b>	<b>187</b>
Ch. 3: 79, 81–83, 86–89,	<b>V_FILL_FIFO1_RX</b>	<b>V_GCI_ABO</b>
94–98, 106, 108, 109,	Ch. 4: <b>131</b>	Ch. 6: 241, <b>261</b>
111, 112, 114, 115	<b>V_FILL_FIFO1_TX</b>	<b>V_GCI_C</b>
Ch. 4: 118, <b>137</b> , 138, 142	Ch. 4: <b>131</b>	Ch. 6: 233, <b>255</b>
Ch. 9: 287, 305–308	<b>V_FILL_FIFO2_RX</b>	<b>V_GCI_EN</b>
<b>V_FIFO_IRQMSK</b>	Ch. 4: <b>131</b>	Ch. 6: 232, 235, 237, <b>257</b>
Ch. 4: <b>142</b>	<b>V_FILL_FIFO2_TX</b>	<b>V_GCI_I</b>

Ch. 6: 233, <b>260</b>	Ch. 10: <b>318</b>	V_GPIO_IN8
V_GCI_MR	V_GPIO_EN4	Ch. 10: 313, <b>320</b>
Ch. 6: <b>261</b>	Ch. 10: <b>318</b>	V_GPIO_IN9
V_GCI_MX	V_GPIO_EN5	Ch. 10: 313, <b>320</b>
Ch. 6: <b>261</b>	Ch. 10: <b>318</b>	V_GPIO_OUT0
V_GCI_RX	V_GPIO_EN6	Ch. 10: <b>318</b>
Ch. 6: <b>261</b>	Ch. 10: <b>318</b>	V_GPIO_OUT1
V_GCI_SL	V_GPIO_EN7	Ch. 10: <b>318</b>
Ch. 6: 231, 232, 235, 237,	Ch. 10: <b>318</b>	V_GPIO_OUT10
<b>258</b>	V_GPIO_EN8	Ch. 10: <b>315</b>
V_GCI_SWAP_RXHS	Ch. 10: 312, 313, <b>316</b>	V_GPIO_OUT11
Ch. 6: 234, 237, <b>256</b> , 257	V_GPIO_EN9	Ch. 10: <b>315</b>
V_GCI_SWAP_STIO	Ch. 10: 313, <b>316</b>	V_GPIO_OUT12
Ch. 6: 234, 237, 256, <b>257</b>	V_GPIO_IN0	Ch. 10: <b>315</b>
V_GCI_SWAP_TXHS	Ch. 10: <b>321</b>	V_GPIO_OUT13
Ch. 6: 234, 237, <b>256</b> , 257	V_GPIO_IN1	Ch. 10: <b>315</b>
V_GLOB_IRQ_EN	Ch. 10: <b>321</b>	V_GPIO_OUT14
Ch. 9: 286, <b>297</b>	V_GPIO_IN10	Ch. 10: <b>315</b>
V_GPIO_BL0	Ch. 10: <b>320</b>	V_GPIO_OUT15
Ch. 10: 313, <b>316</b>	V_GPIO_IN11	Ch. 10: <b>315</b>
V_GPIO_BL1	Ch. 10: <b>320</b>	V_GPIO_OUT16
Ch. 10: <b>316</b>	V_GPIO_IN12	Ch. 10: 313, <b>317</b>
V_GPIO_EN0	Ch. 10: <b>320</b>	V_GPIO_OUT17
Ch. 10: <b>318</b>	V_GPIO_IN13	Ch. 10: <b>317</b>
V_GPIO_EN1	Ch. 10: <b>320</b>	V_GPIO_OUT18
Ch. 10: <b>318</b>	V_GPIO_IN14	Ch. 10: <b>317</b>
V_GPIO_EN10	Ch. 10: <b>320</b>	V_GPIO_OUT19
Ch. 10: <b>316</b>	V_GPIO_IN15	Ch. 10: <b>317</b>
V_GPIO_EN11	Ch. 10: <b>320</b>	V_GPIO_OUT2
Ch. 10: <b>316</b>	V_GPIO_IN16	Ch. 10: <b>318</b>
V_GPIO_EN12	Ch. 10: 313, <b>320</b>	V_GPIO_OUT3
Ch. 10: <b>316</b>	V_GPIO_IN17	Ch. 10: <b>318</b>
V_GPIO_EN13	Ch. 10: <b>320</b>	V_GPIO_OUT4
Ch. 10: <b>316</b>	V_GPIO_IN18	Ch. 10: <b>318</b>
V_GPIO_EN14	Ch. 10: <b>320</b>	V_GPIO_OUT5
Ch. 10: <b>316</b>	V_GPIO_IN19	Ch. 10: <b>318</b>
V_GPIO_EN15	Ch. 10: <b>320</b>	V_GPIO_OUT6
Ch. 10: <b>316</b>	V_GPIO_IN2	Ch. 10: <b>318</b>
V_GPIO_EN16	Ch. 10: <b>321</b>	V_GPIO_OUT7
Ch. 10: 313, <b>317</b>	V_GPIO_IN3	Ch. 10: <b>318</b>
V_GPIO_EN17	Ch. 10: <b>321</b>	V_GPIO_OUT8
Ch. 10: <b>317</b>	V_GPIO_IN4	Ch. 10: 313, <b>315</b>
V_GPIO_EN18	Ch. 10: <b>321</b>	V_GPIO_OUT9
Ch. 10: <b>317</b>	V_GPIO_IN5	Ch. 10: 313, <b>315</b>
V_GPIO_EN19	Ch. 10: <b>321</b>	V_GPIO_SEL0
Ch. 10: <b>317</b>	V_GPIO_IN6	Ch. 10: 312, <b>319</b>
V_GPIO_EN2	Ch. 10: <b>321</b>	V_GPIO_SEL1
Ch. 10: <b>318</b>	V_GPIO_IN7	Ch. 10: <b>319</b>
V_GPIO_EN3	Ch. 10: <b>321</b>	V_GPIO_SEL2

Ch. 10: <b>319</b>	<b>V_MISC_IRQSTA</b>	Ch. 6: 225, 226, <b>246</b>
<b>V_GPIO_SEL3</b>	Ch. 9: 289, <b>304</b>	<b>V_MS_RX</b>
Ch. 10: <b>319</b>	<b>V_MIX_IRQ</b>	Ch. 5: 166, <b>203</b>
<b>V_GPIO_SEL4</b>	Ch. 4: 137, <b>142</b>	<b>V_MS_RX_RDY</b>
Ch. 10: <b>319</b>	Ch. 9: 287, 305–308	Ch. 5: 166, <b>203</b>
<b>V_GPIO_SEL5</b>	<b>V_MON_CI6</b>	<b>V_MS_SSYNC1</b>
Ch. 10: <b>319</b>	Ch. 6: 233, 255, <b>256</b> , 260	Ch. 5: 188
<b>V_GPIO_SEL6</b>	<b>V_MON_DLL</b>	Ch. 6: 225, <b>251</b>
Ch. 10: <b>319</b>	Ch. 6: 233, <b>256</b>	<b>V_MS_SSYNC2</b>
<b>V_GPIO_SEL7</b>	<b>V_MON_END</b>	Ch. 5: 151, 167, <b>188</b>
Ch. 10: <b>319</b>	Ch. 6: 233, 239, <b>256</b>	Ch. 6: 251
<b>V_HDLC_TRP</b>	<b>V_MON_RX</b>	<b>V_MS_TX</b>
Ch. 3: 79, 81–83, 87–89, 95–98, 106, 108, 109, 111, 112, 114, 115	Ch. 6: <b>261</b>	Ch. 5: 166, <b>190</b> , 194, 197, 198
Ch. 4: 122, <b>137</b>	<b>V_MON_RXR</b>	<b>V_MS_TX_RDY</b>
Ch. 9: 287	Ch. 6: 233, 241, <b>261</b>	Ch. 5: 166, <b>203</b>
<b>V_HFC_RES</b>	<b>V_MON_RX_IRQ</b>	<b>V_MULT_SU</b>
Ch. 4: 118	Ch. 9: 289, 291, <b>302</b>	Ch. 5: <b>183</b>
Ch. 9: 285, <b>293</b>	<b>V_MON_RX_IRQMSK</b>	<b>V_M_S_SRC</b>
<b>V_IDX</b>	Ch. 6: 232	Ch. 5: 151, 167, <b>194</b>
Ch. 3: 93, 95, 96, 98, 99	Ch. 9: 289, 291, <b>296</b>	<b>V_NEXT_FIFO_DIR</b>
Ch. 4: <b>127</b>	<b>V_MON_SLOW</b>	Ch. 3: 95, 96, 98, 99
<b>V_IFF</b>	Ch. 6: <b>256</b>	Ch. 4: <b>141</b>
Ch. 3: 81–83, 87–89, 96–98, 106, 108, 109, 111, 112, 114, 115	<b>V_MON_TX</b>	<b>V_NEXT_FIFO_NUM</b>
Ch. 4: 120, <b>137</b>	Ch. 6: <b>258</b>	Ch. 3: 95, 96, 98, 99
<b>V_INC_F</b>	<b>V_MON_TXR</b>	Ch. 4: <b>141</b>
Ch. 4: 121, 122, <b>126</b>	Ch. 6: 233, 239, <b>261</b>	<b>V_NO_CRC</b>
<b>V_INT_DATA</b>	<b>V_MON_TX_IRQ</b>	Ch. 4: <b>143</b>
Ch. 2: <b>65</b>	Ch. 6: 233	<b>V_NO_REP</b>
<b>V_INV_DATA</b>	Ch. 9: 289, 291, <b>302</b>	Ch. 4: <b>143</b>
Ch. 3: 106, 108, 109, 111, 112, 114, 115	<b>V_MON_TX_IRQMSK</b>	<b>V_NT_SYNC</b>
Ch. 4: <b>139</b>	Ch. 6: 233	Ch. 2: <b>62</b>
<b>V_IRQ_POL</b>	Ch. 9: 289, 291, <b>296</b>	Ch. 6: 219
Ch. 9: 286, <b>297</b>	<b>V_MSS_DLY</b>	<b>V_OSC_OFF</b>
<b>V_LOOP_FIFO</b>	Ch. 6: 225–227, <b>251</b>	Ch. 2: <b>62</b>
Ch. 3: 106, 108, 109, 111, 112, 114, 115	<b>V_MSS_MOD</b>	Ch. 9: 277
Ch. 4: <b>139</b>	Ch. 6: 225, 227, 228, <b>246</b>	<b>V_PAT_SEQ</b>
<b>V_LOST_STA</b>	<b>V_MSS_MOD_REP</b>	Ch. 8: 268–270, <b>272</b>
Ch. 9: <b>304</b>	Ch. 6: 225, 227, <b>246</b>	<b>V_PCM_CLK</b>
<b>V_MAN_SYNC</b>	<b>V_MSS_OFFS</b>	Ch. 9: 276, 277, <b>295</b>
Ch. 5: 177	Ch. 6: 225, <b>251</b>	<b>V_PCM_DR</b>
Ch. 6: 218, 219, <b>254</b>	<b>V_MSS_OUT_EN</b>	Ch. 6: 210, 242, <b>248</b> , 259
<b>V_MISC_IRQ</b>	Ch. 6: 225, 228, <b>246</b>	<b>V_PCM_IDX</b>
Ch. 9: 289, <b>301</b>	<b>V_MSS_OUT_REP</b>	Ch. 6: 229, 230, <b>243</b> , 244–246, 248, 249, 251–253
	Ch. 6: 225, 226, <b>247</b>	<b>V_PCM_INIT</b>
	<b>V_MSS_SRC</b>	Ch. 6: 213
	Ch. 6: 225, 226, <b>247</b>	Ch. 9: 285, <b>304</b>
	<b>V_MSS_SRC_EN</b>	
	Ch. 6: 224–226, <b>246</b>	
	<b>V_MSS_SRC_GRD</b>	

<b>V_PCM_LOOP</b>	<b>V_PWM1_16KHZ</b>	<b>V_SLIP_IRQ</b>
Ch. 6: 212, <b>248</b>	Ch. 7: 264, <b>265</b>	Ch. 9: 289, <b>302</b>
<b>V_PCM_MD</b>	<b>V_PWM1_MD</b>	<b>V_SLIP_IRQMSK</b>
Ch. 4: 120	Ch. 7: 264, <b>266</b>	Ch. 5: 199, 205
Ch. 6: 210, 213, 219, 234, 235, 237, <b>243</b>	<b>V_PWM_FRQ</b>	Ch. 9: 289, <b>296</b>
<b>V_PCM_OD</b>	Ch. 7: 264, <b>265</b>	<b>V_SL_DIR</b>
Ch. 6: 211, <b>248</b>	<b>V_RAM_ADDR0</b>	Ch. 3: 76, 81–84, 88, 89, 97, 108, 109, 114, 115
<b>V_PCM_RES</b>	Ch. 2: <b>63</b>	Ch. 6: 235, 237, <b>242</b>
Ch. 9: 285, <b>294</b>	<b>V_RAM_ADDR1</b>	<b>V_SL_MAX</b>
<b>V_PCM_SMPL</b>	Ch. 2: <b>63</b>	Ch. 6: <b>259</b>
Ch. 6: 212, <b>248</b>	<b>V_RAM_DATA</b>	<b>V_SL_NUM</b>
<b>V_PLL_ADJ</b>	Ch. 2: <b>66</b>	Ch. 3: 76, 81–84, 88, 89, 97, 108, 109, 114, 115
Ch. 6: 218, 220, <b>248</b> , 249	<b>V_RD_SYNC_SRC</b>	Ch. 6: 235, 237, <b>242</b>
<b>V_PLL_FREEZE</b>	Ch. 6: 218, 254	<b>V_SL_SEL0</b>
Ch. 9: 281, <b>299</b>	Ch. 8: <b>273</b>	Ch. 6: 229, <b>244</b>
<b>V_PLL_ICR</b>	<b>V_RES_FIFO</b>	<b>V_SL_SEL1</b>
Ch. 6: 218, 220, <b>249</b> , 250	Ch. 4: 118, <b>126</b>	Ch. 6: 229, 230, 244, <b>245</b>
<b>V_PLL_LOCK</b>	<b>V_RES_FIFO_ERR</b>	<b>V_SL_SEL7</b>
Ch. 9: 281, <b>308</b>	Ch. 4: <b>126</b> , 130	Ch. 6: <b>245</b>
<b>V_PLL_M</b>	<b>V_RES_LOST</b>	<b>V_SQ_T_DST</b>
Ch. 9: 281, <b>299</b>	Ch. 4: <b>126</b>	Ch. 5: 152, 168, <b>194</b> , 206, 207
<b>V_PLL_MAN</b>	Ch. 9: 304	<b>V_SQ_T_SRC</b>
Ch. 6: 218, 220, 249, <b>250</b>	<b>V_REV</b>	Ch. 5: 151, 167, <b>194</b>
<b>V_PLL_N</b>	Ch. 3: 79, 81–83, 86–89, 95–98, 106, 108, 109, 111, 112, 114, 115	<b>V_SRAM_USE</b>
Ch. 9: 280, 281, <b>309</b>	Ch. 4: 123, <b>127</b>	Ch. 2: <b>64</b>
<b>V_PLL_NRES</b>	<b>V_ROUT</b>	<b>V_SRES</b>
Ch. 9: 281, <b>299</b>	Ch. 3: 81–84, 88, 89, 97, 108, 109, 114, 115	Ch. 4: 118
<b>V_PLL_P</b>	Ch. 6: 211, 212, 235, 237, <b>262</b>	Ch. 9: 285, <b>293</b>
Ch. 9: 281, <b>309</b>	<b>V_SEQ_END</b>	<b>V_START_BIT</b>
<b>V_PLL_S</b>	Ch. 3: 93, 95, 96, 98, 99	Ch. 3: 100–103, 106–109, 111, 112, 114, 115
Ch. 9: 281, <b>309</b>	Ch. 4: <b>141</b>	Ch. 4: <b>139</b>
<b>V_PLL_TST</b>	<b>V_SG_AB_INV</b>	<b>V_STUP_IRQ</b>
Ch. 9: <b>299</b>	Ch. 5: 152, 168, <b>194</b>	Ch. 9: 287, <b>301</b>
<b>V_PROC</b>	<b>V_SH0H</b>	<b>V_ST_D_ACT</b>
Ch. 5: 176, 196–198	Ch. 6: 230, <b>252</b>	Ch. 5: <b>204</b>
Ch. 9: 302, <b>304</b>	<b>V_SH0L</b>	<b>V_ST_D_CONT</b>
<b>V_PROC_IRQ</b>	Ch. 6: 230, <b>252</b>	Ch. 5: <b>204</b>
Ch. 9: 289, 290, <b>302</b> , 304	<b>V_SH1H</b>	<b>V_ST_D_HPRI09</b>
<b>V_PROC_IRQMSK</b>	Ch. 6: 230, <b>253</b>	Ch. 5: <b>204</b>
Ch. 9: 289, 290, <b>296</b>	<b>V_SH1L</b>	<b>V_ST_D_LPRIO</b>
<b>V_PWM0</b>	Ch. 6: 230, <b>252</b>	Ch. 5: <b>185</b> , 204
Ch. 7: <b>265</b>	<b>V_SH_SEL0</b>	<b>V_ST_D_LPRIO11</b>
<b>V_PWM0_16KHZ</b>	Ch. 6: 229, <b>244</b>	Ch. 5: <b>204</b>
Ch. 7: 264, <b>265</b>	<b>V_SH_SEL1</b>	Ch. 5: 151, <b>187</b>
<b>V_PWM0_MD</b>	Ch. 6: 229, 230, 244, <b>245</b>	
Ch. 7: 264, <b>266</b>		
<b>V_PWM1</b>		
Ch. 7: <b>266</b>		

<b>V_ST_E_LO</b>	Ch. 5: 151, <b>187</b>	<b>V_UNIDIR_MD</b>	Ch. 4: 119, <b>125</b>
<b>V_ST_PULSE</b>	Ch. 5: 186, <b>191</b>	<b>V_UNIDIR_RX</b>	Ch. 4: 119, <b>125</b>
<b>V_ST_PU_CTRL</b>	Ch. 5: 151, <b>186</b>	<b>V_UP_DC_OFF</b>	Ch. 5: 167, <b>192</b>
<b>V_ST_SEL</b>	Ch. 5: 146, <b>191</b>	<b>V_UP_DC_STR</b>	Ch. 5: 167, <b>192</b>
<b>V_ST_SMPL</b>	Ch. 5: <b>195</b>	<b>V_UP_RPT_PAT</b>	Ch. 5: 167, 173, <b>192</b>
<b>V_ST_SQ_EN</b>	Ch. 5: 149, 151, <b>185</b>	<b>V_UP_SCRM_MD</b>	Ch. 5: 167, 168, 172, 173, <b>192</b>
<b>V_SU0_AF0</b>	Ch. 5: <b>199</b>	<b>V_UP_SCRM_RX_OFF</b>	Ch. 5: 168, 172, 173, <b>193</b>
<b>V_SU0_IRQ</b>	Ch. 5: 176	<b>V_UP_SCRM_TX_OFF</b>	Ch. 5: 167, 172, 173, <b>193</b>
	Ch. 9: 287, 296, <b>303</b>	<b>V_UP_SEL</b>	Ch. 5: <b>192</b>
<b>V_SU0_IRQMSK</b>	Ch. 5: 176	<b>V_UP_S_RX</b>	Ch. 5: 166, <b>203</b>
	Ch. 9: 287, <b>296</b>	<b>V_UP_S_TX</b>	Ch. 5: 166, <b>190, 194</b>
<b>V_SU1_AF0</b>	Ch. 5: <b>199</b>	<b>V_UP_VIO</b>	Ch. 5: <b>192</b>
<b>V_SU1_IRQ</b>	Ch. 5: 176	<b>V_USAGE</b>	Ch. 4: <b>131</b>
	Ch. 9: 287, 296, <b>303</b>	<b>V_WAIT_PROC</b>	Ch. 9: <b>293</b>
<b>V_SU1_IRQMSK</b>	Ch. 5: 176	<b>V_WAIT_REG</b>	Ch. 9: <b>293</b>
	Ch. 9: 287, <b>296</b>	<b>V_WAK_EN</b>	Ch. 7: <b>266</b>
<b>V_SU_2KHZ</b>	Ch. 5: 151, 167, 185, <b>189</b>		Ch. 9: 290, 304
<b>V_SU_ACT</b>	Ch. 5: 153, 169, <b>184</b>	<b>V_WAK_IRQ</b>	Ch. 9: 289, 290, <b>302</b>
<b>V_SU_AF0</b>	Ch. 5: 178, 204, <b>205</b>	<b>V_WAK_IRQMSK</b>	Ch. 9: 289, 290, <b>296</b>
	Ch. 9: 289, 302	<b>V_WAK_STA</b>	Ch. 9: 290, <b>304</b>
<b>V_SU_CLK</b>	Ch. 2: <b>62</b>	<b>V_WD_RES</b>	Ch. 8: <b>272</b>
	Ch. 9: 276, 277		Ch. 9: 292
<b>V_SU_CLK_DLY</b>	Ch. 5: 148, <b>195</b>	<b>V_WD_TS</b>	Ch. 9: 292, <b>298</b>
<b>V_SU_DLYH</b>	Ch. 5: <b>202</b>	<b>V_Z1</b>	Ch. 4: <b>128</b>
<b>V_SU_DLYL</b>	Ch. 5: <b>201</b>	<b>V_Z2</b>	Ch. 4: <b>128</b>
<b>V_SU_EXCHG</b>	Ch. 5: 151, 159, 167, <b>189</b>		
<b>V_SU_FR_SYNC</b>	Ch. 5: <b>200</b>		
<b>V_SU_INFO0</b>	Ch. 5: 153, <b>200</b>		
		<b>V_SU_LD_STA</b>	Ch. 5: 153, 169, 172, <b>184</b>
		<b>V_SU_MD</b>	Ch. 5: 157, 173, <b>185</b>
			Ch. 9: 289
		<b>V_SU_RES</b>	Ch. 9: 285, <b>294</b>
		<b>V_SU_RX_VAL</b>	Ch. 5: 152, 168, <b>194, 206, 207</b>
		<b>V_SU_SEL</b>	Ch. 5: 146, <b>183</b>
		<b>V_SU_SET_G2_G3</b>	Ch. 5: 154, 170, <b>184, 187</b>
		<b>V_SU_SET_STA</b>	Ch. 5: <b>184</b>
		<b>V_SU_STA</b>	Ch. 5: <b>200</b>
		<b>V_SU_STOP</b>	Ch. 5: 152, 168, <b>186</b>
		<b>V_SU_SYNC_NT</b>	Ch. 5: 157, 173, <b>188</b>
		<b>V_SU_T2_EXP</b>	Ch. 5: <b>200</b>
		<b>V_SU_TRI</b>	Ch. 5: 151, 157, 167, 173, <b>189</b>
		<b>V_SU_TST_SIG</b>	Ch. 5: 151, 167, <b>185, 189</b>
		<b>V_SYNC_OUT1</b>	Ch. 6: <b>249</b>
		<b>V_SYNC_OUT2</b>	Ch. 6: <b>249, 250</b>
		<b>V_SYNC_SEL</b>	Ch. 5: 177
			Ch. 6: 218, 219, <b>254</b>
		<b>V_SYNC_SRC</b>	Ch. 6: 218, <b>249</b>
		<b>V_S_TX</b>	Ch. 5: 197, <b>198</b>
		<b>V_THRES_RX</b>	Ch. 4: <b>124, 131-134</b>
		<b>V_THRES_TX</b>	Ch. 4: <b>124, 131-134</b>
		<b>V_TI_IRQ</b>	Ch. 9: 289, 290, <b>302</b>
		<b>V_TI_IRQMSK</b>	Ch. 9: 289, 290, <b>296</b>

# Index of pin names

Index entries are sorted by name. Pages of the pin list are printed in bold type.

/CS	Ch. 1: 30, <b>32</b> Ch. 2: 40, 41, 43, 44, 46, 47	Ch. 1: 30, <b>35</b> Ch. 2: 39–41, 46, 47	Ch. 1: 30, <b>32</b> Ch. 2: 43, 44
/INT	Ch. 1: 30, 31, <b>32</b> Ch. 4: 142 Ch. 9: 286, 296, 297, 300	C2IO Ch. 1: 30, 31, <b>33</b> Ch. 6: 213, 220, 249	D7 Ch. 1: 30, <b>32</b> Ch. 2: 39, 40, 43, 44
/IOR	Ch. 1: 30, <b>35</b> Ch. 2: 40, 41	C4IO Ch. 1: 30, 31, <b>33</b> Ch. 4: 120 Ch. 6: 210, 213, 214, 216, 218, 220, 222, 226–228, 231, 234, 243, 248, 252, 253	DN0 Ch. 1: 31, <b>32</b> Ch. 2: 50
/IOW	Ch. 1: 30, <b>35</b> Ch. 2: 40, 41	CLK_OUT Ch. 10: 314, 319 Ch. 1: 30, 31, <b>33</b> Ch. 9: 276, 277, 295	DN1 Ch. 1: 31, <b>32</b> Ch. 2: 50
/RES	Ch. 1: 30, 31, <b>32</b> Ch. 9: 285	D0 Ch. 1: 30, <b>32</b> Ch. 2: 39, 40, 43, 44	DN2 Ch. 1: 31, <b>32</b> Ch. 2: 50
/SPISEL	Ch. 1: 29, 31, <b>32</b> Ch. 2: 50, 52–58	D1 Ch. 1: 30, <b>32</b> Ch. 2: 43, 44	DN3 Ch. 1: 31, <b>32</b> Ch. 2: 50
/WAIT	Ch. 1: 30, <b>35</b> Ch. 9: 293	D2 Ch. 1: 30, <b>32</b> Ch. 2: 43, 44	F0IO Ch. 1: 30, 31, <b>33</b> Ch. 4: 118, 120 Ch. 5: 156, 172, 177, 179, 180, 205 Ch. 6: 210, 213, 214, 216, 218, 222, 224, 226–229, 231, 234, 243, 247, 259 Ch. 9: 289, 290
/WD	Ch. 1: 30, 31, <b>33</b> Ch. 9: 292	D3 Ch. 1: 30, <b>32</b> Ch. 2: 43, 44	F1_0 Ch. 10: 314, 319 Ch. 1: 30, 31, <b>33</b> Ch. 6: 229, 244, 252
A0	Ch. 1: 30, <b>32</b> Ch. 2: 39–44	D4 Ch. 1: 30, <b>32</b> Ch. 2: 43, 44	F1_1 Ch. 10: 314, 319 Ch. 1: 30, 31, <b>33</b> Ch. 6: 229, 245, 252, 253 Ch. 9: 276, 277, 295
ADJ_0	Ch. 1: 30, 31, <b>35</b> Ch. 5: 158	D5 Ch. 1: 30, <b>32</b> Ch. 2: 43, 44	
ADJ_1	Ch. 1: 30, 31, <b>34</b> Ch. 5: 158	D6	
ALE			

GND			
Ch. 1: 30, 31, <b>32–35</b>			
GPIO0			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO1			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO10			
Ch. 10: 314–316, 320			
Ch. 1: 30, 31, <b>35</b>			
GPIO11			
Ch. 10: 312, 314–316, 320			
Ch. 1: 30, 31, <b>35</b>			
GPIO12			
Ch. 10: 314–316, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO13			
Ch. 10: 314–316, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO14			
Ch. 10: 314–316, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO15			
Ch. 10: 314–316, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO16			
Ch. 10: 312–314, 316, 317, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO17			
Ch. 10: 312, 314, 316, 317, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO18			
Ch. 10: 314, 316, 317, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO19			
Ch. 10: 314, 316, 317, 320			
Ch. 1: 30, 31, <b>34</b>			
GPIO2			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO3			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO4			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO5			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO6			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>32</b>			
GPIO7			
Ch. 10: 314, 318, 319, 321			
Ch. 1: 30, 31, <b>33</b>			
GPIO8			
Ch. 10: 312–316, 320			
Ch. 1: 30, 31, <b>35</b>			
GPIO9			
Ch. 10: 313–316, 320			
Ch. 1: 30, 31, <b>35</b>			
L_A0			
Ch. 10: 313, 314			
Ch. 1: 30, 31, <b>35</b>			
Ch. 5: 158			
L_A1			
Ch. 10: 314			
Ch. 1: 30, 31, <b>34</b>			
L_B0			
Ch. 10: 314			
Ch. 1: 30, 31, <b>35</b>			
Ch. 5: 158			
L_B1			
Ch. 10: 314			
Ch. 1: 30, 31, <b>34</b>			
MODE0			
Ch. 1: 30, 31, <b>33</b>			
Ch. 2: 38, 39, 50			
Ch. 9: 285			
MODE1			
Ch. 1: 30, 31, <b>33</b>			
Ch. 2: 38, 39, 50			
Ch. 9: 285			
NC			
Ch. 1: 31, <b>35</b>			
OSC_IN			
Ch. 1: 30, 31, <b>33</b>			
Ch. 9: 277, 279, 280, 295			
OSC_OUT			
Ch. 1: 30, 31, <b>33</b>			
Ch. 9: 276, 277, 279			
PWM0			
Ch. 10: 314, 319			
Ch. 1: 30, 31, <b>33</b>			
Ch. 7: 264–266			
PWM1			
Ch. 10: 314, 319			
Ch. 1: 30, 31, <b>33</b>			
Ch. 7: 264–266			
R_A0			
Ch. 10: 313, 314			
Ch. 1: 30, 31, <b>35</b>			
Ch. 5: 156–158, 172, 173			
R_A1			
Ch. 10: 314			
Ch. 1: 30, 31, <b>34</b>			
Ch. 5: 156, 172			
R_B0			
Ch. 10: 314			
Ch. 1: 30, 31, <b>35</b>			
Ch. 5: 156–158, 172, 173			
R_B1			
Ch. 10: 314			
Ch. 1: 30, 31, <b>34</b>			
Ch. 5: 156, 172			
SPI_CLK			
Ch. 1: 31, <b>35</b>			
Ch. 2: 50, 58			
SPI_INV			
Ch. 1: 31, <b>35</b>			
Ch. 2: 50, 58			
SPI_RX			
Ch. 1: 31, <b>32</b>			
Ch. 2: 50–53, 55–57			
SPI_TX			
Ch. 1: 31, <b>32</b>			
Ch. 2: 50–57			
STIO1			
Ch. 10: 314, 319			
Ch. 1: 30, 31, <b>33</b>			
Ch. 3: 81–84, 88, 89, 97, 108, 114			
Ch. 6: 210–214, 216, 226, 231, 232, 234, 235, 237, 247, 248, 257, 262			
STIO2			
Ch. 10: 314, 319			
Ch. 1: 30, 31, <b>33</b>			
Ch. 3: 82–84, 88, 89, 97, 109, 115			

Ch. 6: 210–214, 216, 226, 231, 232, 234, 235, 237, 248, 257, 262	Ch. 1: 30, 31, <b>34</b> Ch. 5: 157, 159, 173	Ch. 1: 30, 31, <b>32–35</b> , 36
SYNC_I	T_A1	VDD_SU0
Ch. 1: 30, 31, <b>32</b>	Ch. 10: 314	Ch. 10: 312–314
Ch. 6: 218, 220, 222, 223, 226, 247, 249, 250, 254	Ch. 1: 30, 31, <b>34</b>	Ch. 1: 30, 31, <b>34</b>
Ch. 8: 273	T_B0	Ch. 5: 159
SYNC_O	Ch. 10: 314	VDD_SU1
Ch. 10: 314, 319	Ch. 1: 30, 31, <b>34</b>	Ch. 10: 312, 314
Ch. 1: 30, 31, <b>32</b>	Ch. 5: 157, 159, 173	Ch. 1: 30, 31, <b>34</b>
Ch. 6: 220, 222, 223, 249	T_B1	Ch. 5: 159
T_A0	Ch. 10: 314	WAKEUP
Ch. 10: 313, 314	Ch. 1: 30, 31, <b>34</b>	Ch. 1: 30, 31, <b>33</b>
	VDD	Ch. 2: 62
	Ch. 10: 313, 314	Ch. 7: 266
		Ch. 9: 276, 293, 302, 304



**Cologne Chip AG**  
**Data Sheet of XHFC-2SU**

