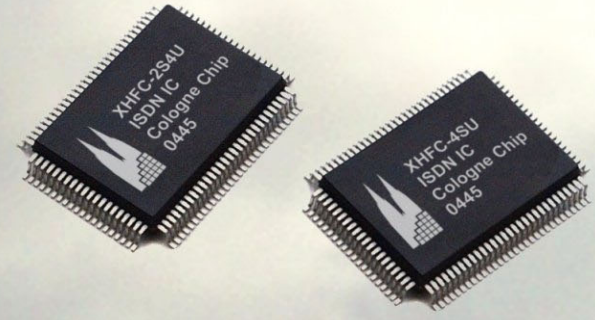
Cologne Chip

XHFC - 2S4U
XHFC - 4SU

Extended ISDN HDLC FIFO controller
with
multiple Universal ISDN Ports

(XHFC-2S4U: 2 ST / U_p interfaces and 2 U_p interfaces)
(XHFC-4SU: 4 ST / U_p interfaces)



Data Sheet ✦ May 2010

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

Date	Remarks
May 2010	No changes with regard to contents, only a few typographical corrections have been made.
April 2010	Sections 6.5.6, 6.5.7, 9.1.3.5, 10.3 and 13 added. Figures 9.3 and 10.2 added. Tables 2.3, 10.3, 13.1 and 13.2 added. Register R_SU_LED_CTRL added. Bit V_WD_EN in register R_BERT_WD_MD added. Reset groups and I/O characteristic of the pin list completely reworked. Section "Reset" also reworked. Register descriptions improved for e.g. A_B1_RX, A_B2_RX, A_D_RX, A_E_RX, A_FIFO_CTRL, A_SL_CFG, R_CHIP_ID, R_CIRM, R_CLK_CFG, R_FIFO_BL0_IRQ . . R_FIFO_BL3_IRQ, R_IRQ_CTRL, R_IRQ_OVIEW, R_PCM_MD1 and R_RAM_CTRL. Register descriptions clarified and corrected for A_CON_HDLC, A_F1, A_F2, A_SU_CTRL0, R_IRQ_OVIEW, R_PCM_MD1, R_STATUS. Schematics slightly changed for processor interface (recommended capacitor added), component values of the quarz circuitry improved (see also calculation procedure in section 9.1.3). /WAIT signal added to parallel processor interface timing diagrams. Signals /SPICLK, SPICLK_IDLE and /SPICLK_IDLE added to timing diagram 2.20. Internal signal FSC renamed to FSC_RX (and as a consequence of this FSC_0 and FSC_1 renamed to FSC_TX_0 and FSC_TX_1). Internal signal FSC_TX added to figures 5.1, 5.5, 5.13, 5.15 and 5.16 and to the corresponding sections. Note *1 changed in NT state matrices for S/T (Table 5.4) and U_p (Table 5.9) line interfaces. Remarks added for bit scrambler configuration in sections 5.3.4 and 5.3.7. Remark added for monitoring applications in section 5.3.6. Remark concerning internal PCM loop added to section 6.2. Register bit V_FIFO_IRQMSK added to Figure 9.7. Remark concerning unused line interfaces added to section 9.4.4.1. Chapter 11 reworked and supplemented.
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 12 and 13 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 9.5), PWM description and A_CON_HDLC register description.
February 2006	Corrections are made on pin list (pins 77 . . 81), PLL frequency range and PLL setup examples. External U_p circuitry has been improved. Power consumption information has been added.
December 2005	First edition of this data sheet.

Contents

List of Figures

List of Tables

List of Registers

Please note !

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

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

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

Multi-registers have multiple instances, too, but they are selected by a bitmap value. With this value, different registers can be selected with the same address. Multi-register addresses are 0x15 (11 instances selected by R_PCM_MD0), 0x0F (2 instances selected by R_FIFO_MD) and 0x35 (2 instances selected by A_ST_CTRL3 / A_UP_CTRL3) for XHFC-2S4U / 4SU. Multi-registers have different names, bitmap structure and meaning for each instance.

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

Registers sorted by name

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

See Table 9.4 on page 300 for a detailed reset group explanation.

Please note !

See explanation of register types on page 21.

Write only registers:

Address	Name	Reset group	Page
0x3C	A_B1_TX[ST/Up]	–	208
0x3D	A_B2_TX[ST/Up]	–	208
0x3F	A_BAC_S_TX[ST/Up]	–	210
0x3E	A_D_TX[ST/Up]	–	209
0x0E	A_INC_RES_FIFO[FIFO]	H, 0, 1, 2, 3	138
0x36	A_MS_DF[ST/Up]	H, 0, 3	206
0x34	A_MS_TX[ST/Up]	H, 0, 3	202
0x35	A_ST_CTRL3[ST/Up]	H, 0, 3	203
0x37	A_SU_CLK_DLY[ST/Up]	H, 0, 3	207
0x31	A_SU_CTRL0[ST/Up]	H, 0, 3	197
0x32	A_SU_CTRL1[ST/Up]	H, 0, 3	199
0x33	A_SU_CTRL2[ST/Up]	H, 0, 3	200
0x30	A_SU_WR_STA[ST/Up]	H, 0, 3	196
0x35	A_UP_CTRL3[ST/Up]	H, 0, 3	204
0x1B	R_BERT_WD_MD	H, 0	286
0x28	R_CI_TX	H, 0, 2	269
0x00	R_CIRM	H	308
0x02	R_CLK_CFG	H	310
0x01	R_CTRL	H	72
0x0D	R_FIFO_MD	H	137
0x0C	R_FIFO_THRES	H, 0, 1, 2, 3	136
0x0F	R_FIFO	H, 0, 1	139
0x0B	R_FIRST_FIFO	H, 0, 1	135
0x0F	R_FSM_IDX	H, 0, 1	139
0x29	R_GCI_CFG0	H, 0, 2	270
0x2A	R_GCI_CFG1	H, 0, 2	272
0x4A	R_GPIO_EN0	H, 0	339
0x42	R_GPIO_EN1	H, 0	334
0x47	R_GPIO_EN2	H, 0	337
0x43	R_GPIO_EN3	H, 0	335
0x48	R_GPIO_OUT0	H, 0	338
0x40	R_GPIO_OUT1	H, 0	333
0x45	R_GPIO_OUT2	H, 0	337
0x41	R_GPIO_OUT3	H, 0	334
0x44	R_GPIO_SEL_BL	H, 0	336
0x4C	R_GPIO_SEL	H, 0	340
0x13	R_IRQ_CTRL	H, 0	313
0x11	R_MISC_IRQMSK	H	311
0x2B	R_MON_TX	H, 0, 2	272

Address	Name	Reset group	Page
0x15	R_MSS0	H, 0, 2	259
0x15	R_MSS1	H, 0, 2	265
0x14	R_PCM_MD0	H, 0, 2	256
0x15	R_PCM_MD1	H, 0, 2	261
0x15	R_PCM_MD2	H, 0, 2	263
0x50	R_PLL_CTRL	H	315
0x1E	R_PWM_CFG	H, 0	279
0x46	R_PWM_MD	H, 0	280
0x38	R_PWM0	H, 0	279
0x39	R_PWM1	H, 0	280
0x08	R_RAM_ADDR	H, 0	73
0x09	R_RAM_CTRL	H, 0	73
0x15	R_SH0H	H, 0, 2	266
0x15	R_SH0L	H, 0, 2	266
0x15	R_SH1H	H, 0, 2	267
0x15	R_SH1L	H, 0, 2	266
0x15	R_SL_SEL0	H, 0, 2	257
0x15	R_SL_SEL1	H, 0, 2	258
0x15	R_SL_SEL7	H, 0, 2	258
0x10	R_SLOT	H, 0, 2	255
0x12	R_SU_IRQMSK	H, 0, 3	312
0x4D	R_SU_LED_CTRL	H, 0	341
0x16	R_SU_SEL	H, 0, 3	195
0x17	R_SU_SYNC	H, 0, 3	268
0x1A	R_TI_WD	H, 0	314

Read only registers:

Address	Name	Reset group	Page
0x3C	A_B1_RX[ST/Up]	–	217
0x3D	A_B2_RX[ST/Up]	–	218
0x3E	A_D_RX[ST/Up]	–	219
0x3F	A_E_RX[ST/Up]	–	220
0x0C	A_F1[FIFO]	H, 0, 1	140
0x0D	A_F2[FIFO]	H, 0, 1	141
0x0E	A_FIFO_STA[FIFO]	H, 0, 1	142
0x34	A_MS_RX[ST/Up]	H, 0, 3	215
0x32	A_SU_DLYH[ST/Up]	–	214
0x31	A_SU_DLYL[ST/Up]	–	213
0x30	A_SU_RD_STA[ST/Up]	H, 0, 3	212
0x35	A_SU_STA[ST/Up]	H, 0, 3	216
0x14	A_USAGE[FIFO]	H, 0, 1	143
0x04	A_Z1[FIFO]	H, 0, 1	140
0x06	A_Z2[FIFO]	H, 0, 1	140
0x13	R_AF0_OVIEW	H, 0, 3	211
0x1B	R_BERT_ECH	H, 0, 1	288
0x1A	R_BERT_ECL	H, 0, 1	287
0x17	R_BERT_STA	H, 0, 1	287
0x16	R_CHIP_ID	H	74
0x1F	R_CHIP_RV	H	74
0x28	R_CI_RX	–	274
0x19	R_F0_CNTH	H, 0, 1	273
0x18	R_F0_CNTL	H, 0, 1	273
0x20	R_FIFO_BL0_IRQ	H, 0, 1	321
0x21	R_FIFO_BL1_IRQ	H, 0, 1	322
0x22	R_FIFO_BL2_IRQ	H, 0, 1	323
0x23	R_FIFO_BL3_IRQ	H, 0, 1	324
0x24	R_FILL_BL0	H, 0, 1	143
0x25	R_FILL_BL1	H, 0, 1	144
0x26	R_FILL_BL2	H, 0, 1	145
0x27	R_FILL_BL3	H, 0, 1	146
0x29	R_GCI_STA	H, 0, 2	275
0x48	R_GPIO_IN0	–	344
0x40	R_GPIO_IN1	–	342
0x45	R_GPIO_IN2	–	343
0x41	R_GPIO_IN3	–	343
0x88	R_INT_DATA	–	75
0x10	R_IRQ_OVIEW	H, 0, 1	316
0x11	R_MISC_IRQ	H, 0, 1	318
0x2A	R_MON_RX	–	275
0x50	R_PLL_STA	H	324
0x15	R_RAM_USE	–	74
0x1D	R_SL_MAX	–	273
0x1C	R_STATUS	H, 0, 3	320
0x12	R_SU_IRQ	H, 0	319

Read / Write registers:

Address	Name	Reset group	Page
0xF4	A_CH_MSK[FIFO]	H, 0, 1	148
0xFC	A_CHANNEL[FIFO]	H, 0, 1	152
0xFA	A_CON_HDLC[FIFO]	H, 0, 1	149
0xFF	A_FIFO_CTRL[FIFO]	H, 0, 1	154
0x84	A_FIFO_DATA_NOINC[FIFO]	–	147
0x80	A_FIFO_DATA[FIFO]	–	147
0xFD	A_FIFO_SEQ[FIFO]	H, 0, 1	153
0xD0	A_SL_CFG[SLOT]	H, 0, 2	276
0xFB	A_SUBCH_CFG[FIFO]	H, 0, 1	151
0x52	R_PLL_N	H	325
0x51	R_PLL_P	H	325
0x53	R_PLL_S	H	325
0xC0	R_RAM_DATA	–	76

Registers sorted by address

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

See Table 9.4 on page 300 for a detailed reset group explanation.

Please note !

See explanation of register types on page 21.

Write only registers:

Address	Name	Reset group	Page
0x00	R_CIRM	H	308
0x01	R_CTRL	H	72
0x02	R_CLK_CFG	H	310
0x08	R_RAM_ADDR	H, 0	73
0x09	R_RAM_CTRL	H, 0	73
0x0B	R_FIRST_FIFO	H, 0, 1	135
0x0C	R_FIFO_THRES	H, 0, 1, 2, 3	136
0x0D	R_FIFO_MD	H	137
0x0E	A_INC_RES_FIFO[FIFO]	H, 0, 1, 2, 3	138
0x0F	R_FSM_IDX	H, 0, 1	139
0x0F	R_FIFO	H, 0, 1	139
0x10	R_SLOT	H, 0, 2	255
0x11	R_MISC_IRQMSK	H	311
0x12	R_SU_IRQMSK	H, 0, 3	312
0x13	R_IRQ_CTRL	H, 0	313
0x14	R_PCM_MD0	H, 0, 2	256
0x15	R_MSS0	H, 0, 2	259
0x15	R_MSS1	H, 0, 2	265
0x15	R_PCM_MD1	H, 0, 2	261
0x15	R_PCM_MD2	H, 0, 2	263
0x15	R_SH0H	H, 0, 2	266
0x15	R_SH1H	H, 0, 2	267
0x15	R_SH0L	H, 0, 2	266
0x15	R_SH1L	H, 0, 2	266
0x15	R_SL_SEL0	H, 0, 2	257
0x15	R_SL_SEL1	H, 0, 2	258
0x15	R_SL_SEL7	H, 0, 2	258
0x16	R_SU_SEL	H, 0, 3	195
0x17	R_SU_SYNC	H, 0, 3	268
0x1A	R_TI_WD	H, 0	314
0x1B	R_BERT_WD_MD	H, 0	286
0x1E	R_PWM_CFG	H, 0	279
0x28	R_CI_TX	H, 0, 2	269
0x29	R_GCI_CFG0	H, 0, 2	270
0x2A	R_GCI_CFG1	H, 0, 2	272
0x2B	R_MON_TX	H, 0, 2	272
0x30	A_SU_WR_STA[ST/Up]	H, 0, 3	196
0x31	A_SU_CTRL0[ST/Up]	H, 0, 3	197
0x32	A_SU_CTRL1[ST/Up]	H, 0, 3	199
0x33	A_SU_CTRL2[ST/Up]	H, 0, 3	200
0x34	A_MS_TX[ST/Up]	H, 0, 3	202
0x35	A_ST_CTRL3[ST/Up]	H, 0, 3	203
0x35	A_UP_CTRL3[ST/Up]	H, 0, 3	204
0x36	A_MS_DF[ST/Up]	H, 0, 3	206
0x37	A_SU_CLK_DLY[ST/Up]	H, 0, 3	207
0x38	R_PWM0	H, 0	279
0x39	R_PWM1	H, 0	280
0x3C	A_B1_TX[ST/Up]	–	208
0x3D	A_B2_TX[ST/Up]	–	208
0x3E	A_D_TX[ST/Up]	–	209
0x3F	A_BAC_S_TX[ST/Up]	–	210
0x40	R_GPIO_OUT1	H, 0	333
0x41	R_GPIO_OUT3	H, 0	334
0x42	R_GPIO_EN1	H, 0	334
0x43	R_GPIO_EN3	H, 0	335
0x44	R_GPIO_SEL_BL	H, 0	336
0x45	R_GPIO_OUT2	H, 0	337
0x46	R_PWM_MD	H, 0	280
0x47	R_GPIO_EN2	H, 0	337
0x48	R_GPIO_OUT0	H, 0	338
0x4A	R_GPIO_EN0	H, 0	339
0x4C	R_GPIO_SEL	H, 0	340
0x4D	R_SU_LED_CTRL	H, 0	341
0x50	R_PLL_CTRL	H	315

Read only registers:

Address	Name	Reset group	Page
0x04	A_Z1[FIFO]	H, 0, 1	140
0x06	A_Z2[FIFO]	H, 0, 1	140
0x0C	A_F1[FIFO]	H, 0, 1	140
0x0D	A_F2[FIFO]	H, 0, 1	141
0x0E	A_FIFO_STA[FIFO]	H, 0, 1	142
0x10	R_IRQ_OVIEW	H, 0, 1	316
0x11	R_MISC_IRQ	H, 0, 1	318
0x12	R_SU_IRQ	H, 0	319
0x13	R_AF0_OVIEW	H, 0, 3	211
0x14	A_USAGE[FIFO]	H, 0, 1	143
0x15	R_RAM_USE	–	74
0x16	R_CHIP_ID	H	74
0x17	R_BERT_STA	H, 0, 1	287
0x18	R_F0_CNTL	H, 0, 1	273
0x19	R_F0_CNTH	H, 0, 1	273
0x1A	R_BERT_ECL	H, 0, 1	287
0x1B	R_BERT_ECH	H, 0, 1	288
0x1C	R_STATUS	H, 0, 3	320
0x1D	R_SL_MAX	–	273
0x1F	R_CHIP_RV	H	74
0x20	R_FIFO_BL0_IRQ	H, 0, 1	321
0x21	R_FIFO_BL1_IRQ	H, 0, 1	322
0x22	R_FIFO_BL2_IRQ	H, 0, 1	323
0x23	R_FIFO_BL3_IRQ	H, 0, 1	324
0x24	R_FILL_BL0	H, 0, 1	143
0x25	R_FILL_BL1	H, 0, 1	144
0x26	R_FILL_BL2	H, 0, 1	145
0x27	R_FILL_BL3	H, 0, 1	146
0x28	R_CI_RX	–	274
0x29	R_GCI_STA	H, 0, 2	275
0x2A	R_MON_RX	–	275
0x30	A_SU_RD_STA[ST/Up]	H, 0, 3	212
0x31	A_SU_DLYL[ST/Up]	–	213
0x32	A_SU_DLYH[ST/Up]	–	214
0x34	A_MS_RX[ST/Up]	H, 0, 3	215
0x35	A_SU_STA[ST/Up]	H, 0, 3	216
0x3C	A_B1_RX[ST/Up]	–	217
0x3D	A_B2_RX[ST/Up]	–	218
0x3E	A_D_RX[ST/Up]	–	219
0x3F	A_E_RX[ST/Up]	–	220
0x40	R_GPIO_IN1	–	342
0x41	R_GPIO_IN3	–	343
0x45	R_GPIO_IN2	–	343
0x48	R_GPIO_IN0	–	344
0x50	R_PLL_STA	H	324
0x88	R_INT_DATA	–	75

Read / Write registers:

Address	Name	Reset group	Page
0x51	R_PLL_P	H	325
0x52	R_PLL_N	H	325
0x53	R_PLL_S	H	325
0x80	A_FIFO_DATA[FIFO]	–	147
0x84	A_FIFO_DATA_NOINC[FIFO]	–	147
0xC0	R_RAM_DATA	–	76
0xD0	A_SL_CFG[SLOT]	H, 0, 2	276
0xF4	A_CH_MSK[FIFO]	H, 0, 1	148
0xFA	A_CON_HDLC[FIFO]	H, 0, 1	149
0xFB	A_SUBCH_CFG[FIFO]	H, 0, 1	151
0xFC	A_CHANNEL[FIFO]	H, 0, 1	152
0xFD	A_FIFO_SEQ[FIFO]	H, 0, 1	153
0xFF	A_FIFO_CTRL[FIFO]	H, 0, 1	154

About this data sheet and Cologne Chip technical support

This data sheet covers all the features of XHFC-2S4U / 4SU. The reader who absorbs the information in this data sheet will gain a deep and broad understanding of XHFC-2S4U and XHFC-4SU microchips.

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-2S4U / 4SU 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-2S4U/4SU, 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-2S4U/4SU 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 inferface 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-2S4U / XHFC-4SU 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-2S4U/4SU.)

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-2S4U/4SU has several 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-2S4U/4SU. (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-2S4U/4SU are given in this chapter. (Information for hardware design.)

Chapter "XHFC-2S4U / XHFC-4SU package dimensions" (12) shows the XHFC-2S4U/4SU package dimensions. (Information for hardware design.)

General remarks to notations

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

Chapter 1

General description

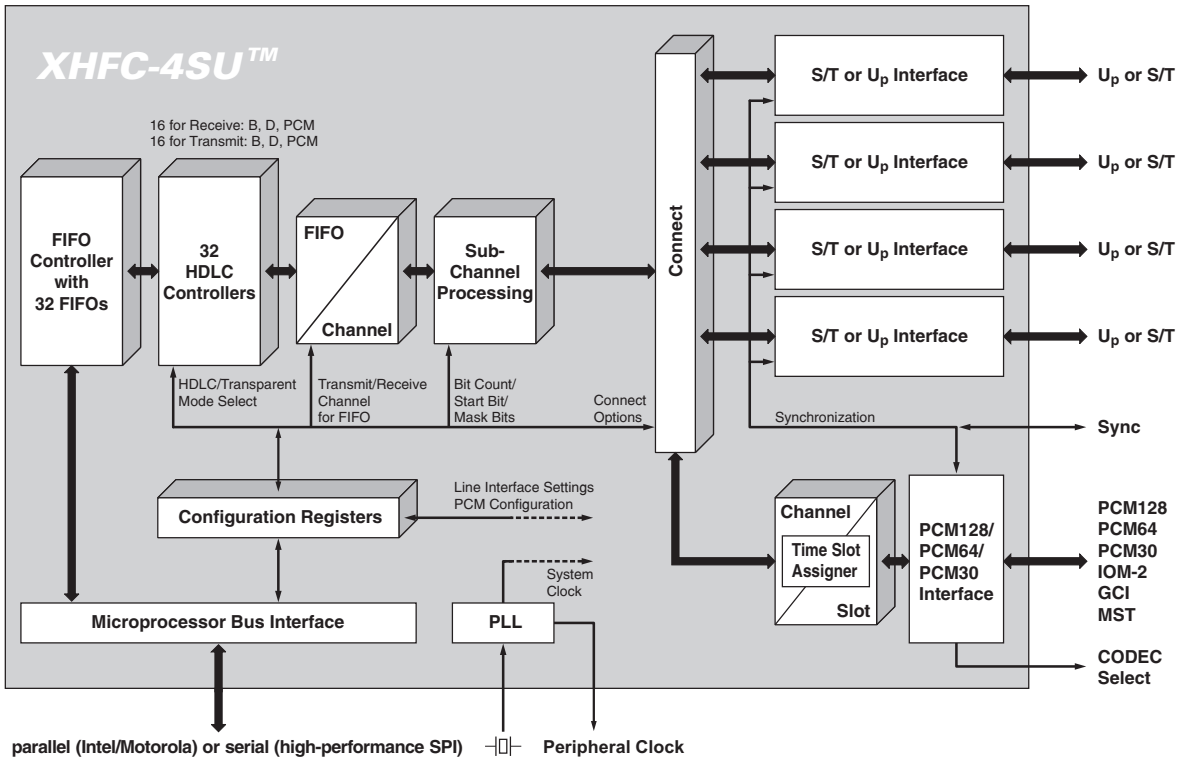


Figure 1.1: *XHFC-4SU block diagram*

1.1 System overview

XHFC-2S4U / 4SU are single-chip ISDN transceiver for four ISDN S/T or U_{pN}/ U_{p0}[1] Basic Rate Interfaces with integrated HDLC controllers for all kinds of BRI equipment, such as

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

XHFC-2S4U has two Universal ISDN Ports which can be configurated either in S/T mode or in U_p mode (see Chapter 5), numbered 0 and 1. Two additional U_p interfaces 2 and 3 are available.

XHFC-4SU has four Universal ISDN Ports which can be configurated either in S/T mode or in U_p mode, numbered 0 . . 3.

The integrated microprocessor bus interface of XHFC-2S4U / 4SU 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-2S4U / 4SU are realized with an internal SRAM.

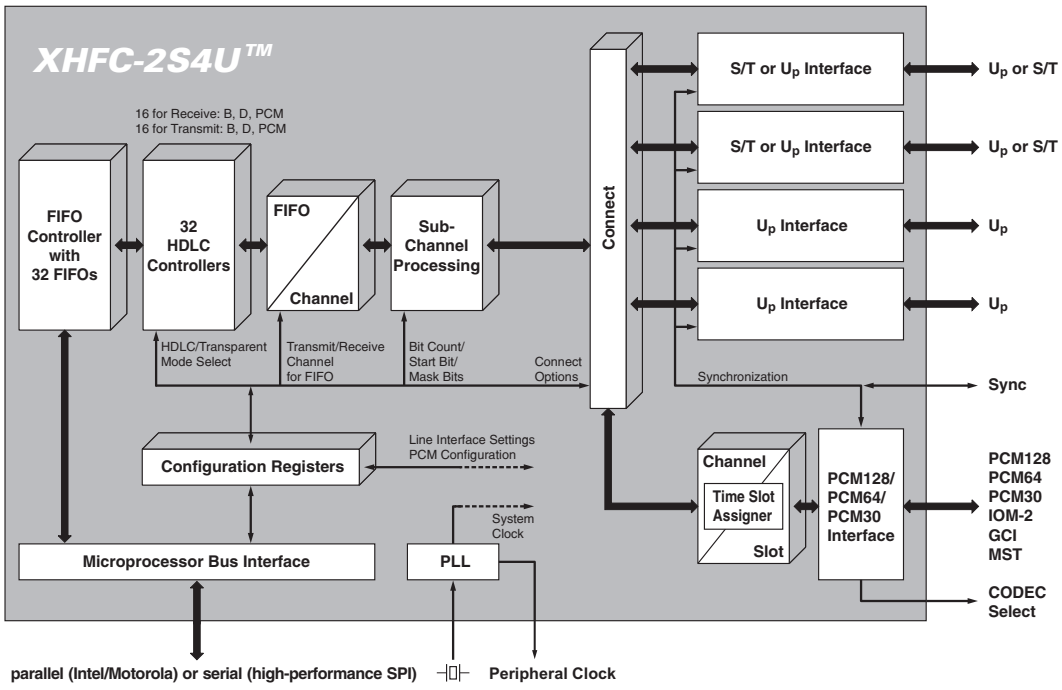


Figure 1.2: *XHFC-2S4U block diagram*

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

1.2 Features

Line interfaces

- **XHFC-2S4U:** 4 ISDN interfaces, two are selectable as S/T or U_{pN}/ U_{p0} interfaces (Universal ISDN Ports), two operate always as U_{pN}/ U_{p0} interfaces
- **XHFC-4SU:** 4 ISDN interfaces, each selectable as S/T or U_{pN}/ U_{p0} interfaces (Universal ISDN Ports)
- S/T ISDN interfaces in TE and NT mode conform to ITU-T I.430 and TBR 3 [9, 4]
- U_p signal range exceeding U_{pN}/ U_{p0} specification [3]
- Simple external line interface circuitry

HDLC-controller and FIFO controller

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

PCM interface

- PCM128 / PCM64 / PCM30 interface configurable to MST (MVIP) [2] or Siemens IOM™-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)

Miscelleanous 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
- PQFP 100 package, 0.65 mm pin pitch
- RoHS compliant

[2] Mitel Serial Telecom bus

1.3 Pin description

1.3.1 Pinout diagram

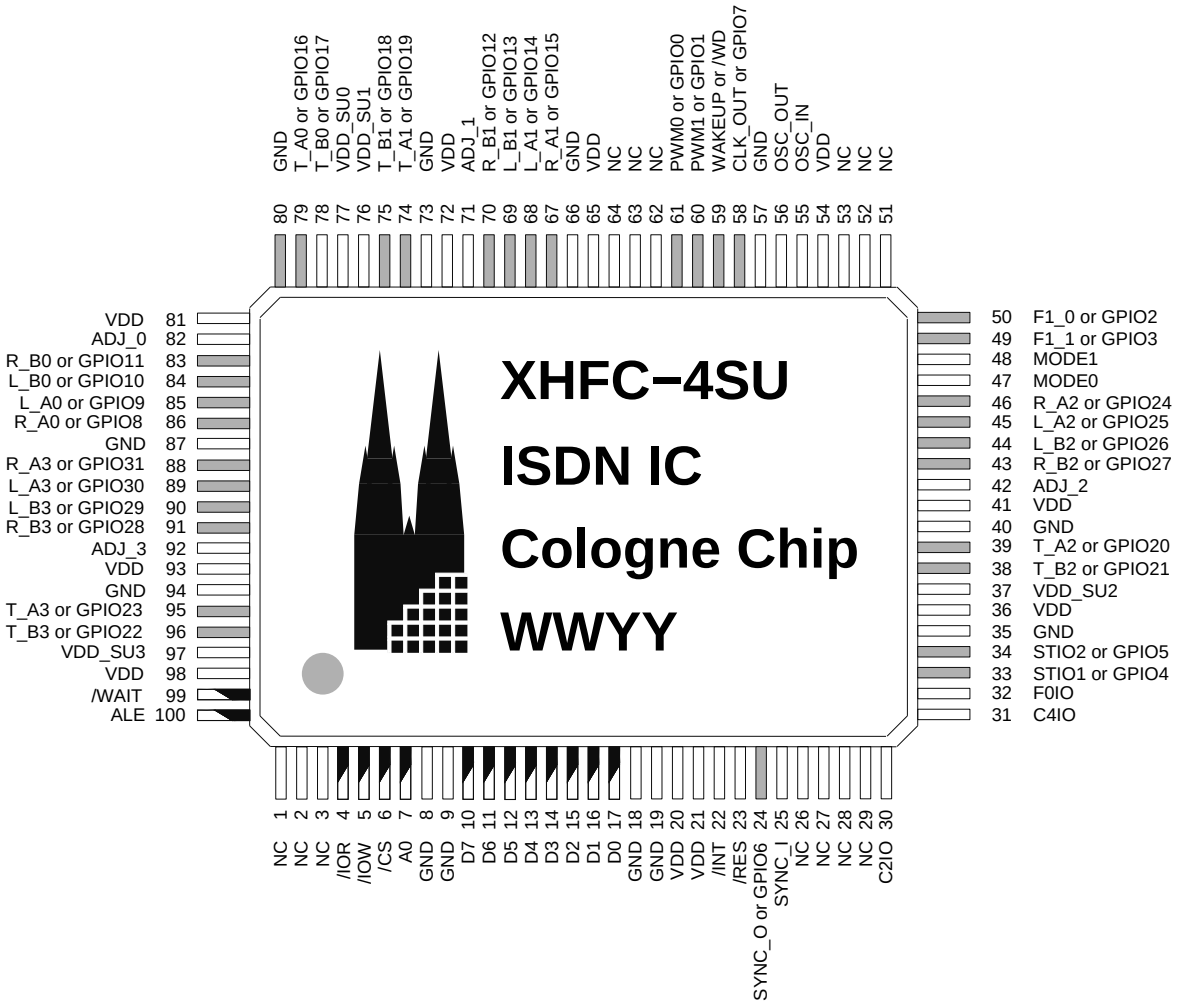


Figure 1.3: *XHFC-4SU pinout in parallel processor interface mode*

Note: XHFC-2S4U pinning is identical with XHFC-4SU.

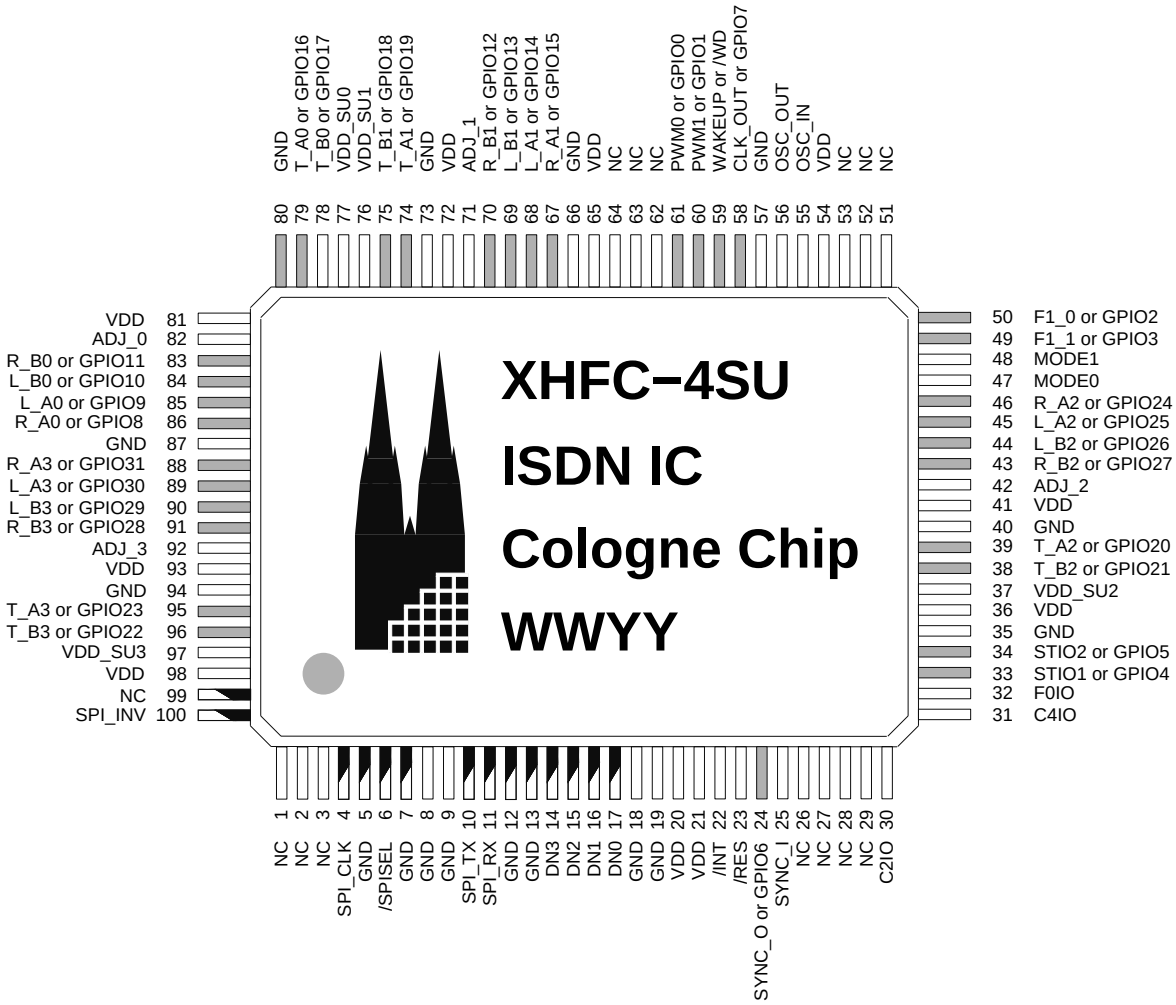


Figure 1.4: *XHFC-4SU pinout in serial processor interface mode*

Note: XHFC-2S4U pinning is identical with XHFC-4SU.

1.3.2 Pin list

Pin	Function	Name	I/O (type)	Description	Input	Output
		Microprocessor bus interface				
1		NC		Must not be connected		
2		NC		Must not be connected		
3		NC		Must not be connected		
4	Processor SPI	/IOR SPI_CLK	I (2 #) I (2 #)	Read enable SPI clock input	TTL TTL	
5	Processor SPI	/IOW GND	I (2 #)	Write enable Ground	TTL	
6	Processor SPI	/CS /SPISEL	I (2 #) I (2 #)	Chip select, active low SPI device select, active low	TTL TTL	
7	Processor SPI	A0 GND	I (2 #)	Address Ground	TTL	
8		GND		Ground		
9		GND		Ground		
10	Processor SPI	D7 SPI_TX	IO (1 #) O (1)	Data bit 7 SPI transmit data output	TTL	8 mA 8 mA
11	Processor SPI	D6 SPI_RX	IO (1 #) I (1 #)	Data bit 6 SPI receive data input	TTL TTL	8 mA
12	Processor SPI	D5 GND	IO (1 #)	Data bit 5 Ground	TTL	8 mA
13	Processor SPI	D4 GND	IO (1 #)	Data bit 4 Ground	TTL	8 mA
14	Processor SPI	D3 DN3	IO (1 #) I (1 #)	Data bit 3 SPI device number, bit 3	TTL TTL	8 mA
15	Processor SPI	D2 DN2	IO (1 #) I (1 #)	Data bit 2 SPI device number, bit 2	TTL TTL	8 mA
16	Processor SPI	D1 DN1	IO (1 #) I (1 #)	Data bit 1 SPI device number, bit 1	TTL TTL	8 mA
17	Processor SPI	D0 DN0	IO (1 #) I (1 #)	Data bit 0 SPI device number, bit 0	TTL TTL	8 mA
18		GND		Ground		
19		GND		Ground		

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	Input	Output
20		VDD		+3.3 V power supply		
21		VDD		+3.3 V power supply		
			Miscellaneous			
22		/INT	Ood #	Interrupt request pin, active low / high programmable		8 mA
23		/RES	Isch #	Reset input pin, active low	TTL	
			PCM interface			
24	1st function	SYNC_O	IO (1 #)	Synchronization output	TTL	8 mA
	2nd function	GPIO6	IO (1 #)	General purpose I/O pin 6	TTL	8 mA
25		SYNC_I	I (2 #)	Synchronization input	TTL	
26		NC		Must not be connected		
27		NC		Must not be connected		
28		NC		Must not be connected		
29		NC		Must not be connected		
30		C2IO	IOpu #	PCM bit clock	TTL	8 mA
31		C4IO	IOpu #	PCM double bit clock	TTL	8 mA
32		F0IO	IOpu #	PCM frame clock (8 kHz)	TTL	8 mA
33	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
34	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
35		GND		Ground		
36		VDD		+3.3 V power supply		
			Universal ISDN Ports			
37		VDD_SU2		Power supply for ST / Up interface no. 3		
38	1st function	T_B2	O (2)	Combined ST / Up interface no. 2 transmit output B		S/T / Up
	2nd function	GPIO21	IO (3)	General purpose I/O pin 21	TTL	16 mA

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	Input	Output
39	1st function	T_A2	O (2)	Combined ST / Up interface no. 2 transmit output A		S/T / Up
	2nd function	GPIO20	IO (3)	General purpose I/O pin 20	TTL	16 mA
40		GND		Ground		
41		VDD		+3.3 V power supply		
42		ADJ_2	Ood #	Combined ST / Up interface no. 2 level generator		2 mA
43	1st function	R_B2	I (3)	Combined ST / Up interface no. 2 receive input B	S/T / Up	
	2nd function	GPIO27	IO (3)	General purpose I/O pin 27	TTL	3 mA
44	1st function	L_B2	I (3)	Combined ST / Up interface no. 2 level detect B	S/T / Up	
	2nd function	GPIO26	IO (3)	General purpose I/O pin 26	TTL	3 mA
45	1st function	L_A2	I (3)	Combined ST / Up interface no. 2 level detect A	S/T / Up	
	2nd function	GPIO25	IO (3)	General purpose I/O pin 25	TTL	3 mA
46	1st function	R_A2	I (3)	Combined ST / Up interface no. 2 receive input A	S/T / Up	
	2nd function	GPIO24	IO (3)	General purpose I/O pin 24	TTL	3 mA
				Miscellaneous		
47		MODE0	I (2 #)	Interface mode pin 0	TTL	
48		MODE1	I (2 #)	Interface mode pin 1	TTL	
49	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
50	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
51		NC		Must not be connected		
52		NC		Must not be connected		
53		NC		Must not be connected		
				Clock		
54		VDD		+3.3 V power supply		
55		OSC_IN	I (4)	Oscillator input signal	Oscillator	
56		OSC_OUT	O (3)	Oscillator output signal		Oscillator

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	Input	Output
57		GND		Ground		
58	1st function	CLK_OUT	O (1)	Clock output signal		8 mA
	2nd function	GPIO7	IO (1 #)	General purpose I/O pin 7	TTL	8 mA
				Miscellaneous		
59	1st function	WAKEUP	I (1 #)	Wakeup input pin for external awake circuitry	TTL	
	2nd function	/WD	Ood #	Watchdog output signal		8 mA
60	1st function	PWM1	O (1)	Pulse width modulator output 1		
	2nd function	GPIO1	IO (1 #)	General purpose I/O pin 1	TTL	
61	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
62		NC		Must not be connected		
63		NC		Must not be connected		
64		NC		Must not be connected		
65		VDD		+3.3 V power supply		
66		GND		Ground		
				Universal ISDN Ports		
67	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
68	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
69	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
70	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
71		ADJ_1	Ood #	Combined ST / Up interface no. 1 level generator		2 mA
72		VDD		+3.3 V power supply		
73		GND		Ground		

(continued on next page)

(continued from previous page)

Pin	Function	Name	I/O (type)	Description	Input	Output
74	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
75	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
76		VDD_SU1		Power supply for ST / Up interface no. 1		
77		VDD_SU0		Power supply for ST / Up interface no. 0		
78	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
79	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
80		GND		Ground		
81		VDD		+3.3 V power supply		
82		ADJ_0	Ood #	Combined ST / Up interface no. 0 level generator		2 mA
83	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
84	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
85	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
86	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
87		GND		Ground		
88	1st function	R_A3	I (3)	Combined ST / Up interface no. 3 receive input A	S/T / Up	
	2nd function	GPIO31	IO (3)	General purpose I/O pin 31	TTL	3 mA
89	1st function	L_A3	I (3)	Combined ST / Up interface no. 3 level detect A	S/T / Up	
	2nd function	GPIO30	IO (3)	General purpose I/O pin 30	TTL	3 mA

(continued on next page)

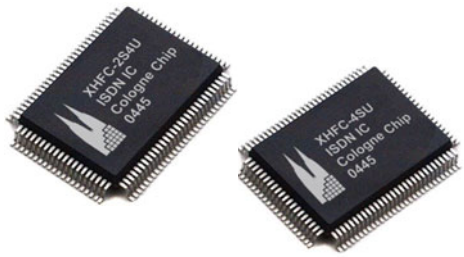
(continued from previous page)

Pin	Function	Name	I/O (type)	Description	Input	Output
90	1st function	L_B3	I (3)	Combined ST / Up interface no. 3 level detect B	S/T / Up	
	2nd function	GPIO29	IO (3)	General purpose I/O pin 29	TTL	3 mA
91	1st function	R_B3	I (3)	Combined ST / Up interface no. 3 receive input B	S/T / Up	
	2nd function	GPIO28	IO (3)	General purpose I/O pin 28	TTL	3 mA
92		ADJ_3	Ood #	Combined ST / Up interface no. 3 level generator		2 mA
93		VDD		+3.3 V power supply		
94		GND		Ground		
95	1st function	T_A3	O (2)	Combined ST / Up interface no. 3 transmit output A		S/T / Up
	2nd function	GPIO23	IO (3)	General purpose I/O pin 23	TTL	16 mA
96	1st function	T_B3	O (2)	Combined ST / Up interface no. 3 transmit output B		S/T / Up
	2nd function	GPIO22	IO (3)	General purpose I/O pin 22	TTL	16 mA
97		VDD_SU3		Power supply for ST / Up interface no. 3		
		Microprocessor bus interface				
98		VDD		+3.3 V power supply		
99	Processor	/WAIT	Ood #	Wait signal for external processor, active low		8 mA
	SPI	NC		Must not be connected		
100	Processor	ALE	I (2 #)	Address latch enable (only in multiplexed modes)	TTL	
	SPI	SPI_INV	I (2 #)	Invert SPI clock	TTL	

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

Pins with 5 V tolerant input are marked with #.

Unused input pins should be connected to ground. Unused I/O pins should be connected with pull-down resistor to ground.



Chapter 2

Microprocessor bus interface

Table 2.1: *Overview of the XHFC-2S4U / 4SU bus interface registers*

Write only registers:

Address	Name	Page
0x01	R_CTRL	72
0x08	R_RAM_ADDR	73
0x09	R_RAM_CTRL	73

Read only registers:

Address	Name	Page
0x15	R_RAM_USE	74
0x16	R_CHIP_ID	74
0x1F	R_CHIP_RV	74
0x88	R_INT_DATA	75

Read / write registers:

Address	Name	Page
0xC0	R_RAM_DATA	76

2.1 Mode selection

XHFC-2S4U / 4SU has an integrated microprocessor bus interface which can be configured as parallel 8 bit microprocessor interface and serial processor interface (SPI). Table 2.2 shows how to select these bus modes via the two pins MODE0 and MODE1 .

Table 2.2: *Microprocessor access types*

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

The Auto-EEPROM mode lets XHFC-2S4U / 4SU 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-2S4U / 4SU in the different bus modes.

2.2 Parallel processor interface

Table 2.3: *Overview of the parallel processor interface pins*

Number	Name	Description
22	/INT	Interrupt request pin, active low / high programmable
23	/RES	Reset input pin, active low
47	MODE0	Interface mode pin 0
48	MODE1	Interface mode pin 1
99	/WAIT	Wait signal for external processor, active low
100	ALE	Address latch enable (only in multiplexed modes)
4	/IOR	Read enable
5	/IOW	Write enable
6	/CS	Chip select, active low
7	A0	Address
17 . . 10	D0 . . D7	Data bit 0 . . Data bit 7

2.2.1 Overview

XHFC-2S4U / 4SU has four different parallel microprocessor interface modes. According to the name conventions of other Cologne Chip products (HFC series) the non-multiplexed processor interface modes are numbered 2 and 3 like shown in Table 2.4. The corresponding multiplexed modes are named 2m and 3m.[1]

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

Multiplexed modes are selected after reset with the first rising edge of ALE . XHFC-2S4U / 4SU then switches permanently from mode 2 into mode 2m or from mode 3 into mode 3m respectively. XHFC-2S4U / 4SU cannot switch to multiplexed modes before end of reset time. Rising and falling edges of ALE are ignored during reset time.

2.2.2 Interface signals

The processor interface signals have different names for Motorola and Intel microprocessors. Table 2.5 shows the mapping with the pin names of XHFC-2S4U / 4SU.

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

Table 2.4: *Parallel processor interface mode selection*

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

*1: This pin should be fixed to ground

*2: 1-pulse latches register address

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

XHFC-2S4U / 4SU pins		Signal names			
Number	Name	Mode 2 Motorola Non-multiplexed	Mode 3 Intel Non-multiplexed	Mode 2m Motorola Multiplexed	Mode 3m Intel Multiplexed
6	/CS	/CS	/CS	/CS	/CS
4	/IOR	/DS	/RD	/DS	/RD
5	/IOW	R/W	/WR	R/W	/WR
100	ALE	'0'	'0'	ALE	ALE
7	A0	A0	A0	'0'	'0'
10..17	D7..D0	D7..D0	D7..D0	AD7..AD0	AD7..AD0

2.2.3 Register access

2.2.3.1 Non-multiplexed / multiplexed access

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

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

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

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

2.2.3.2 Read* access

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

The Read* access performs two consecutive read accesses to XHFC-2S4U / 4SU:

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

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

2.2.3.3 Register address read-back capability

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

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

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

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

This procedure is important to avoid data read or write to an unexpected register address after a register read or write access has been split by an interrupt service routine which executes any access to XHFC-2S4U / 4SU.

2.2.4 Signal and timing characteristics

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

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

Read access

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

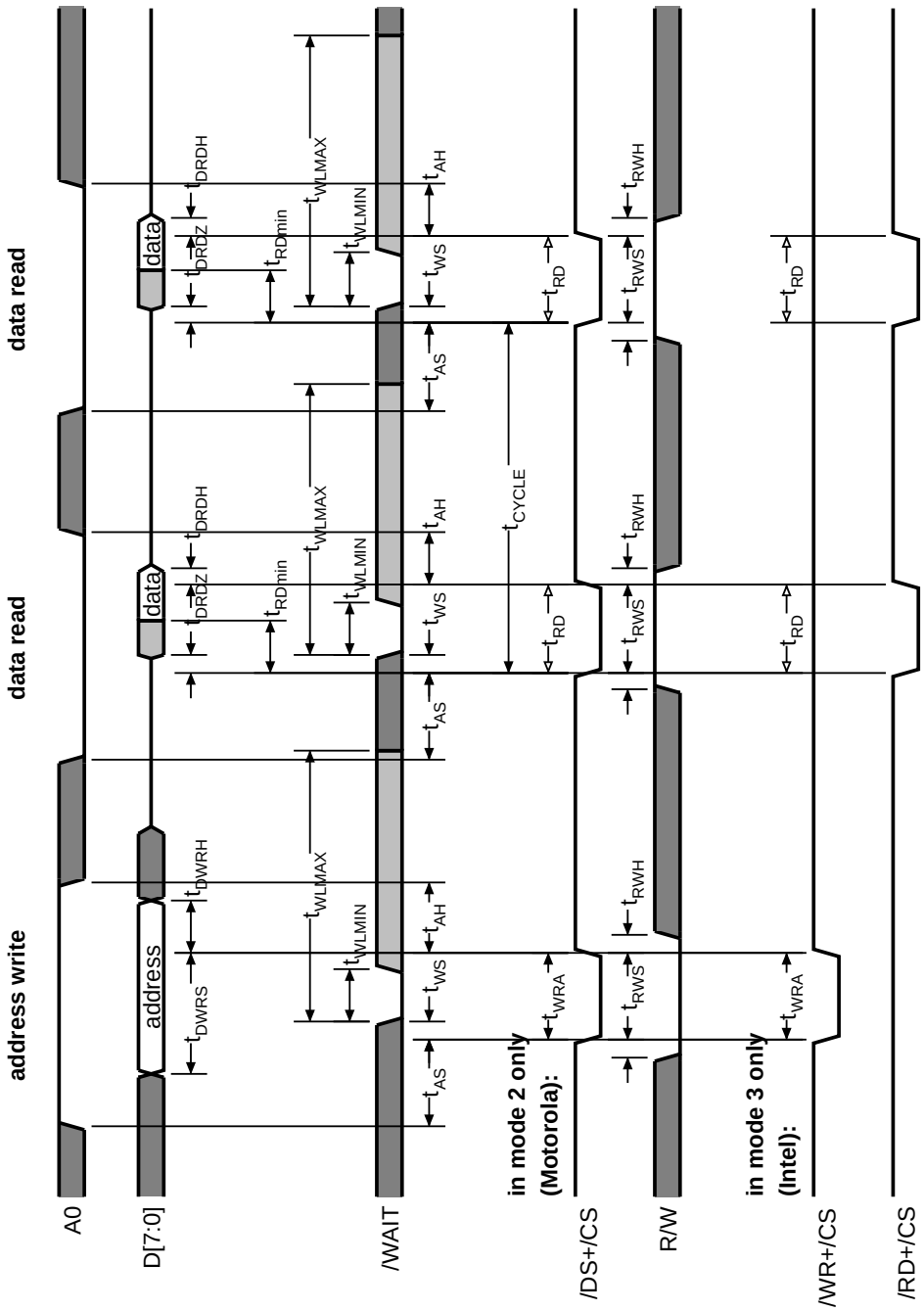


Figure 2.1: *Bus interface read access in mode 2 (Motorola) and mode 3 (Intel)*

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

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

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

*: 1-pulse latches register address

Table 2.7: *Timing diagrams of the parallel microprocessor interface*

Mode	Access type	Timing Figure	Timing on page	Timing values table	Timing values on page
2 & 3	read	2.1	48	2.8	50
2 & 3	write	2.2	51	2.9	52
2m & 3m	read	2.3	54	2.10	53
2m & 3m	write	2.4	56	2.11	55

Data can be read with [2]

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

in mode 2 (Motorola) or with

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

in mode 3 (Intel). The data bus is stable after $t_{RD\,min}$ and returns into tristate after t_{DRDH}. The address line A0 requires a setup time t_{AS}. The hold time of this line is t_{AH}.

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

Table 2.8: *Symbols of read accesses in Figure 2.1*

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

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

Write access

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

Data is written with ⌐ of (/DS + /CS) in mode 2 (Motorola) or with ⌐ of (/WR + /CS) in mode 3 (Intel) respectively. XHFC-2S4U / 4SU requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

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

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

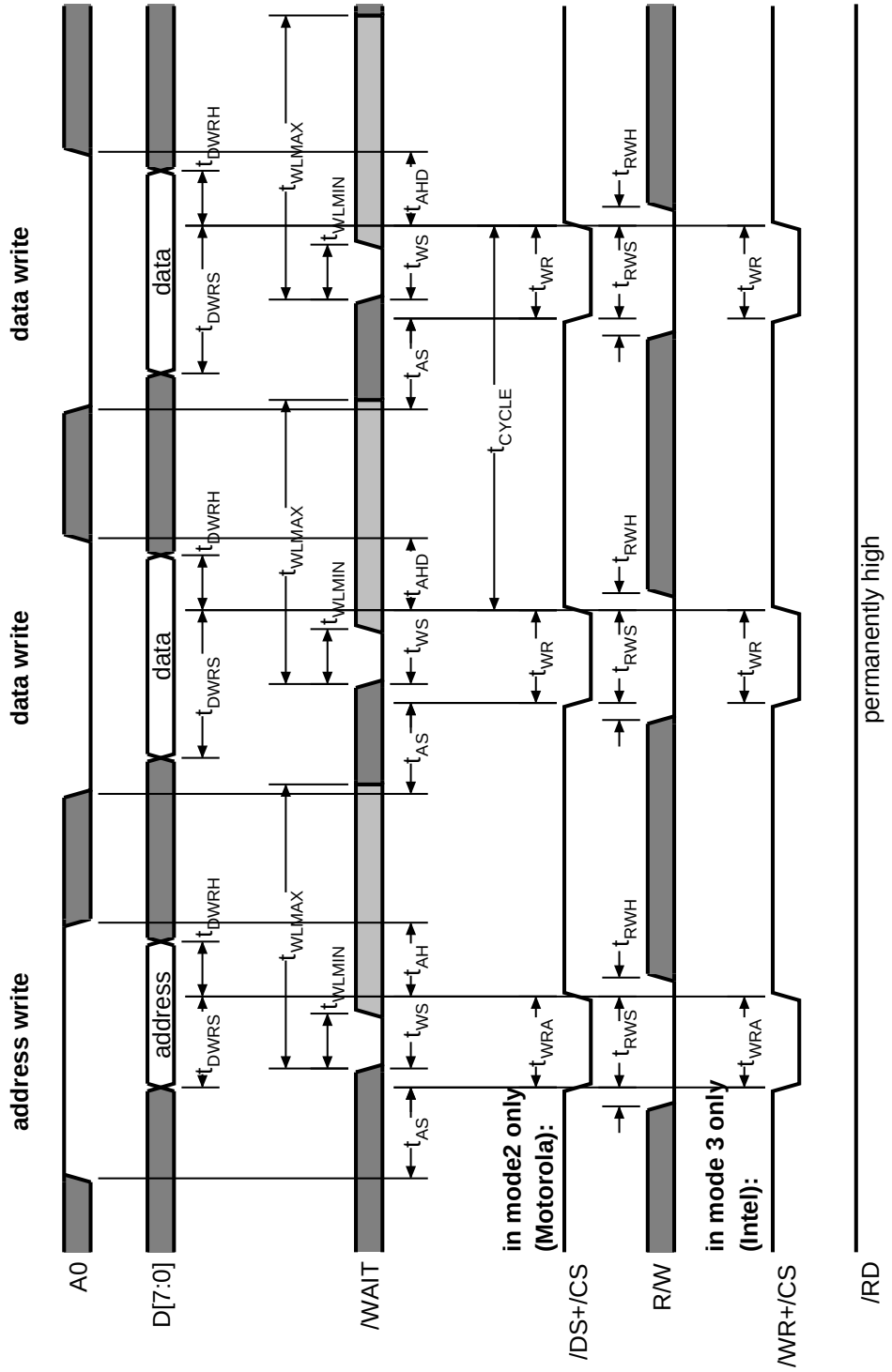


Figure 2.2: *Bus interface write access in mode 2 (Motorola) and mode 3 (Intel)*

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

Table 2.9: *Symbols of write accesses in Figure 2.2*

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

Table 2.10: *Symbols of read accesses in Figure 2.3*

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

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

Read access

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

Data can be read with [4]

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

in mode 2m (Motorola) or with

$$/WR = '1'$$

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

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

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

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

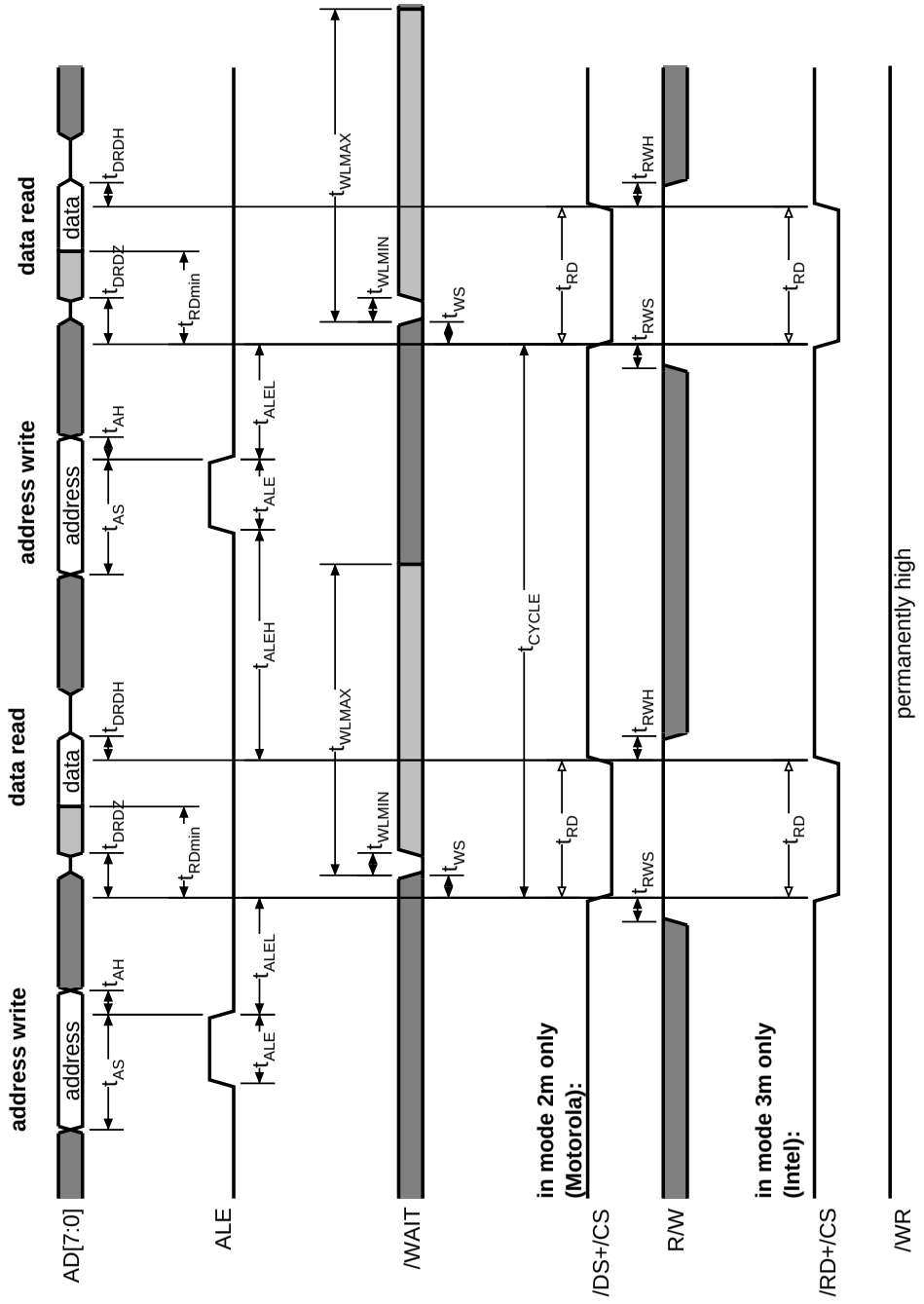


Figure 2.3: *Bus interface read access in modes 2m and 3m (multiplexed)*

Table 2.11: *Symbols of write accesses in Figures 2.4*

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

Write access

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

Data is written with ⌟ of (/DS + /CS) in mode 2m (Motorola) or with ⌟ of (/WR + /CS) in mode 3m (Intel) respectively. XHFC-2S4U / 4SU requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

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

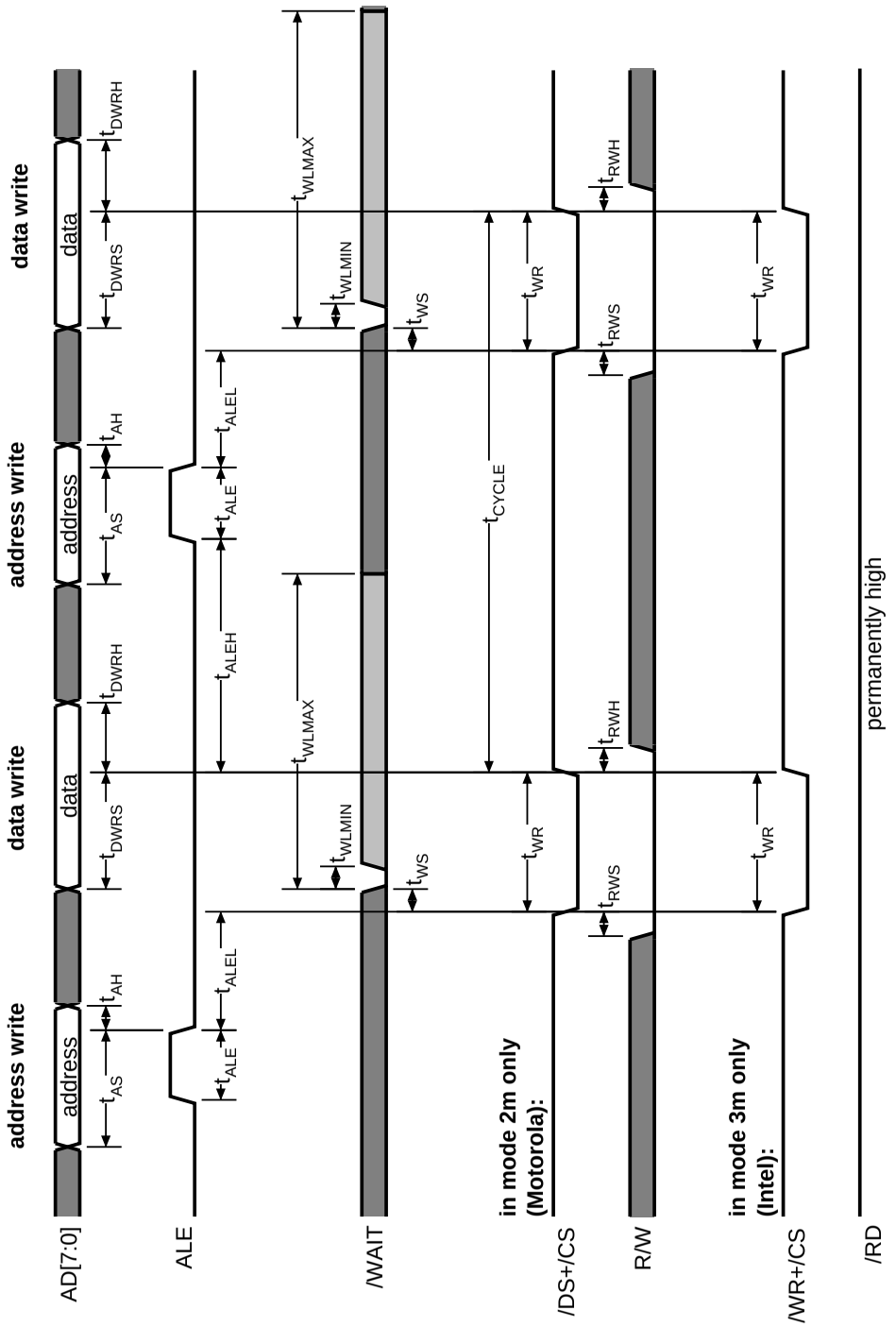


Figure 2.4: *Bus interface write access in mode 2m and 3m (multiplexed)*

2.2.5 Microprocessor connection circuitries

Figures 2.5 to 2.8 show examples how to connect XHFC-2S4U / 4SU to different parallel processor interfaces.

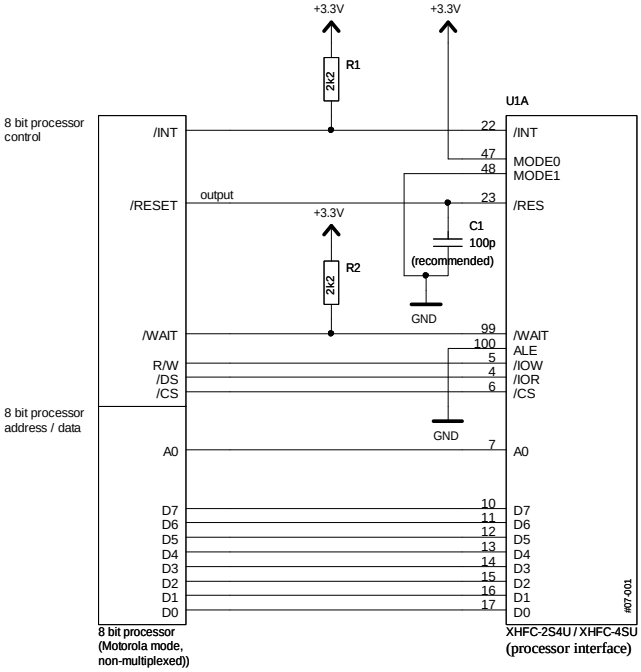


Figure 2.5: *8 bit Motorola processor circuitry example (mode 2)*

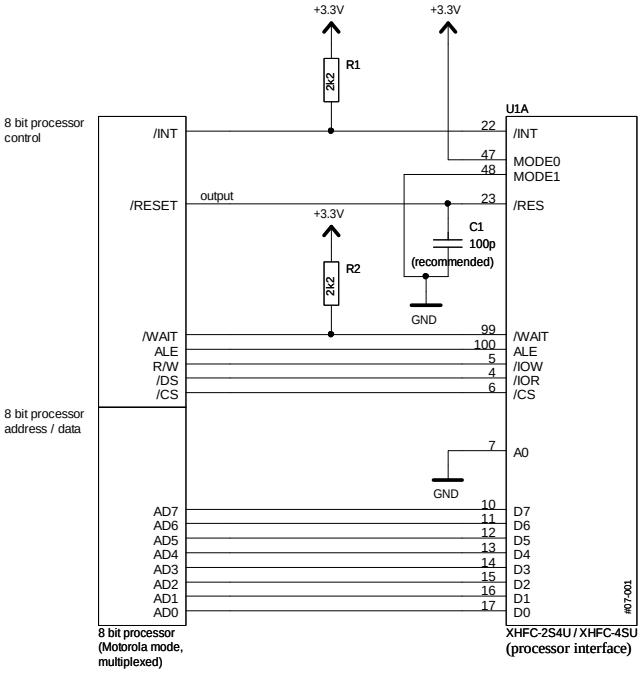


Figure 2.6: *8 bit Motorola processor circuitry example (mode 2m)*

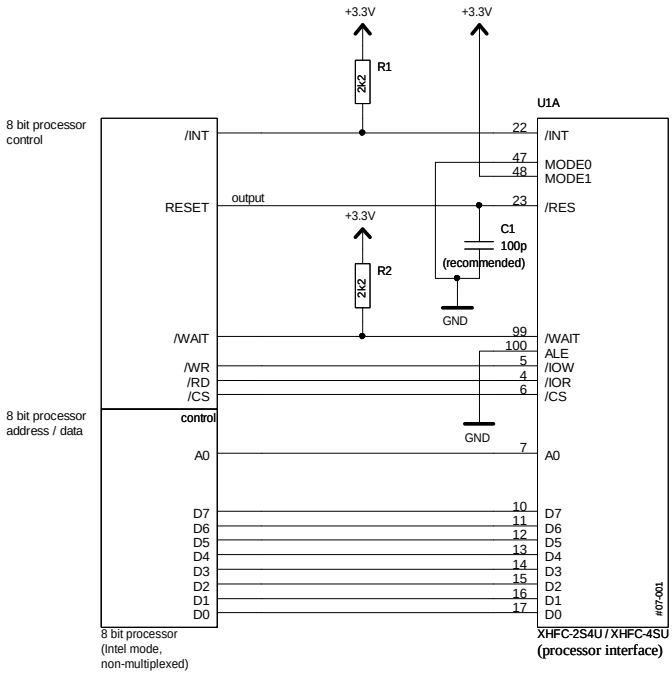


Figure 2.7: *8 bit Intel processor circuitry example (mode 3)*

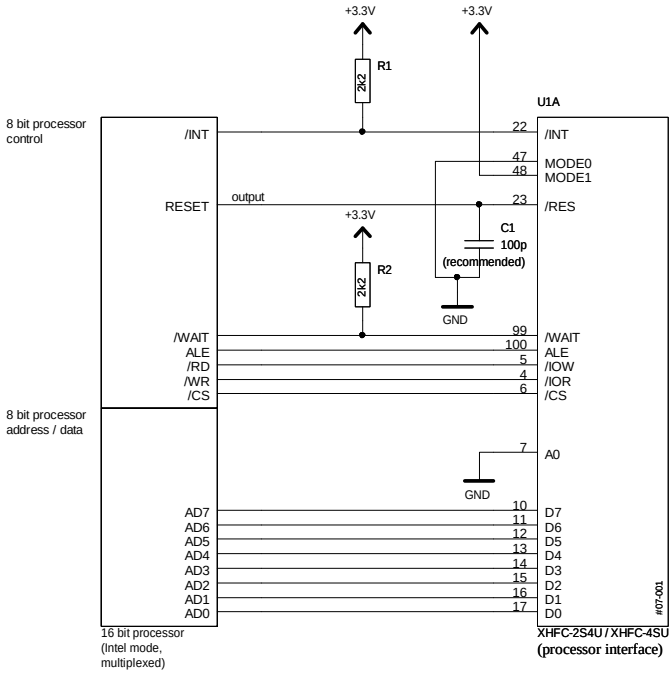


Figure 2.8: *8 bit Intel processor circuitry example (mode 3m)*

2.3 Serial processor interface (SPI)

Table 2.12: *Overview of the SPI interface pins*

Number	Name	Description
22	/INT	Interrupt request pin, active low / high programmable
23	/RES	Reset input pin, active low
47	MODE0	Interface mode pin 0
48	MODE1	Interface mode pin 1
10	SPI_TX	SPI transmit data output
11	SPI_RX	SPI receive data input
4	SPI_CLK	SPI clock input
6	/SPISEL	SPI device select, active low
100	SPI_INV	Invert SPI clock
17	DN0	SPI device number, bit 0
16	DN1	SPI device number, bit 1
15	DN2	SPI device number, bit 2
14	DN3	SPI device number, bit 3

XHFC-2S4U / 4SU has a serial processor interface (SPI) which is compatible with Motorola's SPI. CPUs and MCUs with SPI interface are also available from a variety of semiconductor vendors. The SPI interface has four signal pins as shown in the upper part of Table 2.12. Additional pins are used for SPI clock inversion and SPI device number selection.

SPI interface mode is selected by MODE0 = '0' and MODE1 = '0' (pins 47 and 48). XHFC-2S4U / 4SU support 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-2S4U / 4SU have either a length of 16 bits for single byte accesses or 40 bits for high performance accesses. The first byte is a control byte in both cases, whereas the following bits are either one data byte or four data bytes. Control and data bytes are transmitted with MSB first.

2.3.1 SPI control byte

Tables 2.13 and 2.14 show an overwiev 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 2.15 summarizes the SPI control commands which are coded in the control byte.

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

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

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

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

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

Control byte					R	A	M/B	Timing
R	A	M/B	0	C3..C0				diagram
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			–

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-2S4U / 4SU 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-2S4U / 4SU receives the control byte on the SPI_RX line and transmits the requested data byte on the SPI_TX line afterwards.

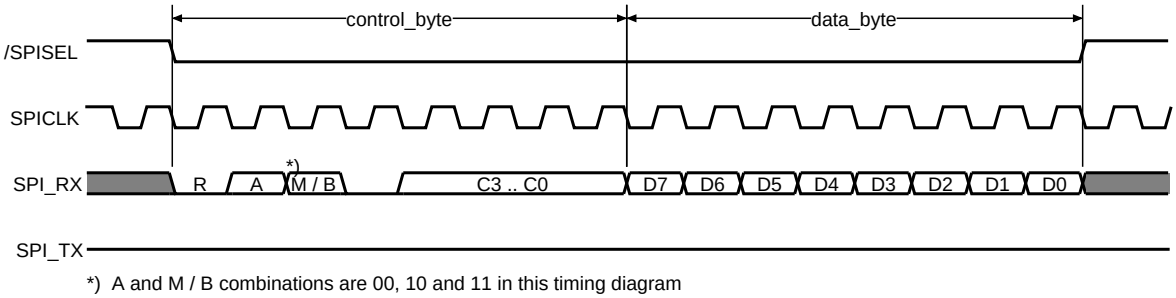


Figure 2.9: *16 bit SPI write transaction (R = '0', one data byte)*

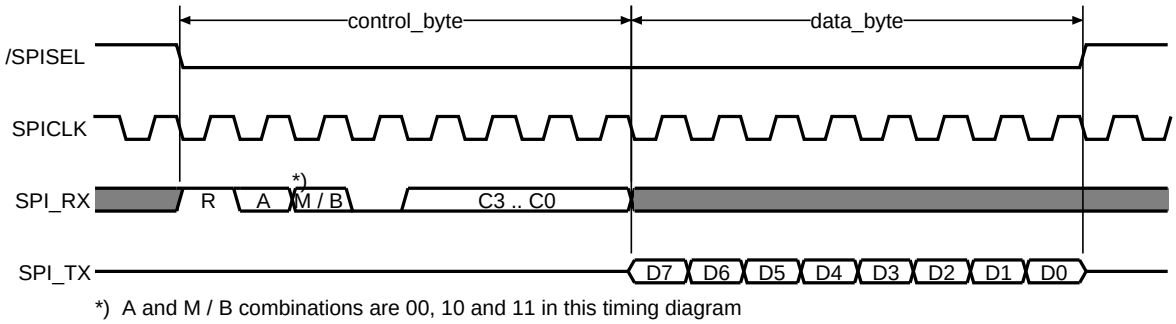


Figure 2.10: *16 bit SPI read transaction (R = '1', one data byte)*

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-2S4U / 4SU. Otherwise master and slave could have different views wether 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 $f_{\mathrm{SPICLK}} = 4\,\mathrm{MHz}$, e.g., a 16 bit transaction takes 4 µs. XHFC-2S4U / 4SU can operate with a maximum SPI clock of 25 MHz which

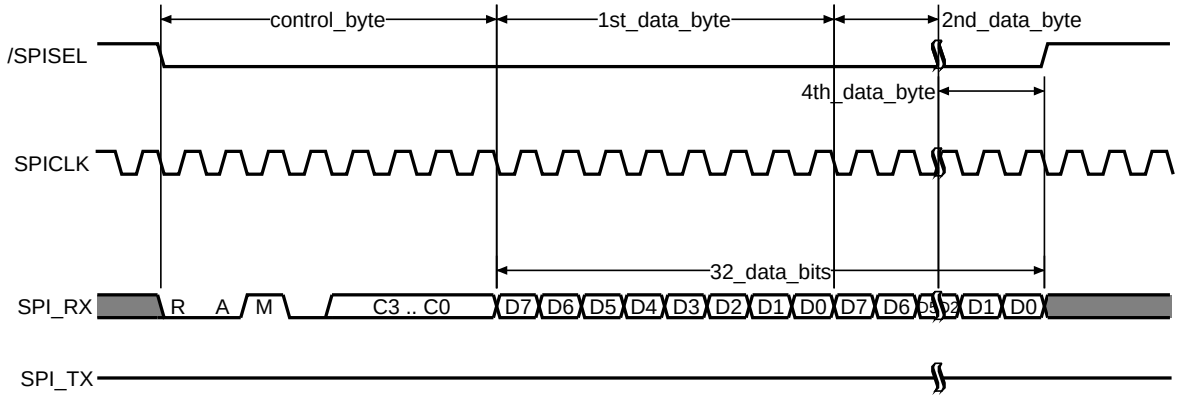


Figure 2.11: *40 bit SPI write transaction (R = '0', four data bytes)*

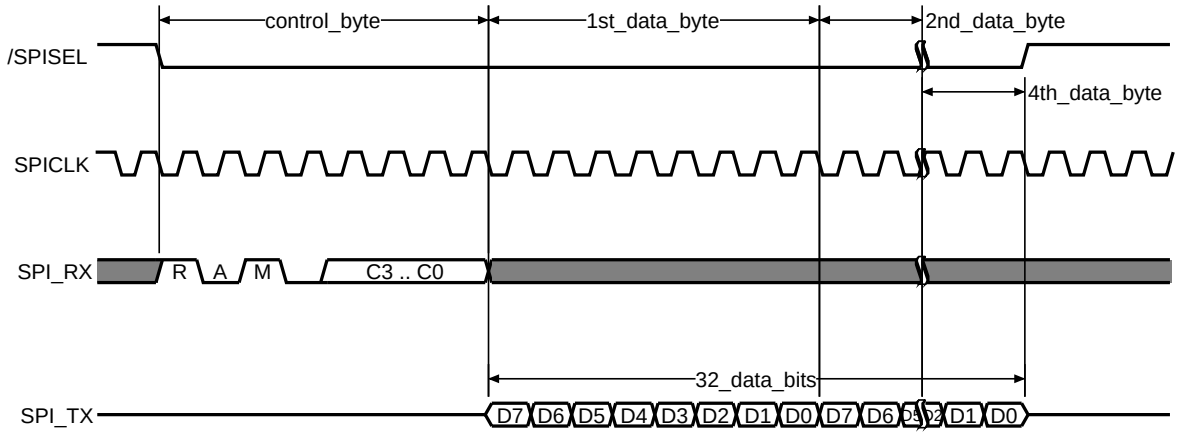


Figure 2.12: *40 bit SPI read transaction (R = '1', four data bytes)*

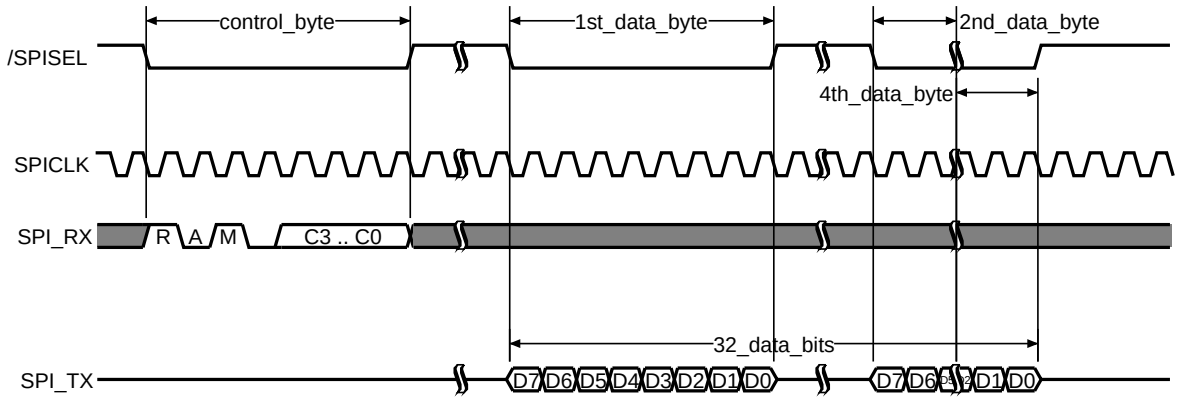


Figure 2.13: *Split 40 bit SPI read transaction (R = '1')*

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

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

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

The XHFC-2S4U / 4SU 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.

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-2S4U / 4SU 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-2S4U / 4SU (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-2S4U / 4SU (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.

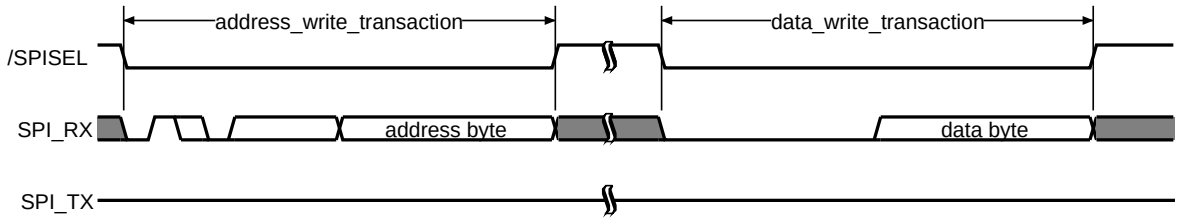


Figure 2.14: *Register write access (transaction sequence)*

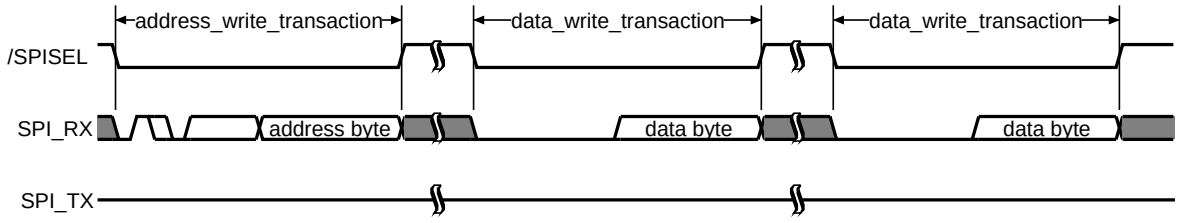


Figure 2.15: *Multiple write accesses to the same register*

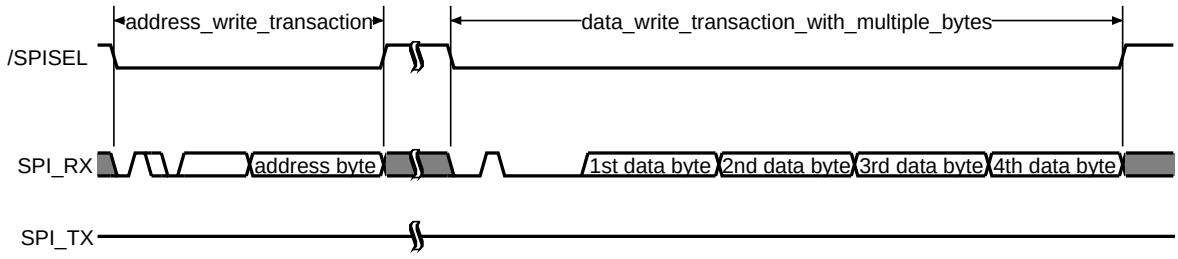


Figure 2.16: *Register write access with a 40 bit transaction*

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

The XHFC-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU 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.

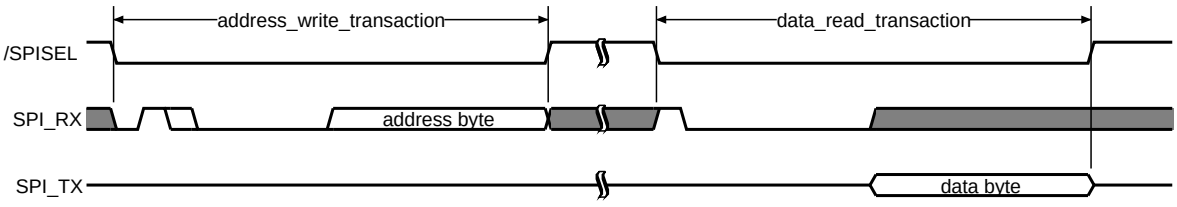


Figure 2.17: *Register read access (transaction sequence)*

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

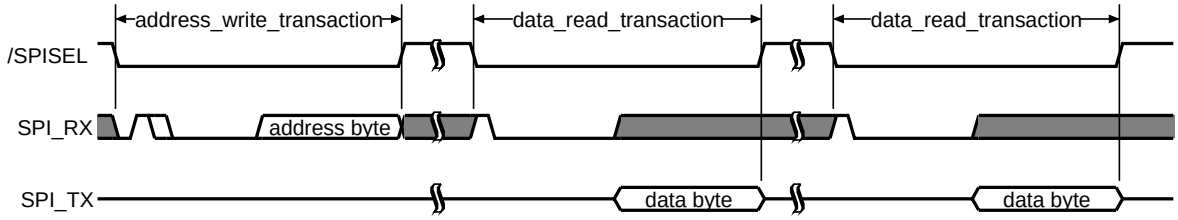


Figure 2.18: *Multiple read accesses to the same register*

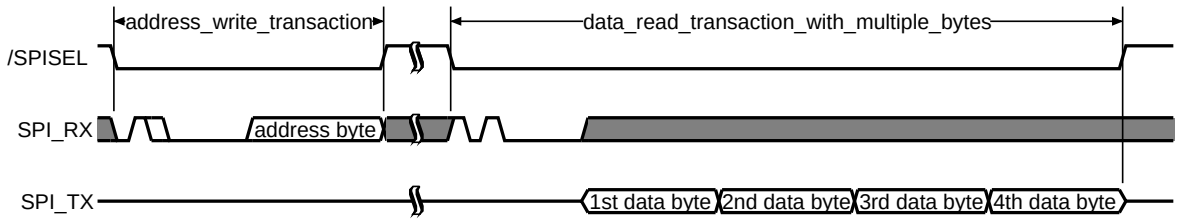


Figure 2.19: *Register read access with a 40 bit transaction*

2.3.6 Register access duration

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

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

Table 2.16: *SPI data rates*

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

2.3.7 Register address read-back capability

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

This address read transaction does not perform a register read access. Thus it can be used even on

those registers that change their contents on a read access, i.e. register contents is changed not until a data read transaction has been executed.

2.3.8 Problems with interrupts during transaction sequences

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

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

This procedure is important to avoid data read or write to an unexpected register address after the transaction sequence has been split between transactions by an interrupt service routine which executes any access to XHFC-2S4U / 4SU. Please note, that transactions are not allowed to be split from the SPI master (see Section 2.3.2).

2.3.9 SPI timing diagrams

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

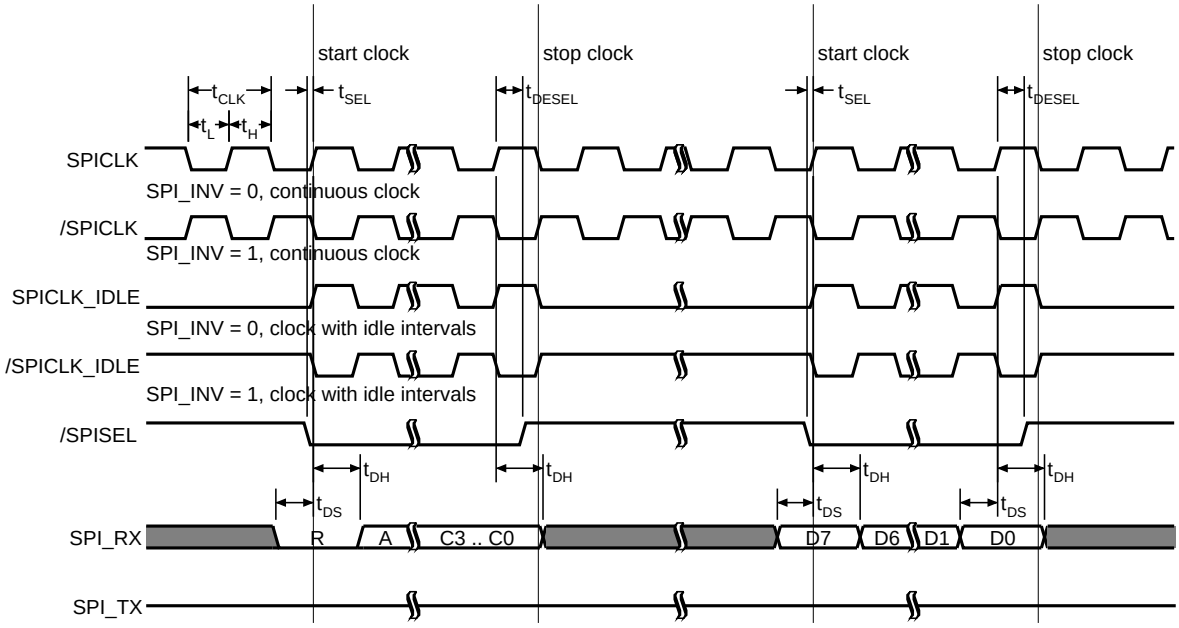


Figure 2.20: *SPI timing diagram for data write transactions*

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

When the SPI access is splitted by the SPI master, the rising edge of /SPISEL is required to be $t_{\mathrm{DESEL}} \geq 2\,\mathrm{ns}$ after the sampling time at the earliest. When clock idle intervals are used, it is necessary

Table 2.17: *Symbols of write access in Figure 2.20*

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

to have at least one clock edge after the last sample edge (marked with ‘stop clock’ in Figure 2.20). This is important to terminate the access internally.

The timing diagram for data read transactions (data from slave to master) is shown in Figure 2.21. The same different variations of the SPI clock which are shown in Figure 2.20 are valid for data read transactions but only continiuos clock with SPI_INV = ’0’ is shown here.

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

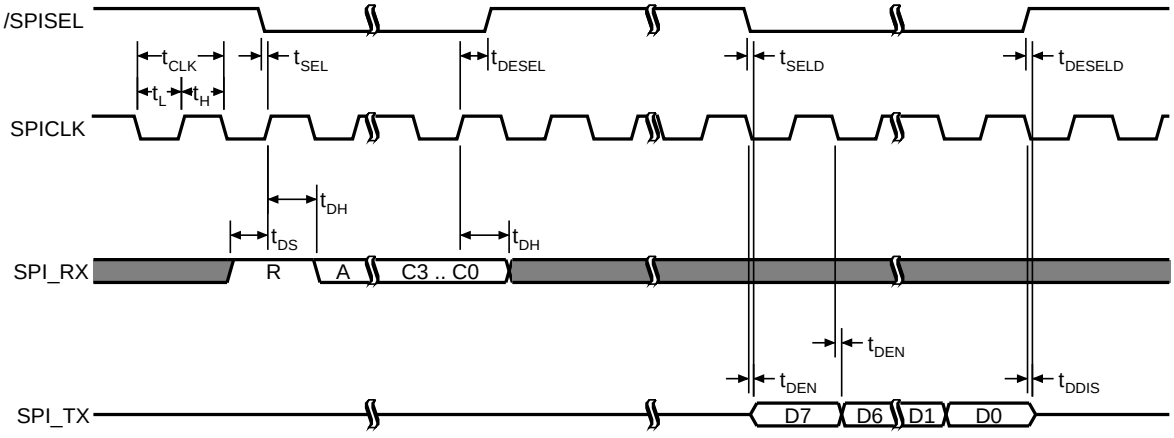


Figure 2.21: *SPI timing diagram for data read transactions (see Figure 2.20 for additional clock signals /SPICLK, SPICLK_IDLE and /SPICLK_IDLE)*

Table 2.18: *Symbols of read access in Figure 2.21*

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

2.3.10 SPI connection circuitry

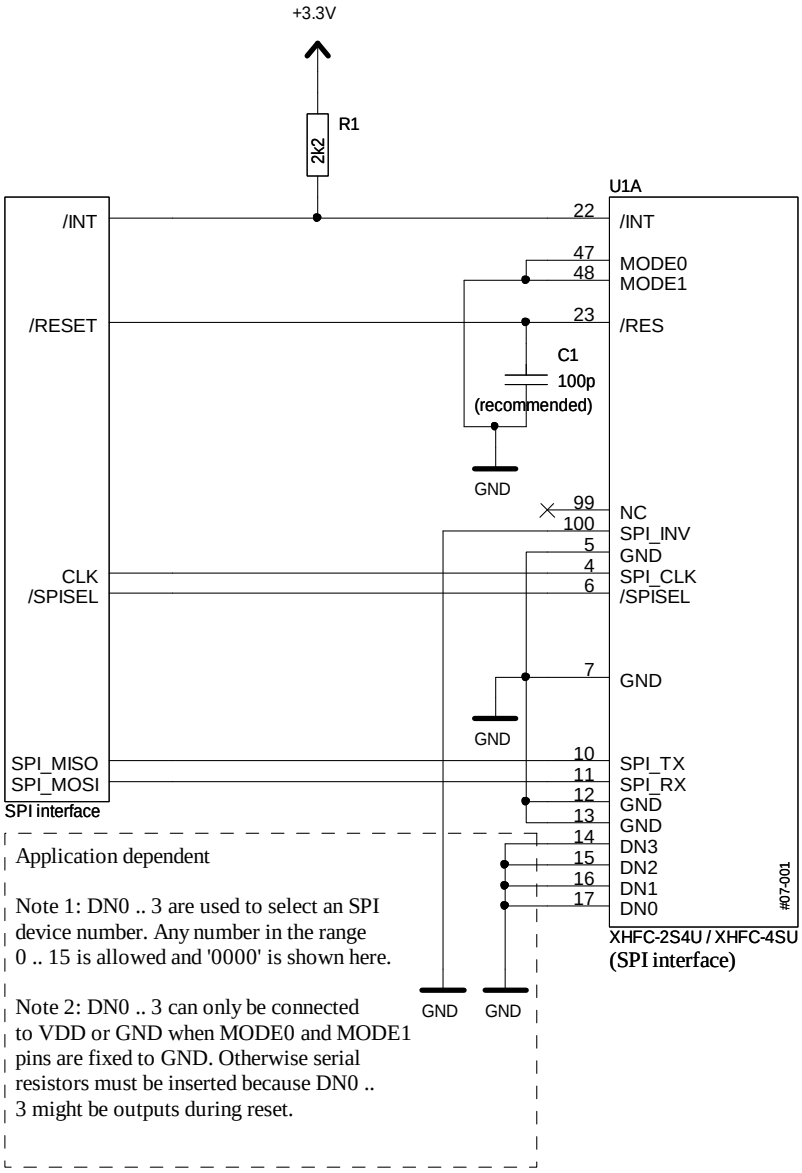


Figure 2.22: *SPI connection circuitry*

2.4 Auto-EEPROM mode

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

2.5 Register description

2.5.1 Write only registers

R_CTRL	(w)	(Reset group: H)	0x01

Common control register

Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	**V_FIFO_LPRIO**	**FIFO access priority for host accesses** '0' = normal priority '1' = low priority
2	0	(reserved)	Must be '0'.
3	0	**V_NT_SYNC**	**Synchronization source for NT mode** The transmit data path of the Universal ISDN Ports can be synchronized to different signals. '0' = F0I is used as NT synchronization source '1' = F1_7 is used as NT synchronization source (see register R_SL_SEL7 for time slot selection) **Note:** This bit selects the synchronization source for all Universal ISDN Ports in NT / LT mode together. It is ignored in TE mode.
4	0	(reserved)	Must be '0'.
5	0	**V_OSC_OFF**	**Disable oscillator** '0' = normal operation '1' = clock oscillator is switched off This bit is reset at every write access to XHFC-2S4U / 4SU or with a wake-up signal on pin WAKEUP . Any chip access is valid not before the oscillator frequency is stable again.
7..6	0	**V_SU_CLK**	**Line interface clock selection** The line interface clock f_{SU} is derived from the system clock f_{SYS} '00' = f_{SYS} / 2 '01' = f_{SYS} / 4 '10' = f_{SYS} (normally unused) '11' = f_{SYS} / 8 (normally unused) f_{SU} must be 12.288 MHz.

R_RAM_ADDR	(w)	(Reset group: H, 0)	0x08

Address pointer, lower part

Lower address byte for SRAM access.

Bits	Reset value	Name	Description
7 . . 0	0x00	**V_RAM_ADDR0**	**Address bits 7 . . 0**

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_RAM_CTRL	(w)	(Reset group: H, 0)	0x09

SRAM access control register

High address bits and control bits for SRAM access.

Bits	Reset value	Name	Description
3 . . 0	0	**V_RAM_ADDR1**	**Address bits 11 . . 8**
5 . . 4	0	(reserved)	Must be ’00’.
6	0	**V_ADDR_RES**	**Address reset** ’0’ = normal operation ’1’ = address bits 0 . . 11 are set to zero This bit is automatically cleared.
7	0	**V_ADDR_INC**	**Address increment** ’0’ = no address increment ’1’ = automatically increment of the address after every write or read on register R_RAM_DATA

2.5.2 Read only registers

R_RAM_USE	(r)	(Reset group: –)	0x15

SRAM duty factor

Usage of SRAM access bandwidth by the internal data processor.

Bits	Reset value	Name	Description
7..0		V_SRAM_USE	**Relative duty factor** 0x00 = 0% bandwidth used 0x7C = 100% bandwidth used

R_CHIP_ID	(r)	(Reset group: H)	0x16

Chip identification register

Bits	Reset value	Name	Description
7..0		V_CHIP_ID	**Chip identification code** '01100010' (0x62) means XHFC-2S4U, '01100011' (0x63) means XHFC-4SU. The LSB should be masked out for compatibility reasons. **Note:** XHFC-2S4U and XHFC-4SU are pin compatible and software compatible. Thus, they can be exchanged on the same hardware platform without software adaption when LSB is masked out.

(See Section 9.4.4.1 on page 303 to avoid unexpected line interface interrupts when using XHFC-2S4U.)

R_CHIP_RV	(r)	(Reset group: H)	0x1F

XHFC-2S4U / 4SU revision

Bits	Reset value	Name	Description
3..0	0	V_CHIP_RV	**Chip revision 0**
7..4		(reserved)	

R_INT_DATA	(r)	(Reset group: –)	0x88

Internal data register

This register can be read to access data with Read* method.

Bits	Reset value	Name	Description
7..0		V_INT_DATA	Internal data buffer

(See Section 2.2.3.2 on page 47 for details on Read* access.)

2.5.3 Read / write register

R_RAM_DATA	(r*/w)	(Reset group: –)	0xC0

SRAM data access

Direct access to the internal SRAM

Bits	Reset value	Name	Description
7..0		V_RAM_DATA	**SRAM data access** The address must be written into registers R_RAM_ADDR and R_RAM_CTRL in advance.

(See Section 2.2.3.2 on page 47 for details on Read* access.)

Chapter 3

XHFC-2S4U / 4SU data flow

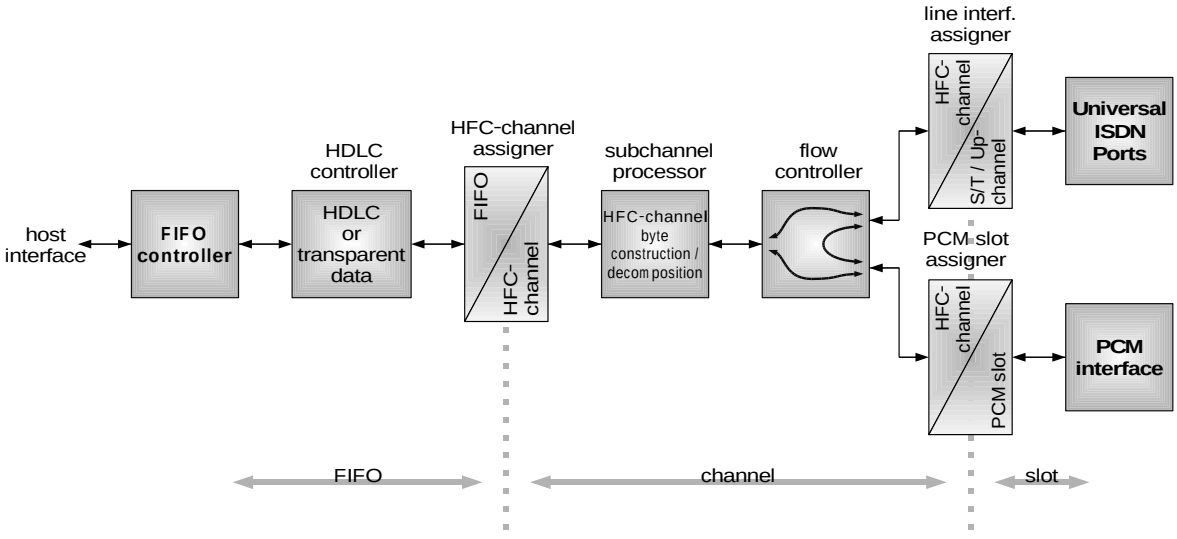


Figure 3.1: *Data flow block diagram*

3.1 Data flow concept

3.1.1 Overview

XHFC-2S4U / 4SU 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 [1].

The fundamental features of the XHFC-2S4U / 4SU data flow are as follows:

- programmable interconnection capability between FIFOs, PCM time slots and channels of the Universal ISDN Ports
- XHFC-2S4U: 2 Universal ISDN Ports (combined ST / U_p interfaces) and 2 additional U_p interfaces, XHFC-4SU: 4 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-2S4U / 4SU 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.

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

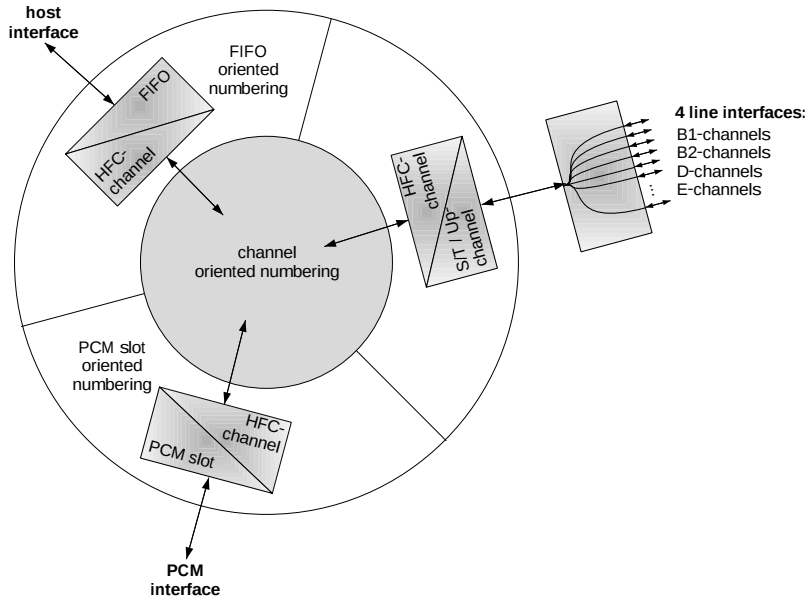


Figure 3.2: *Areas of FIFO oriented, HFC-channel oriented and PCM time slot oriented numbering*

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 µs 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 µs cycle. In the reverse case, a received byte which is stored in a switching buffer is copied to the FIFO during the next 125 µs 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 µs cycle, then copied into the transmit buffer during the second 125 µs cycle and finally transmitted from the interface during the third 125 µs 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 µs 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 130). 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-2S4U / 4SU 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.

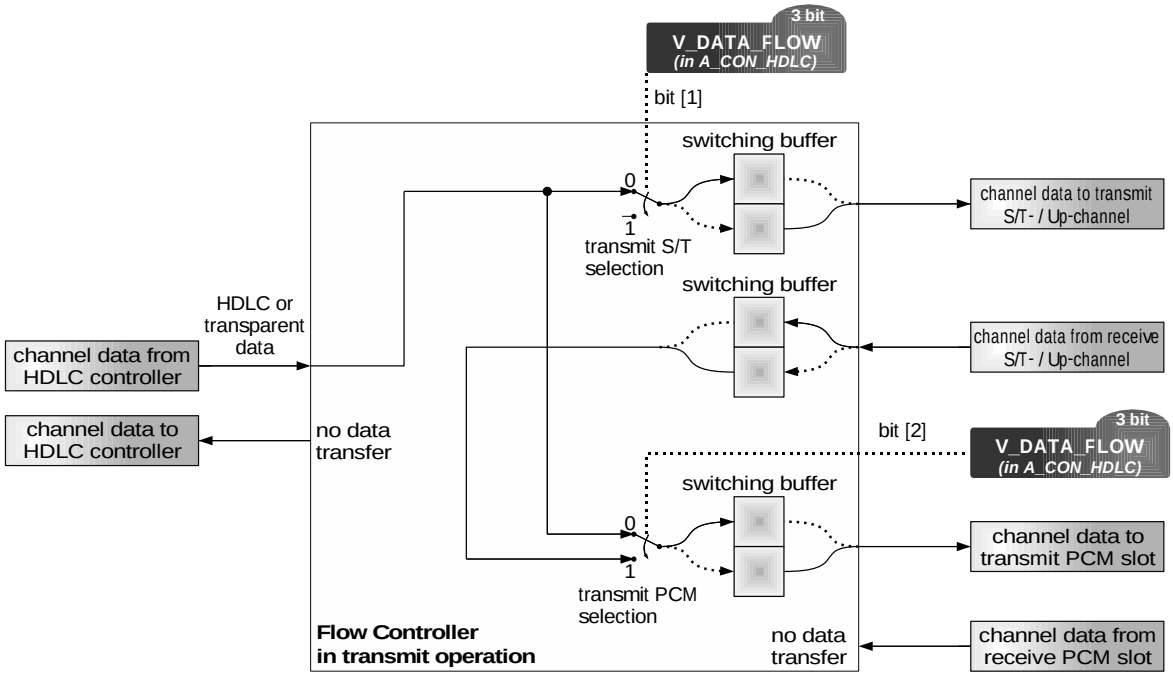


Figure 3.3: *The flow controller in transmit operation*

- 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.

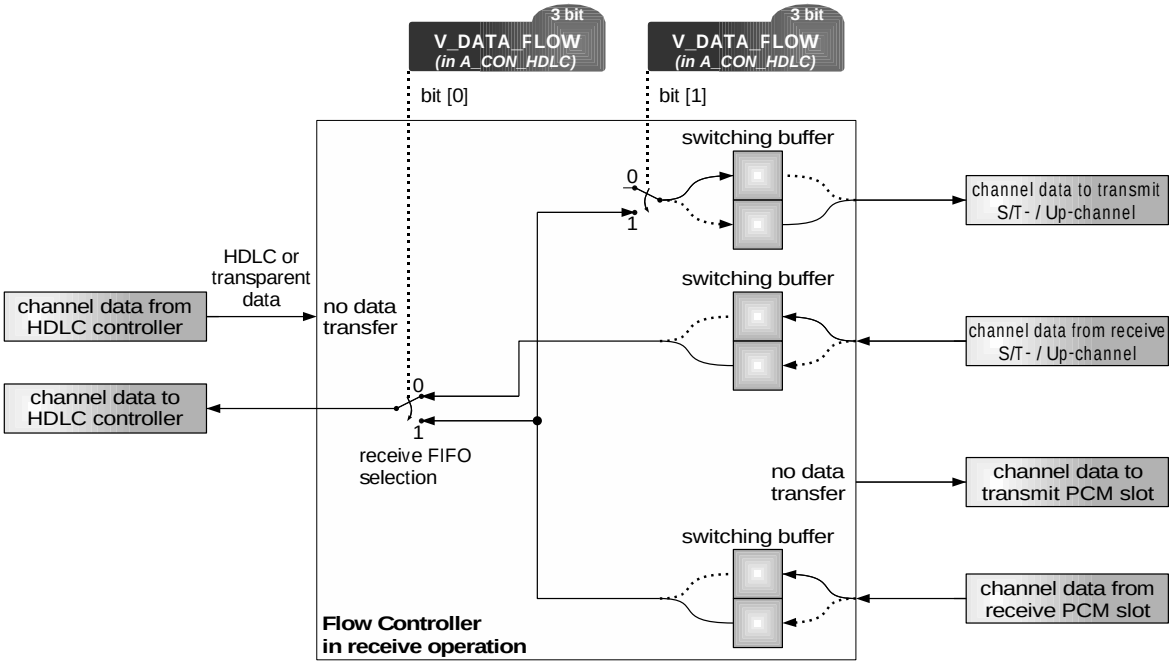


Figure 3.4: *The flow controller in receive FIFO operation*

- 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 [2] 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.

[2] 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_p	FIFO ← ST / U_p	
'001'	FIFO → ST / U_p	FIFO → PCM	FIFO ← PCM	
'010'	FIFO → PCM		FIFO ← ST / U_p	ST / U_p ← PCM
'011'		FIFO → PCM	FIFO ← PCM	ST / U_p ← PCM
'100'	ST / U_p → PCM	FIFO → ST / U_p	FIFO ← ST / U_p	
'101'	FIFO → ST / U_p	ST / U_p → PCM	FIFO ← PCM	
'110'		ST / U_p → PCM	ST / U_p ← PCM	FIFO ← ST / U_p
'111'		ST / U_p → PCM	ST / U_p ← PCM	FIFO ← PCM

Table 3.2: *V_DATA_FLOW programming values for bidirectional connections*

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

3.3 Assigners

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

3.3.1 HFC-channel assigner

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

3.3.2 PCM slot assigner

The PCM slot assigner can connect each PCM time slot to an arbitrary HFC-channel. Therefore, for a selected time slot [3] the connected HFC-channel number and data direction must be written into register A_SL_CFG[SLOT] as follows:

Register setup:

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

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

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

3.3.3 Line interface assigner

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

If ST / U_p-channels are coded as

$$\begin{aligned} \text{B1-channel} &= 0 \\ \text{B2-channel} &= 1 \\ \text{D-channel} &= 2 \\ \text{E-channel} &= 3 \end{aligned}$$

it is possible to calculate

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

For a given HFC-channel number the belonging ST / U_p-channel is calculated with [4]

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

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

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

Table 3.3: *Line interface assigner*

HFC-channel		ST / U_p-channel			HFC-channel		ST / U_p-channel		
number	direction	interface	channel	direction	number	direction	interface	channel	direction
[0,TX]		#0	B1	TX	[8,TX]		#2	B1	TX
[0,RX]		#0	B1	RX	[8,RX]		#2	B1	RX
[1,TX]		#0	B2	TX	[9,TX]		#2	B2	TX
[1,RX]		#0	B2	RX	[9,RX]		#2	B2	RX
[2,TX]		#0	D	TX	[10,TX]		#2	D	TX
[2,RX]		#0	D	RX	[10,RX]		#2	D	RX
[3,TX]		#0	BAC / S	TX	[11,TX]		#2	BAC / S	TX
[3,RX]		#0	E	RX	[11,RX]		#2	E	RX
[4,TX]		#1	B1	TX	[12,TX]		#3	B1	TX
[4,RX]		#1	B1	RX	[12,RX]		#3	B1	RX
[5,TX]		#1	B2	TX	[13,TX]		#3	B2	TX
[5,RX]		#1	B2	RX	[13,RX]		#3	B2	RX
[6,TX]		#1	D	TX	[14,TX]		#3	D	TX
[6,RX]		#1	D	RX	[14,RX]		#3	D	RX
[7,TX]		#1	BAC / S	TX	[15,TX]		#3	BAC / S	TX
[7,RX]		#1	E	RX	[15,RX]		#3	E	RX

In both cases the equivalence

$$\text{HFC-channel direction} \;=\; \text{ST / U}_\text{p}\text{-channel direction}$$

is valid.

3.3.4 Assigner summary

The three different assigner types of XHFC-2S4U / 4SU are shown in Figure 3.5. Assigner programming is always handled with array registers. This can be a FIFO array register or a PCM slot array register.

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

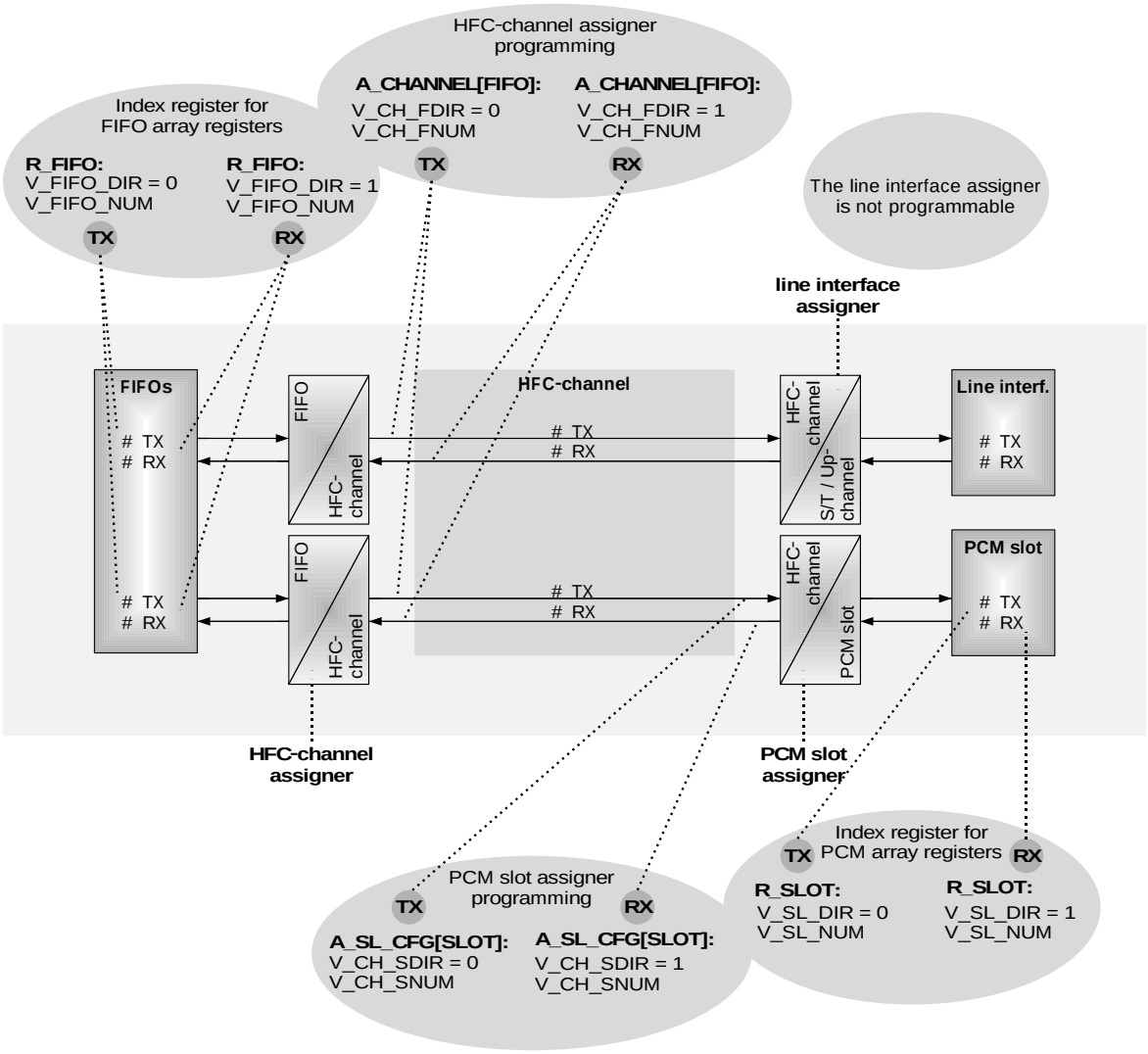


Figure 3.5: *Overview of the assigner programming*

3.4 Data flow modes

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

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

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

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

3.4.1 Simple Mode (SM)

3.4.1.1 Mode description

In *Simple Mode* (SM) only one-to-one connections are possible. That means one FIFO, one ST / U_p-channel or one PCM time slot can be connected to each other. The number of connections is limited by the number of FIFOs. It is possible to establish as many connections as there are FIFOs[5]. 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 usefull 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.

[5] 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 name	Array register address	I/O mode	Index register in SM	Index register in CSM	Index register in FSM	Index meaning in SM	Index meaning in CSM	Index meaning in 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 µs.

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 110 for details concerning the subchannel processor.

3.4.1.3 Example for SM

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

> **Please note !**
>
> All settings in Figure 3.6 are configured in bidirectional data paths due to typical applications of XHFC-2S4U / 4SU. 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_p**

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

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

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

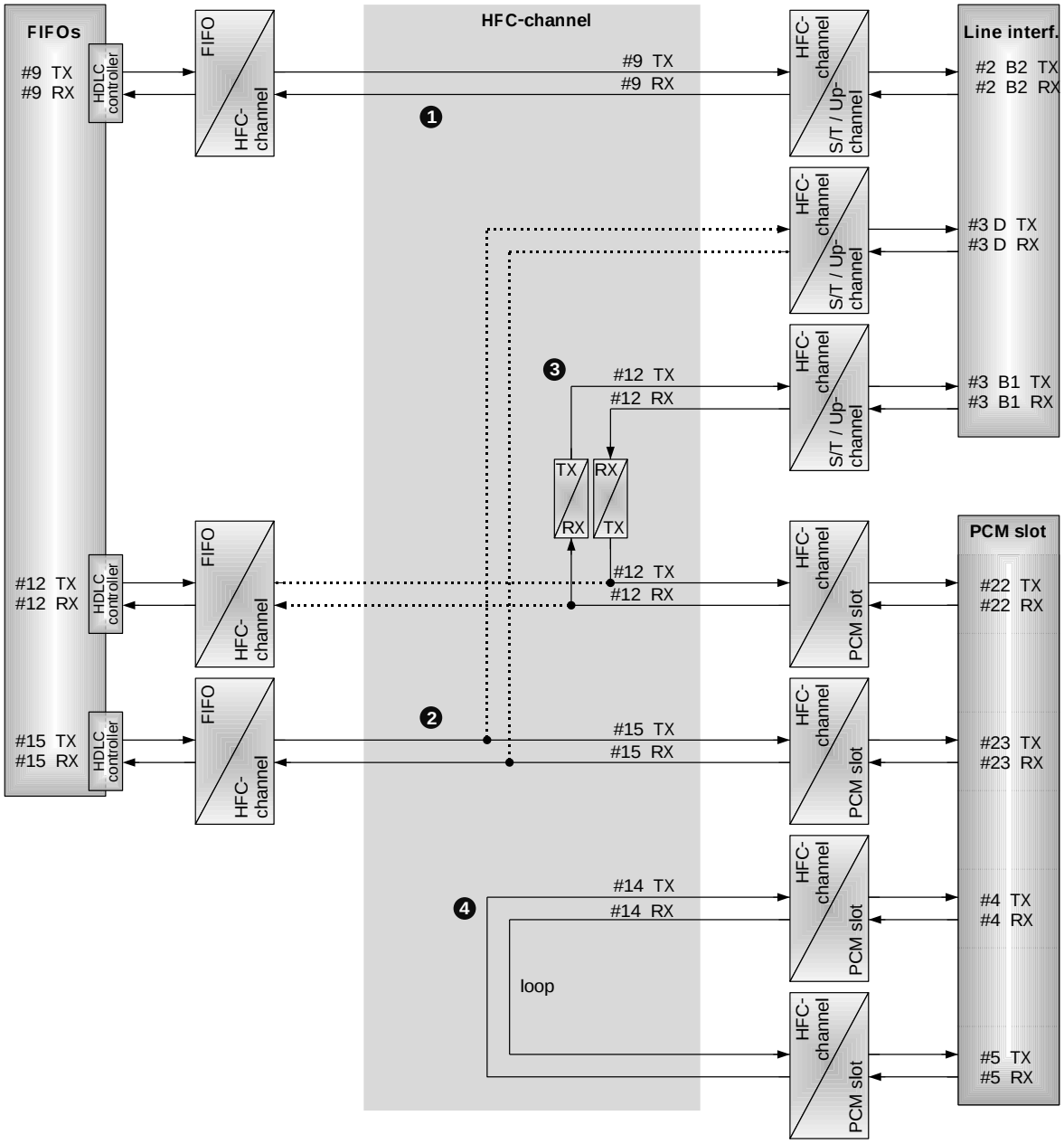


Figure 3.6: *SM example*

Register setup:				(SM ❶ TX)
R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	9	(FIFO #9)
	: V_REV	=	0	(normal bit order)
A_CON_HDLC[9,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	0	(HDLC mode)
	: V_FIFO_IRQ	=	7	(enable FIFO)
	: V_DATA_FLOW	=	'000'	(FIFO → ST / U_p , FIFO → PCM)

Register setup:				(SM ❶ RX)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	9	(FIFO #9)
	: V_REV	=	0	(normal bit order)
A_CON_HDLC[9,RX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	0	(HDLC mode)
	: V_FIFO_IRQ	=	7	(enable FIFO)
	: V_DATA_FLOW	=	'000'	(FIFO ← ST / U_p)

❷ FIFO-to-PCM

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

As the line interface assigner links the HFC-channels to the ST / U_p -channels, every used HFC-channel blocks the connected ST / U_p-channel . In this example the E-channel of line interface #3 is blocked in both data directions. As this channel does only exist in S/T interface mode, there are no resources blocked if the line interface operates in U_p mode. As this interface does not exist for XHFC-2SU, in this case the HFC-channels are unassigned and do not block any ST / U_p resource.

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

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

❸ PCM-to-ST / U_p

A direct PCM-to-ST / U_p coupling is shown in the third connection set. The array registers of FIFO[12,TX] and FIFO[12,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 $\neq 0$ in register A_CON_HDLC[FIFO].

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

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

Register setup:				(SM ❸ 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	= 1		(transparent mode)
	: V_FIFO_IRQ	= 7		(enable data transmission)
	: V_DATA_FLOW	= '110'		(ST / U_p → PCM)
R_SLOT	: V_SL_DIR	= 0		(transmit slot)
	: V_SL_NUM	= 22		(slot #22)
A_SL_CFG[22,TX]	: V_CH_SDIR	= 0		(transmit HFC-channel)
	: V_CH_SNUM	= 12		(HFC-channel #12)
	: V_ROUT	= '10'		(data to pin STIO1)

Register setup:				(SM ❸ 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	= 1		(transparent mode)
	: V_FIFO_IRQ	= 7		(enable data transmission)
	: V_DATA_FLOW	= '110'		(FIFO ← ST / U_p , ST / U_p ← PCM)
R_SLOT	: V_SL_DIR	= 1		(receive slot)
	: V_SL_NUM	= 22		(slot #22)
A_SL_CFG[22,RX]	: V_CH_SDIR	= 1		(receive HFC-channel)
	: V_CH_SNUM	= 12		(HFC-channel #12)
	: V_ROUT	= '10'		(data from pin STIO2)

❹ 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.

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

Register setup:				(SM ❹ 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 110 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_p , ❷ FIFO-to-PCM and ❸ PCM-to-ST / U_p). 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.

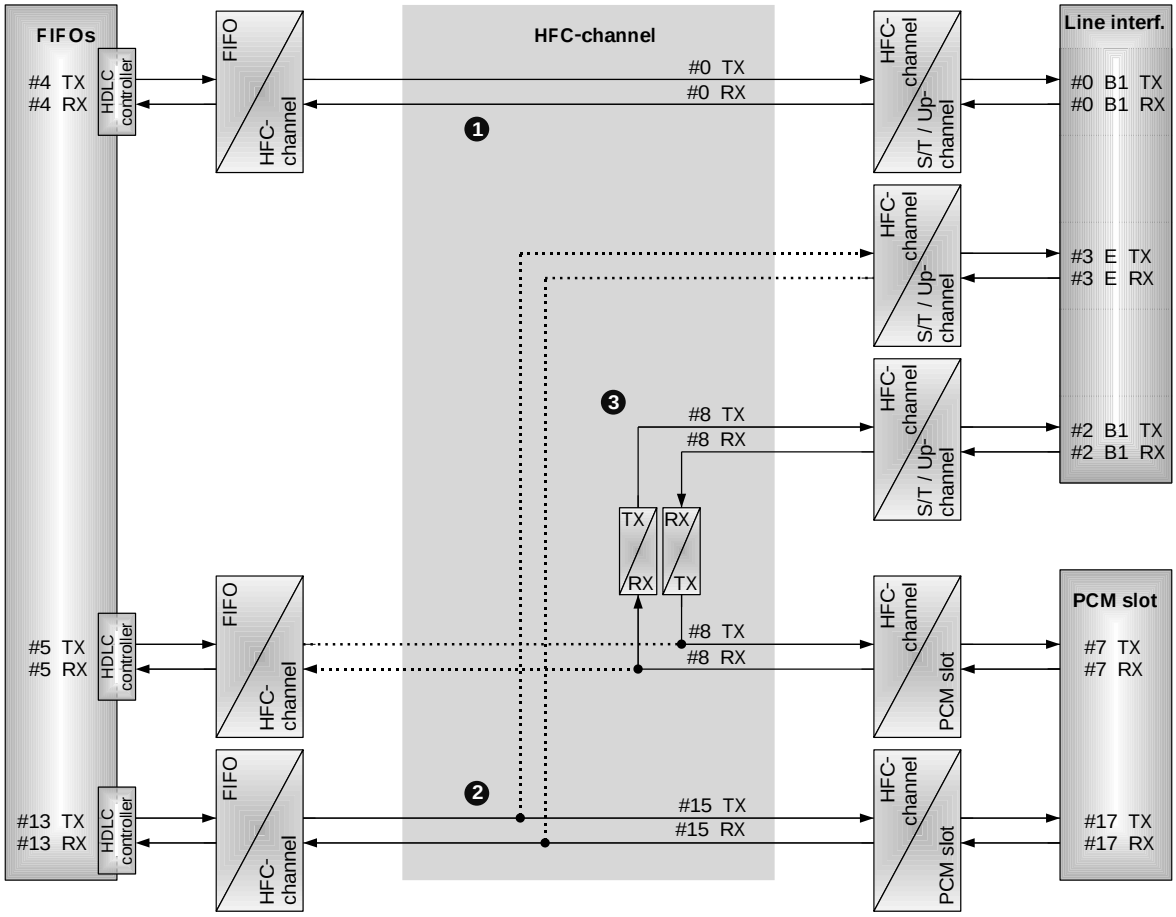


Figure 3.8: *CSM example*

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_p**

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

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

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

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

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

❷ FIFO-to-PCM

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

In this example, the selected ST / U_p-channel exists only in S/T interface mode. So there are no resources blocked if this interface operates in U_p mode.

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

Register setup:				(CSM ❷ 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)

❸ PCM-to-ST / U_p

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

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

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

PCM side of the TX-RX-exchanger, typically. [6]

Register setup:				(CSM ❸ TX)
R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	5	(FIFO #5)
	: V_REV	=	0	(normal bit order)
A_CON_HDLC[5,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	1	(transparent mode)
	: V_FIFO_IRQ	=	7	(enable data transmission)
	: V_DATA_FLOW	=	'110'	($ST/U_p \rightarrow$ PCM)
A_CHANNEL[5,TX]	: V_CH_FDIR	=	0	(transmit HFC-channel)
	: V_CH_FNUM	=	8	(HFC-channel #8)
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	=	8	(HFC-channel #8)
	: V_ROUT	=	'10'	(data to pin STIO1)

Register setup:				(CSM ❸ 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 $\leftarrow ST/U_p$, $ST/U_p \leftarrow$ PCM)
A_CHANNEL[5,RX]	: V_CH_FDIR	=	1	(receive HFC-channel)
	: V_CH_FNUM	=	8	(HFC-channel #8)
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	=	8	(HFC-channel #8)
	: V_ROUT	=	'10'	(data from pin STIO2)

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

Rule

In *Channel Select Mode*

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

3.4.3 FIFO Sequence Mode (FSM)

3.4.3.1 Mode description

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

Figure 3.9: *HFC-channel assigner in FSM*

FIFO Sequence Mode is selected with V_DF_MD = '11' in register R_FIFO_MD. This data flow mode selects the multi-register R_FSM_IDX at the address 0x0F for some FIFO array registers (see Table 3.4 on page 88).

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 $n \cdot 8\,\text{kByte/s}$.

The complete list is processed every 125 µs 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.

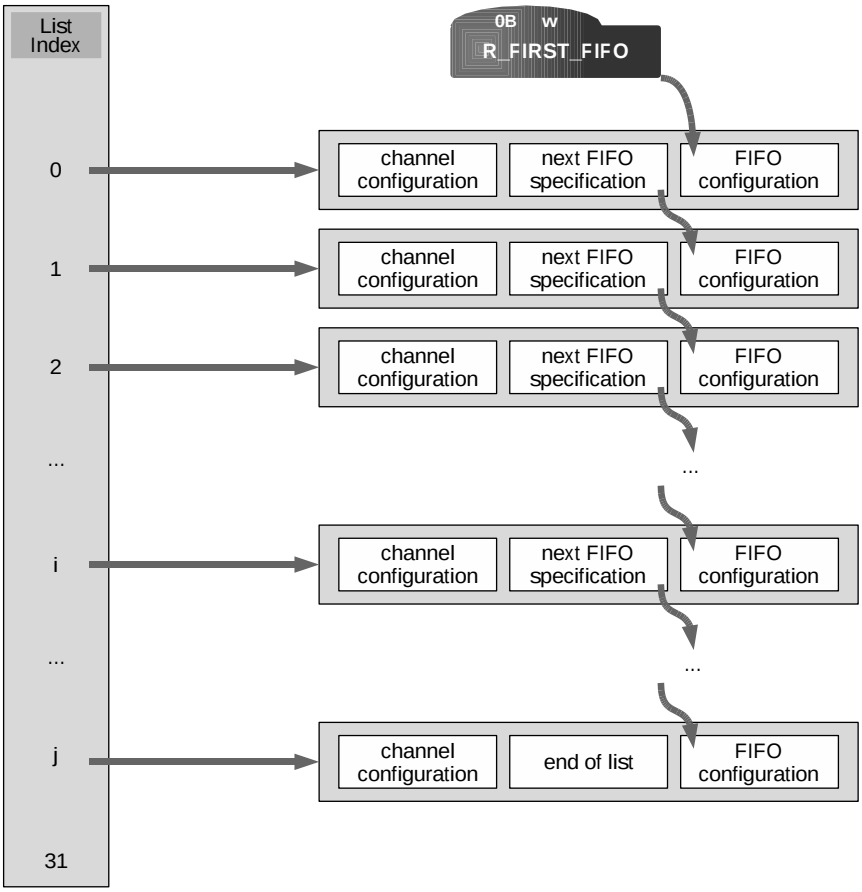


Figure 3.10: *FSM list processing*

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 88). All other FIFO array registers remain indexed by R_FIFO.

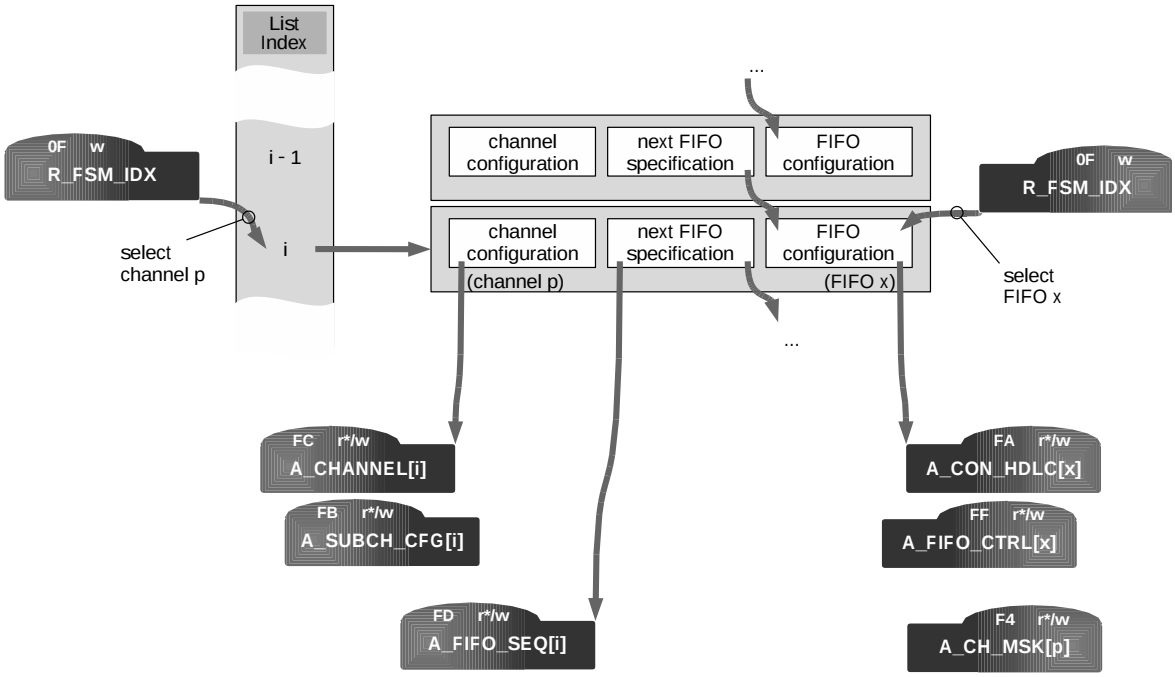


Figure 3.11: *FSM list programming*

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

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

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

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

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

in the usual way. Please note, that register A_CH_MSK requires the addressed HFC-channel to be specified in register R_FIFO (see remark on page 111).

3.4.3.4 Example for FSM

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

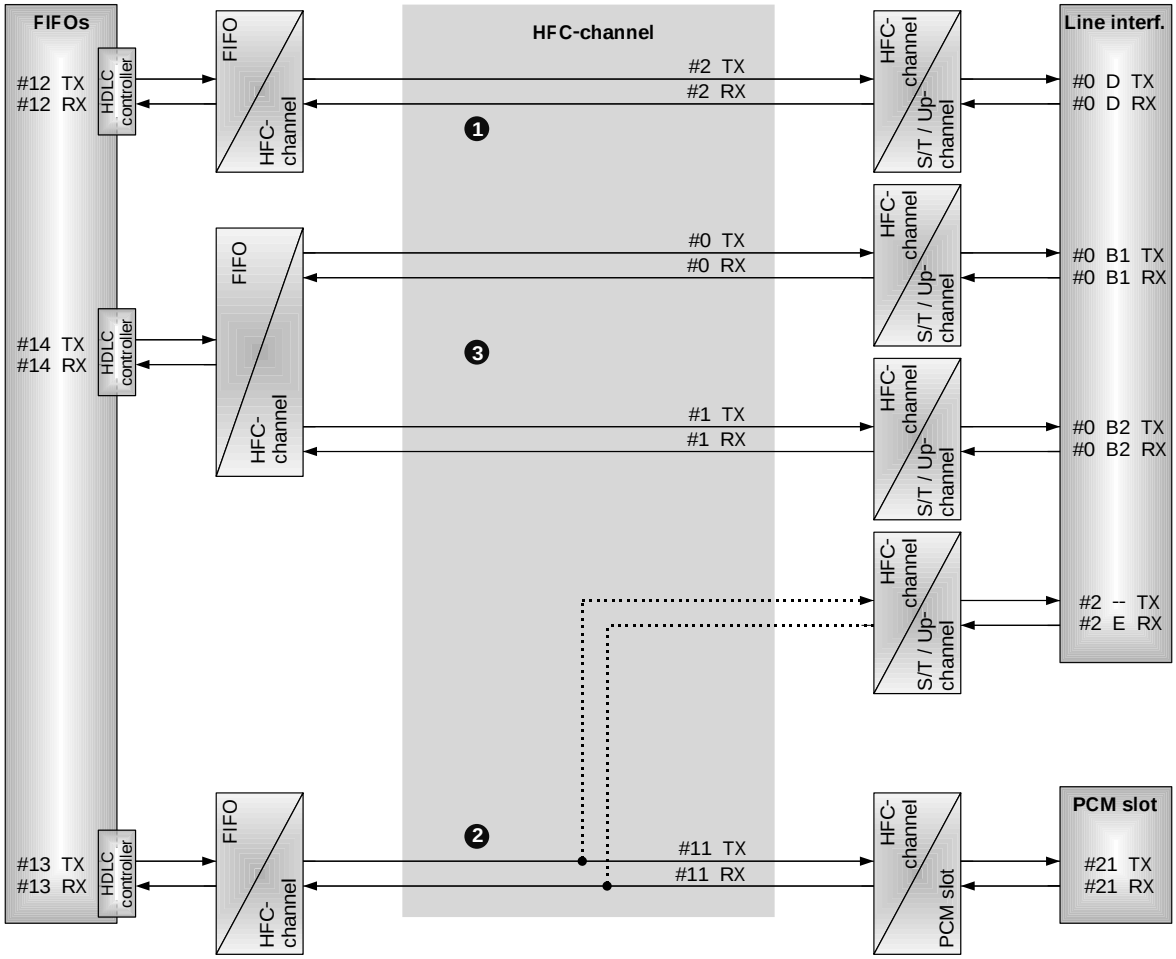


Figure 3.12: *FSM example*

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

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] beeing 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_p interf. #0, D TX
❶	1	FIFO[12,RX] ← ST / U_p 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_p interf. #0, B1 TX
❸	5	FIFO[14,RX] ← ST / U_p interf. #0, B1 RX
❸	6	FIFO[14,TX] → ST / U_p interf. #0, B2 TX
❸	7	FIFO[14,RX] ← ST / U_p interf. #0, B2 RX

❶ FIFO-to-ST / U_p

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

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

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

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

❷ FIFO-to-PCM

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

Two ST / U_p-channels are blocked. But ST / U_p-channel resources are saved because the HFC-channels are assigned to an E-channel which is normally not used and an unused transmit channel.

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

❸ FIFO to multiple ST / U_p-channels

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

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

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

Register setup:				(FSM ❸ 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 subchannel 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_p 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.

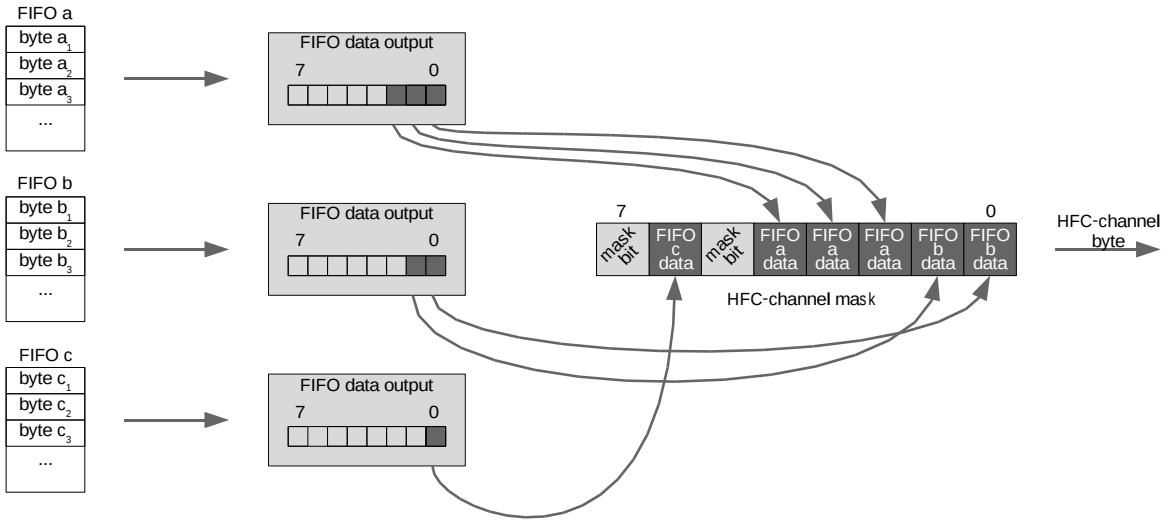


Figure 3.13: *General structure of the subchannel processor shown with an example of three connected FIFOs*

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

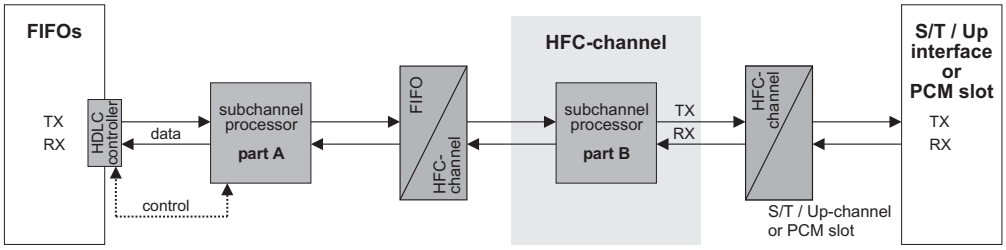


Figure 3.14: *Location of the subchannel parts A and B in the data flow diagram*

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 µs 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 µs 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 µs 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.

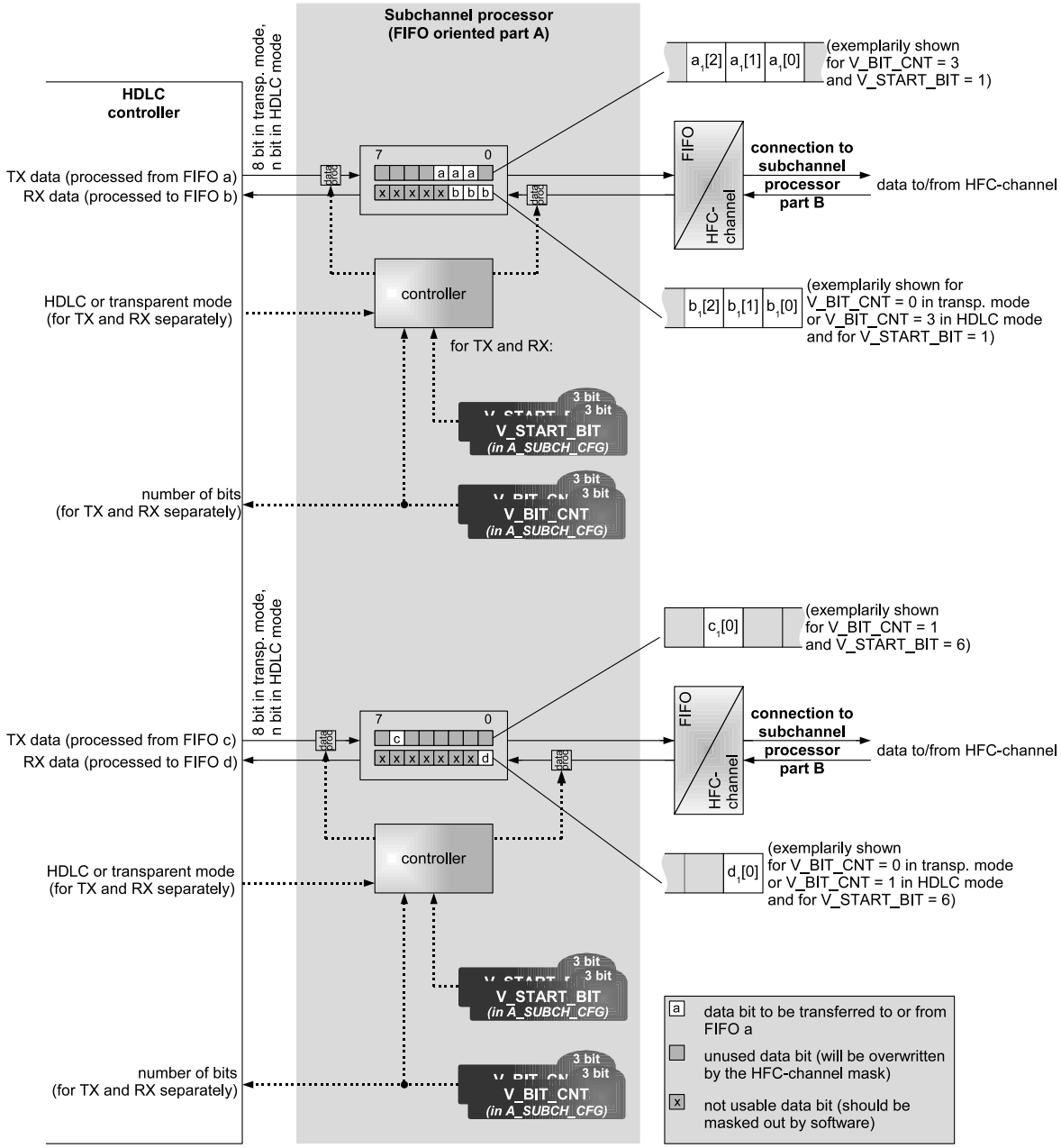


Figure 3.15: *Part A of the subchannel processor*

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

$$DR_{FIFO} = \begin{cases} \text{V\_BIT\_CNT}\,\text{kBit/s} & : \quad \text{V\_BIT\_CNT} > 0 \\ 8\,\text{kBit/s} & : \quad \text{V\_BIT\_CNT} = 0 \end{cases}$$

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

$$DR_{FIFO} = \begin{cases} \text{V\_BIT\_CNT}\,\text{kBit/s} & : \quad \text{V\_BIT\_CNT} > 0 \\ 8\,\text{kBit/s} & : \quad \text{V\_BIT\_CNT} = 0 \end{cases}$$

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_p interface). On the other side, to / from every connected HFC-channel assigner one byte is transferred in both transmit and receive directions. Figure 3.16 shows the block diagram of this module.

3.5.4.1 FIFO transmit operation in SM

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

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

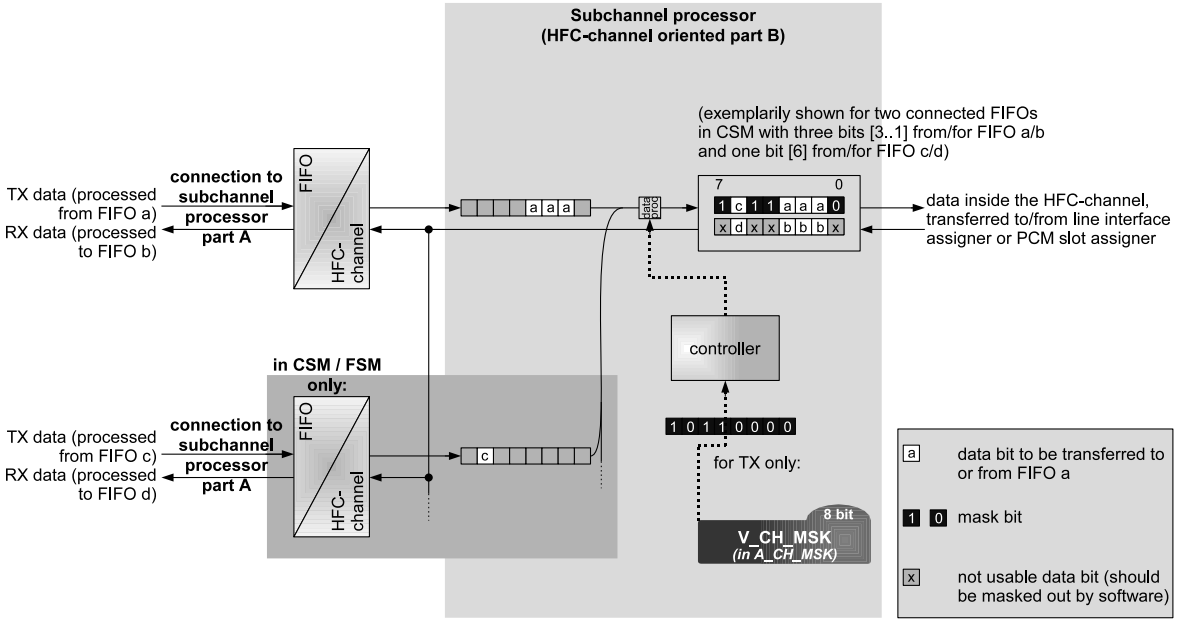


Figure 3.16: *Part B of the subchannel processor*

3.5.4.2 FIFO transmit operation in CSM and FSM

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

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

3.5.4.3 FIFO receive operation in SM

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

3.5.4.4 FIFO receive operation in CSM and FSM

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

3.5.5 Subchannel example for SM

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

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

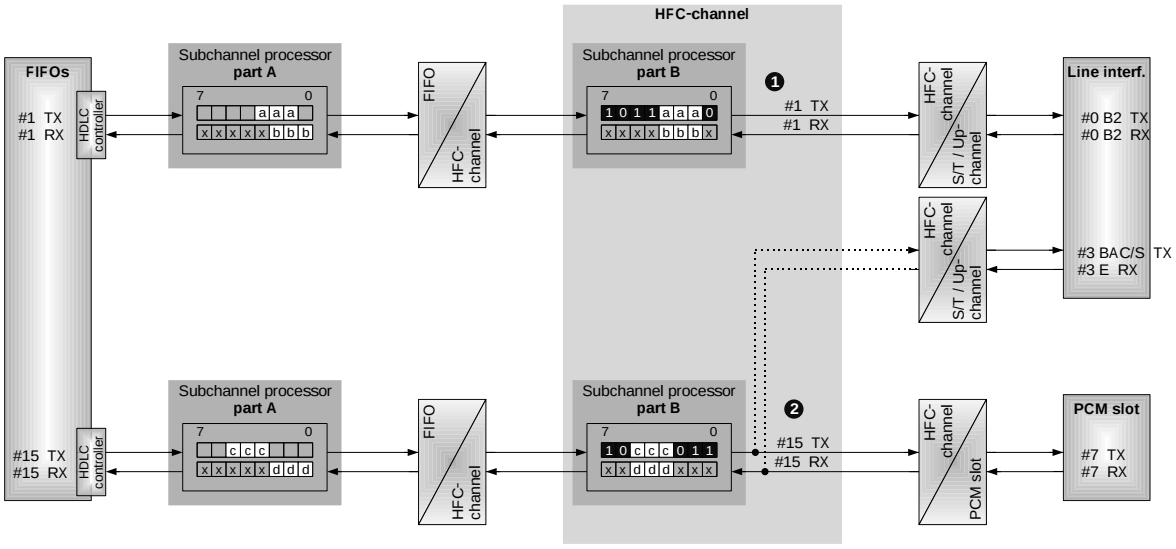


Figure 3.17: *SM example with subchannel processor*

❶ FIFO-to-ST / U_p (TX)

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

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

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

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

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

❶ FIFO-to-ST / U_p (RX)

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

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

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

❷ FIFO-to-PCM (TX)

The second *Simple Mode* configuration connects a FIFO in HDLC mode with the PCM interface [7]. Bitmap V_BIT_CNT in register A_SUBCH_CFG[FIFO] defines three FIFO data bits to

[7] HDLC bit stuffing is not shown in this example.

Table 3.6: *Subchannel processing according to Figure 3.17 (SM ❶ 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 ❶ 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 µs 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 ❷ TX)
R_FIFO	: V_FIFO_DIR	= 0		(transmit FIFO)
	: V_FIFO_NUM	= 15		(FIFO #15)
	: V_REV	= 0		(normal bit order)
A_CON_HDLC[15,TX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_FIFO_IRQ	= 7		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO → ST / U_p , FIFO → PCM)
A_SUBCH_CFG[15,TX]	: V_BIT_CNT	= 3		(process 3 bits)
	: V_START_BIT	= 3		(start with bit 3)
	: V_LOOP_FIFO	= 0		(normal operation)
	: V_INV_DATA	= 0		(normal data transmission)
R_FIFO	: V_FIFO_DIR	= 0		(transmit HFC-channel)
	: V_FIFO_NUM	= 15		(HFC-channel #15)
	: V_REV	= 0		(normal bit order)
A_CH_MSK[15,TX]	: V_CH_MSK	= '1011 0011'		(mask byte)
R_SLOT	: V_SL_DIR	= 0		(transmit slot)
	: V_SL_NUM	= 7		(slot #7)
A_SL_CFG[7,TX]	: V_CH_SDIR	= 0		(transmit HFC-channel)
	: V_CH_SNUM	= 15		(HFC-channel #15)
	: V_ROUT	= '10'		(data to pin STIO1)

❷ 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 ❷ 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 ❷ TX, HDLC mode)*

	7							0
HFC-channel mask:	1	0	0	0	0	0	1	1
HFC-channel transmit byte 1:	1	0	$c_1[2]$	$c_1[1]$	$c_1[0]$	0	1	1
HFC-channel transmit byte 2:	1	0	$c_1[5]$	$c_1[4]$	$c_1[3]$	0	1	1
HFC-channel transmit byte 3:	1	0	$c_2[0]$	$c_1[7]$	$c_1[6]$	0	1	1
HFC-channel transmit byte 4:	1	0	$c_2[3]$	$c_2[2]$	$c_2[1]$	0	1	1
...				...				

3.5.6 Subchannel example for CSM

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

❶ FIFO-to-ST / U_p (TX)

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

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

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

	7							0
HFC-channel receive byte 1:	x	x	$d_1[2]$	$d_1[1]$	$d_1[0]$	x	x	x
HFC-channel receive byte 2:	x	x	$d_1[5]$	$d_1[4]$	$d_1[3]$	x	x	x
HFC-channel receive byte 3:	x	x	$d_2[0]$	$d_1[7]$	$d_1[6]$	x	x	x
HFC-channel receive byte 4:	x	x	$d_2[3]$	$d_2[2]$	$d_2[1]$	x	x	x
...			...					

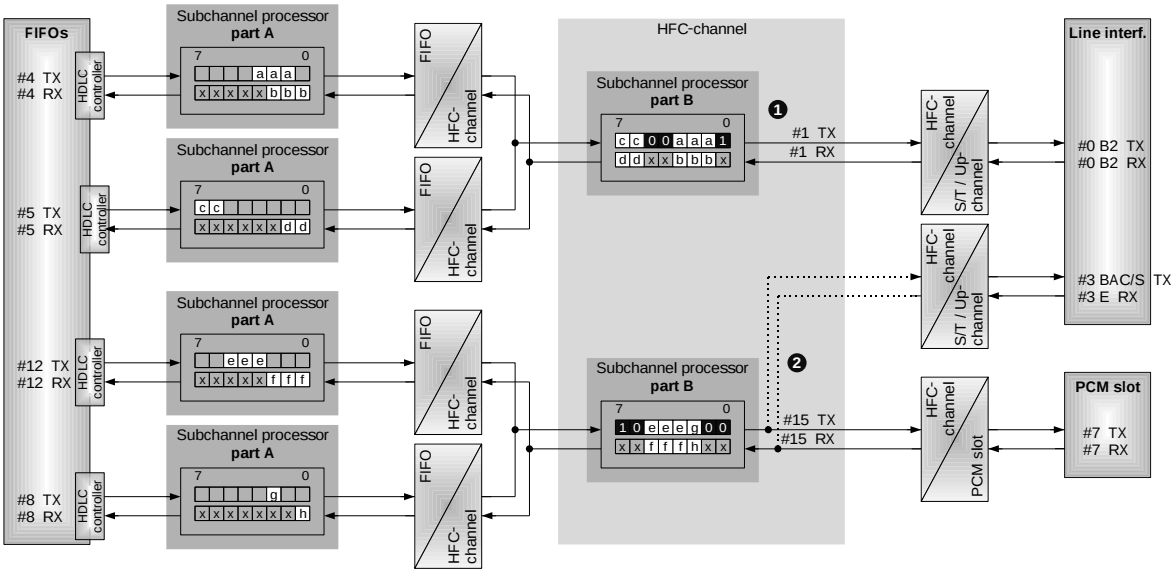


Figure 3.18: *CSM example with subchannel processor*

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_1, the second byte is a_2, and so on. FIFO[5,TX] is represented by the data bytes c_1, c_2, and so on.

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

❶ FIFO-to-ST / U_p (RX)

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

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

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

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

	7							0
HFC-channel mask:	0	0	0	0	0	0	0	1
HFC-channel transmit byte 1:	$c_1[1]$	$c_1[0]$	0	0	$a_1[2]$	$a_1[1]$	$a_1[0]$	1
HFC-channel transmit byte 2:	$c_2[1]$	$c_2[0]$	0	0	$a_2[2]$	$a_2[1]$	$a_2[0]$	1
HFC-channel transmit byte 3:	$c_3[1]$	$c_3[0]$	0	0	$a_3[2]$	$a_3[1]$	$a_3[0]$	1
...					...			

❷ 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 ❶ 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 [8].

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 µs 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.

[8] HDLC bit stuffing is not shown in this example.

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

❷ 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 ❷ 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 ❷ 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 ❷ 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-2S4U / 4SU FIFO registers*

Write only registers:

Address	Name	Page
0x0B	R_FIRST_FIFO	135
0x0C	R_FIFO_THRES	136
0x0D	R_FIFO_MD	137
0x0E	A_INC_RES_FIFO	138
0x0F	R_FIFO	139
0x0F	R_FSM_IDX	139

Read only register:

Address	Name	Page
0x04	A_Z1	140
0x06	A_Z2	140
0x0C	A_F1	140
0x0D	A_F2	141
0x0E	A_FIFO_STA	142
0x14	A_USAGE	143
0x24	R_FILL_BL0	143
0x25	R_FILL_BL1	144
0x26	R_FILL_BL2	145
0x27	R_FILL_BL3	146

Read / write registers:

Address	Name	Page
0x80	A_FIFO_DATA	147
0x84	A_FIFO_DATA_NOINC	147
0xF4	A_CH_MSK	148
0xFA	A_CON_HDLC	149
0xFB	A_SUBCH_CFG	151
0xFC	A_CHANNEL	152
0xFD	A_FIFO_SEQ	153
0xFF	A_FIFO_CTRL	154

4.1 Overview

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

There is a transmit and a receive FIFO for every B-, D- and E-channel.

The FIFO control registers are used to select and control the FIFOs of XHFC-2S4U / 4SU. The FIFO register set exists for every FIFO number and receive / transmit direction. A FIFO is selected by the FIFO select register R_FIFO.

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

4.2 FIFO counters

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

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

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

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

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

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

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

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

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

In addition, a hardware reset initializes the counters.

[1] See Z_{max} value in Table 4.2.

Please note !

Abort D-channel transmission

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

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

Register setup:

R_FIRST_FIFO	: V_FIRST_FIFO_DIR	=	*dir*	Select FIFO data direction *
	: V_FIRST_FIFO_NUM	=	*n*	Select FIFO number
repeat until				
R_STATUS	: V_BUSY	=	0	Wait until not busy
A_SU_CTRL1[ST/Up]	: V_D_RES	=	1	D-channel reset
A_INC_RES_FIFO[FIFO]	: V_RES_FIFO_ERR	=	1	Reset FIFO error *
	: V_RES_FIFO	=	1	Reset selected FIFO
repeat until				
R_STATUS	: V_BUSY	=	0	Wait until not busy
A_SU_CTRL1[ST/Up]	: V_D_RES	=	0	Normal D-channel operation

* Single register write access together with the following command.

Important !

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

Changing a FIFO, reseting a FIFO or incrementing the F-counters causes a short BUSY period of XHFC-2S4U / 4SU. 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 µs). Status, interrupt and control registers can be read and written at any time.

Please note !

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

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

4.3 FIFO size setup

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

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

Table 4.2: *FIFO size setup* *

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

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

4.4 FIFO operation

> **Important !**
>
> The HDLC controller needs F0IO and either C4IO or C2IO clocks for operation. These clocks are generated from XHFC-2S4U / 4SU if PCM master mode is selected (V_PCM_MD = '1').
>
> In PCM slave mode (V_PCM_MD = '0'), F0IO and either C4IO or C2IO clocks must be feed into these pins (see Figure 6.5 on page 231, signal C24I is required).

4.4.1 HDLC transmit FIFOs

Data can be transmitted from the host bus interface to the FIFO with write access to registers A_FIFO_DATA and A_FIFO_DATA_NOINC. XHFC-2S4U / 4SU converts the data into HDLC code and tranfers it from the FIFO to the ST / U_p or the PCM bus interface.

XHFC-2S4U / 4SU checks $Z1$ and $Z2$. If $Z1 = Z2$ (FIFO empty), XHFC-2S4U / 4SU generates a HDLC flag ('0111 1110') or continuous '1's (depending on bit V_IFF in register A_CON_HDLC) and transmits it to the ST / U_p interface. In this case $Z2$ is not incremented. If also $F1 = F2$ only HDLC flags

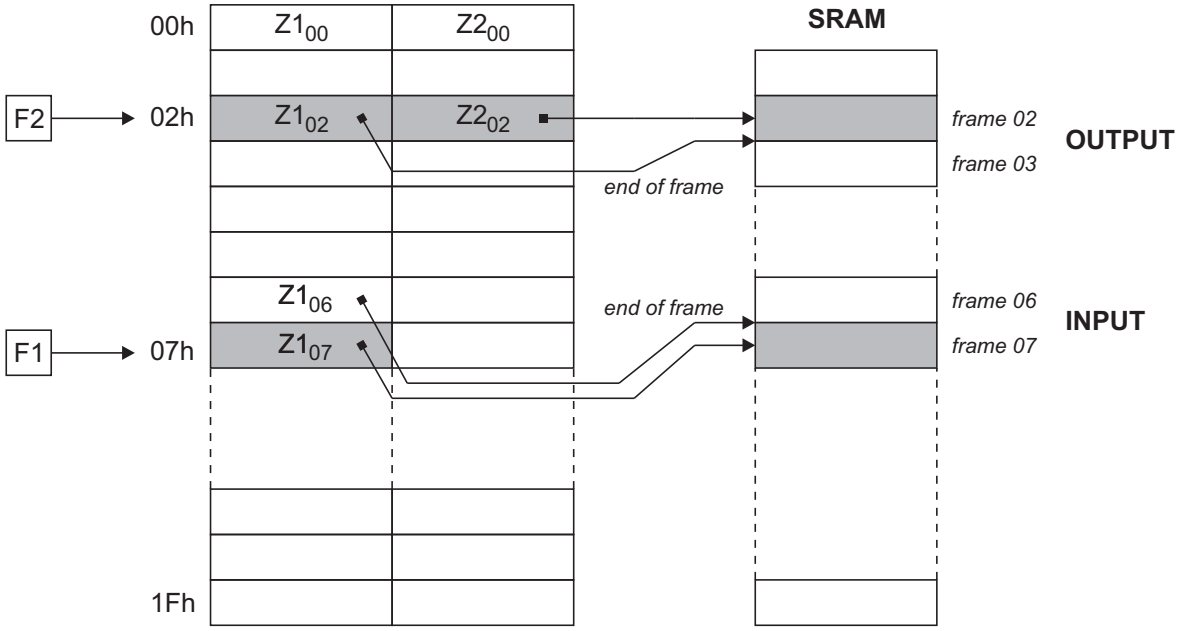


Figure 4.1: *FIFO organization*

or continuous '1's are sent to the ST / U_p interface and all counters remain unchanged. If the frame counters are unequal $F2$ is incremented and XHFC-2S4U / 4SU 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-2S4U / 4SU. $Z1(F2)$ 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-2S4U / 4SU tries to repeat the frame automatically.

Please note !

XHFC-2S4U / 4SU 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.

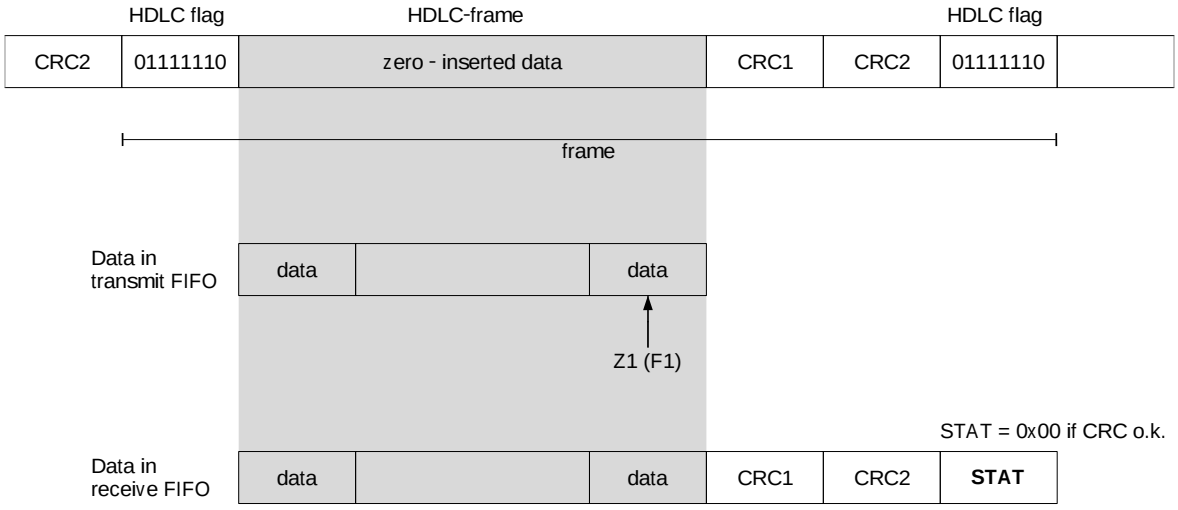


Figure 4.2: *FIFO data organization in HDLC mode*

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-2S4U / 4SU 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-2S4U / 4SU into plain data. After the ending flag of a frame, XHFC-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU. $Z2(F2)$ is used for the frame which is just beeing 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-2S4U / 4SU. 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-2S4U / 4SU

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 accessable and have valid data for FIFO input and output.

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

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

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

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

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

Important !

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

4.5 Register description

4.5.1 Write only registers

R_FIRST_FIFO	(w)	(Reset group: H, 0, 1)	0x0B

First FIFO of the FIFO sequence

This register is only used in *FIFO Sequence Mode*, see register R_FIFO_MD for data flow mode selection.

Bits	Reset value	Name	Description
0	0	**V_FIRST_FIFO_DIR**	**Data direction** This bit defines the data direction of the first FIFO in FIFO sequence. '0' = transmit FIFO data '1' = receive FIFO data
4..1	0	**V_FIRST_FIFO_NUM**	**FIFO number** This bitmap defines the number of the first FIFO in the FIFO sequence.
7..5	0	(reserved)	Must be '000'.

R_FIFO_THRES	(w)	(Reset group: H, 0, 1, 2, 3)	0x0C

FIFO fill level control register

The FIFO fill level can be controlled by a theshold which is specified separately for transmit and receive FIFOs.

Bits	Reset value	Name	Description
3..0	1	V_THRES_TX	**Threshold for all transmit FIFOs** The threshold is a multiple of 16 bytes. 0 = 0 bytes 1 = 16 bytes 2 = 32 bytes ... 15 = 240 bytes
7..4	1	V_THRES_RX	**Threshold for all receive FIFOs** The threshold is a multiple of 16 bytes. 0 = 0 bytes 1 = 16 bytes 0 = 32 bytes ... 15 = 240 bytes

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_FIFO_MD	(w)	(Reset group: H)	0x0D

FIFO mode configuration

This register defines the FIFO number and size. The actual FIFO size depends also on the V_UNIDIR_MD value.

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

A_INC_RES_FIFO [FIFO]	**(w)**	(Reset group: H, 0, 1, 2, 3)	**0x0E**

Increment and reset FIFO register

Before writing this array register the FIFO must be selected by register R_FIFO.

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

R_FIFO	(w)	(Reset group: H, 0, 1)	0x0F

FIFO selection register

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.

Note: This register is a multi-register. It is selected with bitmap V_DF_MD less than '11' in register R_FIFO_MD (SM and CSM). In FSM (V_DF_MD = '11') some FIFO array registers are indexed by the multi-register R_FSM_IDX instead, but most FIFO array registers remain indexed by this register.

Bits	Reset value	Name	Description
0	0	**V_FIFO_DIR**	**FIFO data direction** '0' = transmit FIFO data '1' = receive FIFO data
4..1	0	**V_FIFO_NUM**	**FIFO number**
6..5	0	(reserved)	Must be '00'.
7	0	**V_REV**	**Bit order** '0' = LSB first '1' = MSB first LSB first is used in HDLC mode while MSB first is used in transparent mode. The bit order is being reversed for the data written into the FIFO or when the data is read from the FIFO.

R_FSM_IDX	(w)	(Reset group: H, 0, 1)	0x0F

Index register of the FIFO sequence

This register is used to select a list number in *FIFO Sequence Mode*. 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.

Note: This register is a multi-register. It is selected with bitmap V_DF_MD = '11' in register R_FIFO_MD. In FSM only few FIFO array registers are indexed by this multi-register, but most FIFO array registers remain indexed by R_FIFO.

Bits	Reset value	Name	Description
4..0	0x00	**V_IDX**	**List index** The list index must be in the range 0..31.
7..5	0	(reserved)	Must be '000'.

4.5.2 Read only registers

A_Z1 [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x04**

FIFO input counter $Z1$

Before reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0		**V_Z1**	**Counter value of** $Z1$ The reset value is Z_{max} and depends on the FIFO configuration.

(See Table 4.2 for reset value.)

A_Z2 [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x06**

FIFO output counter $Z2$

Before reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0		**V_Z2**	**Counter value of** $Z2$ The reset value is Z_{max} and depends on the FIFO configuration.

(See Table 4.2 for reset value.)

A_F1 [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x0C**

FIFO input HDLC frame counter $F1$

Before reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0	7	**V_F1**	**Counter value** Up to 7 HDLC frames can be stored in every FIFO.

A_F2 [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x0D**

FIFO output HDLC frame counter *F2*

Before reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0	7	**V_F2**	**Counter value** Up to 7 HDLC frames can be stored in every FIFO.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

A_FIFO_STA [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x0E**

FIFO status register

Before reading this array register the FIFO must be selected by register R_FIFO.

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

A_USAGE [FIFO]	**(r)**	(Reset group: H, 0, 1)	**0x14**

FIFO fill level

Before reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7 . . 0	0x00	**V_USAGE**	**Number of bytes currently stored in the FIFO** The FIFO is empty when this register has the value 0x00.

R_FILL_BL0	**(r)**	(Reset group: H, 0, 1)	**0x24**

FIFO fill level for FIFO block 0

When a bit is set to '1', more than the specified number of bytes is currently being in the FIFO. The threshold is separately defined for transmit and receive FIFOs, V_THRES_TX for transmit FIFOs and V_THRES_RX for receive FIFOs in register R_FIFO_THRES.

Bits	Reset value	Name	Description
0	0	**V_FILL_FIFO0_TX**	**FIFO[0,TX] fill level**
1	0	**V_FILL_FIFO0_RX**	**FIFO[0,RX] fill level**
2	0	**V_FILL_FIFO1_TX**	**FIFO[1,TX] fill level**
3	0	**V_FILL_FIFO1_RX**	**FIFO[1,RX] fill level**
4	0	**V_FILL_FIFO2_TX**	**FIFO[2,TX] fill level**
5	0	**V_FILL_FIFO2_RX**	**FIFO[2,RX] fill level**
6	0	**V_FILL_FIFO3_TX**	**FIFO[3,TX] fill level**
7	0	**V_FILL_FIFO3_RX**	**FIFO[3,RX] fill level**

R_FILL_BL1	(r)	(Reset group: H, 0, 1)	0x25

FIFO fill level for FIFO block 1

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_FIFO4_TX**	**FIFO[4,TX] fill level**
1	0	**V_FILL_FIFO4_RX**	**FIFO[4,RX] fill level**
2	0	**V_FILL_FIFO5_TX**	**FIFO[5,TX] fill level**
3	0	**V_FILL_FIFO5_RX**	**FIFO[5,RX] fill level**
4	0	**V_FILL_FIFO6_TX**	**FIFO[6,TX] fill level**
5	0	**V_FILL_FIFO6_RX**	**FIFO[6,RX] fill level**
6	0	**V_FILL_FIFO7_TX**	**FIFO[7,TX] fill level**
7	0	**V_FILL_FIFO7_RX**	**FIFO[7,RX] fill level**

R_FILL_BL2	(r)	(Reset group: H, 0, 1)	0x26

FIFO fill level for FIFO block 2

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_FIFO8_TX**	**FIFO[8,TX] fill level**
1	0	**V_FILL_FIFO8_RX**	**FIFO[8,RX] fill level**
2	0	**V_FILL_FIFO9_TX**	**FIFO[9,TX] fill level**
3	0	**V_FILL_FIFO9_RX**	**FIFO[9,RX] fill level**
4	0	**V_FILL_FIFO10_TX**	**FIFO[10,TX] fill level**
5	0	**V_FILL_FIFO10_RX**	**FIFO[10,RX] fill level**
6	0	**V_FILL_FIFO11_TX**	**FIFO[11,TX] fill level**
7	0	**V_FILL_FIFO11_RX**	**FIFO[11,RX] fill level**

R_FILL_BL3	(r)	(Reset group: H, 0, 1)	0x27

FIFO fill level for FIFO block 3

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

A_FIFO_DATA [FIFO]	**(r/w)**	(Reset group: –)	**0x80**

FIFO data register

Before writing or reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0		**V_FIFO_DATA**	**Data byte** Read / write one byte from / to the FIFO selected with register R_FIFO and increment *Z*-counter by 1.

A_FIFO_DATA_NOINC [FIFO]	**(r/w)**	(Reset group: –)	**0x84**

FIFO data register

Before writing or reading this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0		**V_FIFO_DATA_NOINC**	**Data byte** Write one byte to or read one byte from the FIFO selected with register R_FIFO without incrementing *Z*-counter.

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

A_CH_MSK [FIFO]	**(r*/w)**	(Reset group: H, 0, 1)	**0xF4**

HFC-channel data mask for the selected transmit HFC-channel

For receive FIFOs this register is ignored.

Before writing this array register, the HFC-channel must be selected by register R_FIFO.

Bits	Reset value	Name	Description
7..0	0xFF	**V_CH_MSK**	**Mask byte** 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. This register has only a meaning when V_BIT_CNT $\neq 0$ in register A_SUBCH_CFG.

(See Section 2.2.3.2 on page 47 for details on Read* access.)

A_CON_HDLC [FIFO] **(r*/w)** (Reset group: H, 0, 1) **0xFA**

HDLC and connection settings of the selected FIFO

Before writing this array register the FIFO must be selected by register R_FIFO.

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

(continued on next page)

(continued from previous page)

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

(See Section 2.2.3.2 on page 47 for details on Read* access.)

A_SUBCH_CFG [FIFO] **(r*/w)** (Reset group: H, 0, 1) **0xFB**

Subchannel parameters for bit processing of the selected FIFO

Before writing this array register the FIFO must be selected by register R_FIFO.

Note: For D- and E-channels this register must be 0x02.

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

(See Section 2.2.3.2 on page 47 for details on Read* access.)

A_CHANNEL [FIFO]	**(r*/w)**	(Reset group: H, 0, 1)	**0xFC**

HFC-channel assignment for the selected FIFO

This register is only used in *Channel Select Mode* and *FIFO Sequence Mode*.

Before writing this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
0	0	**V_CH_FDIR**	**HFC-channel data direction** '0' = HFC-channel for transmit data '1' = HFC-channel for receive data **Reset value:** This bitmap is reset to the same value as the current FIFO, i.e. V_CH_FDIR of A_CHANNEL[*number,direction*] = *direction*.
4..1	0	**V_CH_FNUM**	**HFC-channel number** (0..15) **Reset value:** This bitmap is reset to the same value as the current FIFO, i.e. V_CH_FNUM of A_CHANNEL[*number,direction*] = *number*.
7..5		(reserved)	Must be '000' when written.

(See Section 2.2.3.2 on page 47 for details on Read* access.)

A_FIFO_SEQ [FIFO]	(r*/w)	(Reset group: H, 0, 1)	0xFD

FIFO sequence list

This register is only used in *FIFO Sequence Mode*.

Before writing this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
0	0	**V_NEXT_FIFO_DIR**	**FIFO data direction** This bit defines the data direction of the next FIFO in FIFO sequence. '0' = transmit FIFO data '1' = receive FIFO data **Reset value:** This bitmap is reset to the same value as the current FIFO, i.e. V_NEXT_FIFO_DIR of A_FIFO_SEQ[*number*,*direction*] = *direction*.
4 . . 1	0	**V_NEXT_FIFO_NUM**	**FIFO number** This bitmap defines the FIFO number of the next FIFO in the FIFO sequence. **Reset value:** This bitmap is reset to the same value as the current FIFO, i.e. V_NEXT_FIFO_NUM of A_FIFO_SEQ[*number*,*direction*] = *number*.
5		(reserved)	Must be '0' when written.
6	0	**V_SEQ_END**	**End of FIFO list** '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		(reserved)	Must be '0' when written.

(See Section 2.2.3.2 on page 47 for details on Read* access.)

A_FIFO_CTRL [FIFO] **(r*/w)** (Reset group: H, 0, 1) **0xFF**

Control register for the selected FIFO

Before writing this array register the FIFO must be selected by register R_FIFO.

Bits	Reset value	Name	Description
0	0	**V_FIFO_IRQMSK**	**Interrupt mask for the selected FIFO** '0' = The FIFO interrupt is not used for generating a signal on the interrupt pin 22. 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 22. **Note:** In addition to this interrupt mask, FIFO interrupt must be enabled globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.
1	0	**V_BERT_EN**	**BERT enable** '0' = BERT disabled, normal data is transmitted and received '1' = BERT enabled, output of BERT generator is transmitted and received data is checked by BERT
2	0	**V_MIX_IRQ**	**Mixed interrupt generation in HDLC mode** This bit is only used in HDLC mode and it should be '0' in transparent mode. '0' = FIFO interrupts are generated either on *end of frame* (in HDLC mode) or periodically (in transparent mode) '1' = FIFO interrupts are generated both on *end of frame* and periodically when the bits [n-1:0] of the $Z2$-counter (in transmit direction) or the $Z1$-counter (in receive direction) change from all '1's to all '0's ($n =$ V_FIFO_IRQ $+ 3$ in register A_CON_HDLC)

(continued on next page)

(continued from previous page)

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

(See Section 2.2.3.2 on page 47 for details on Read* access.)

Chapter 5

Universal ISDN Port

Table 5.1: *Overview of the ST / U_p interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x16	R_SU_SEL	195	0x13	R_AF0_OVIEW	211
0x30	A_SU_WR_STA	196	0x30	A_SU_RD_STA	212
0x31	A_SU_CTRL0	197	0x31	A_SU_DLYL	213
0x32	A_SU_CTRL1	199	0x32	A_SU_DLYH	214
0x33	A_SU_CTRL2	200	0x34	A_MS_RX	215
0x34	A_MS_TX	202	0x35	A_SU_STA	216
0x35	A_ST_CTRL3	203	0x3C	A_B1_RX	217
0x35	A_UP_CTRL3	204	0x3D	A_B2_RX	218
0x36	A_MS_DF	206	0x3E	A_D_RX	219
0x37	A_SU_CLK_DLY	207	0x3F	A_E_RX	220
0x3C	A_B1_TX	208			
0x3D	A_B2_TX	208			
0x3E	A_D_TX	209			
0x3F	A_BAC_S_TX	210			

5.1 General overview of the S/T and U_p interfaces

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

5.1.1 Array registers and multiregisters

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

Furthermore, all registers are described to be array registers because there are several line interfaces in XHFC-2S4U / 4SU and each has the complete set of registers. So, the Universal ISDN Port module – with multiple entities – has registers which are multi-registers *and* array registers at the same time.

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

1. **Array register selection:** The line interface number must be selected first by writing the appropriate value into bitmap V_SU_SEL of register R_SU_SEL.
   - XHFC-2S4U: S/T interfaces 0 . . 1 available, U_p interfaces 0 . . 3 available
   - XHFC-4SU: S/T interfaces 0 . . 3 available, U_p interfaces 0 . . 3 available
2. **Multiregister selection:** Then the interface mode can be chosen (only in the initialization procedure). The selected line interface works in S/T mode with the bitmap value V_ST_SEL = '0' in register A_ST_CTRL3. The line interface can be switched into U_p mode with V_ST_SEL = '1' [1]. It is strongly recommended to select the interface mode before any other register of the selected line interface is written.
3. **Any other register access:** Now all registers of the selected line interface in its chosen mode (S/T or U_p) can be accessed.

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

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

5.1.2 Block diagram of the Universal ISDN Port module

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

[1] 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_ST_CTRL3 is used while a line interface in U_p mode uses A_UP_CTRL3.

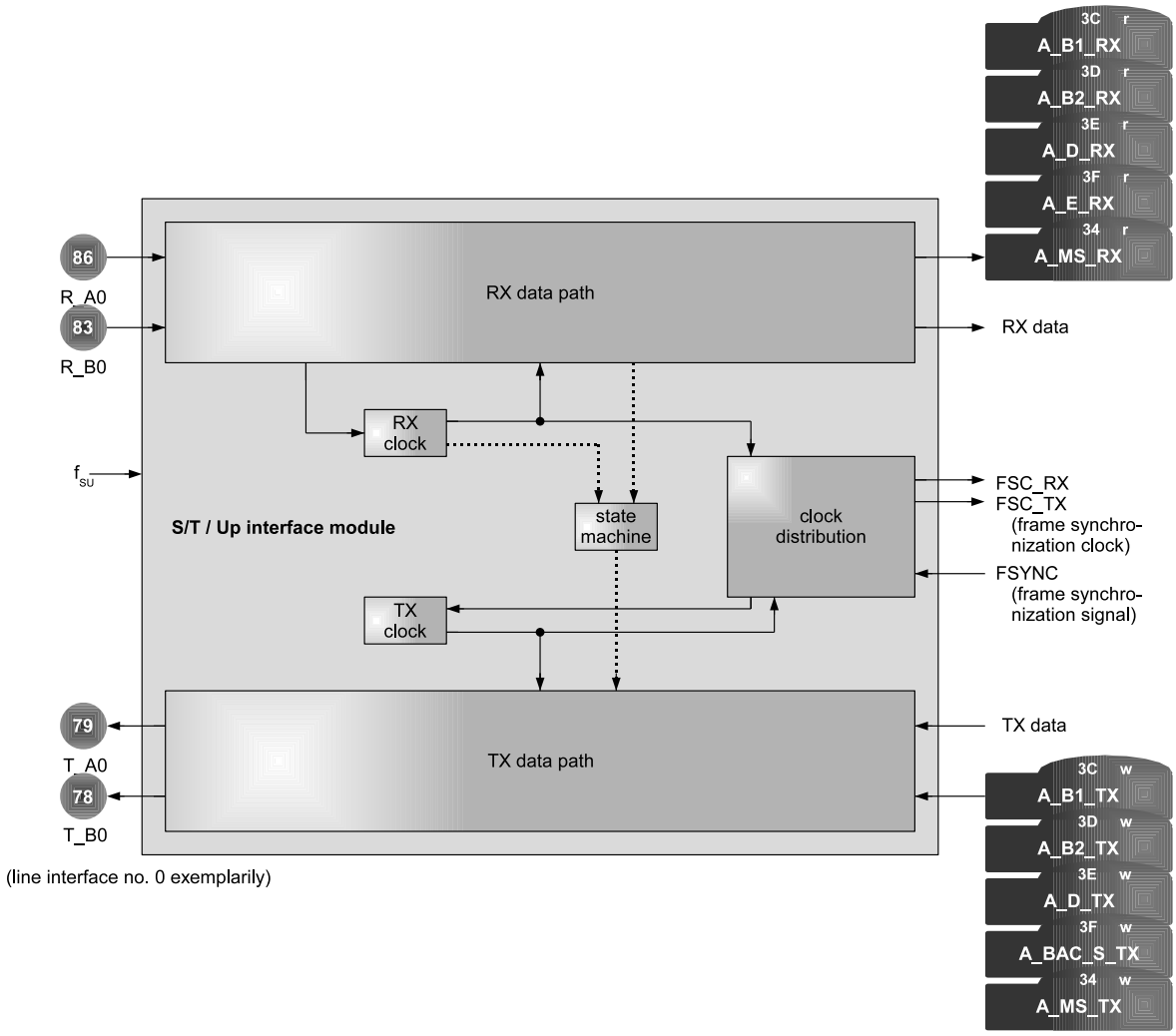


Figure 5.1: *Overview of the Universal ISDN Port module (For details see Figures 5.13 or 5.5)*

5.2 S/T interface description

5.2.1 Overview

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

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

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

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

5.2.2 Frame Structure

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

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

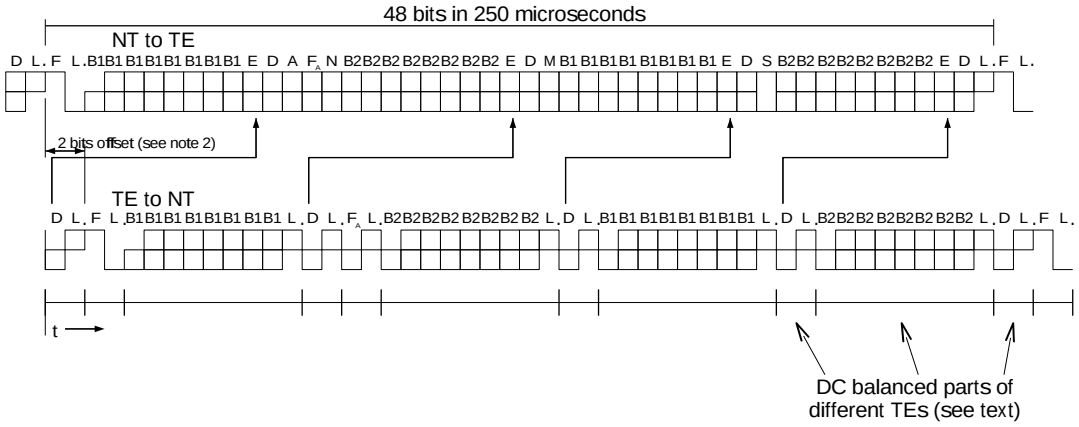


Figure 5.2: *Frame structure at reference points S and T (see legend in Table 5.2 and specification [9])*

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

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

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

Table 5.2: *Legend for Figure 5.2*

NT-to-TE & TE-to-NT:

Code	Description
F	Framing bit, marks the start of the frame (1 bit/frame)
B1	Bit within the B1-channel (2 byte/frame)
B2	Bit within the B2-channel (2 byte/frame)
D	Bit within the D-channel (4 bit/frame)
L	DC balancing bit (NT-to-TE: 2 bit/frame, TE-to-NT: 10 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)

NT-to-TE only:

Code	Description
E	Bit within the E-channel (D-echo-channel, 4 bit/frame)
M	Multiframing bit, marks the start of the multiframe in every 20th frame (1 bit/frame)
N	Boolean complementation of the auxiliary framing bit F_A, $N = \overline{F}_A$ (1 bit/frame)
S	S-bit of the multiframe (1 bit/frame)
A	Activation bit (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 µs 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 multiframes. Table 5.3 shows the position of the multiframe bits. A detailed specification of the multiframe structure is given in [9].

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

5.2.4 Data transmission

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

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

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

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

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

Multiframe bits can be handled by the data flow controller (involving registers A_BAC_S_TX and

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

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

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

*2: The specification [9] defines five subchannels for the S-multiframe. Only subchannel 1 is used from XHFC-2S4U/4SU. All other subchannels are set to '0'.

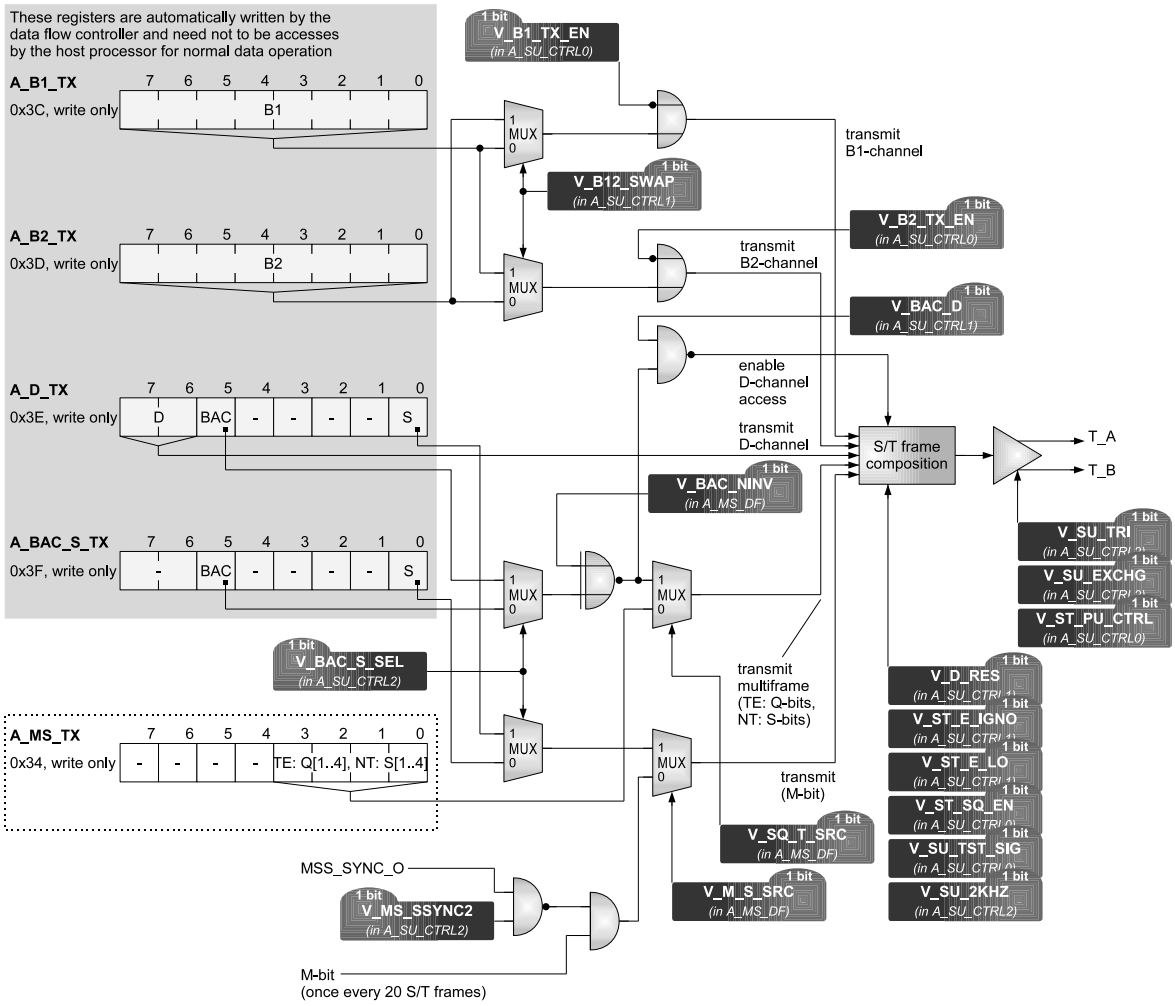


Figure 5.3: *S/T frame composition for B1-, B2-, D- and multiframe bits (S/T interface mode, transmit direction)*

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 [2]:

INFO 0: No signal on line.

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

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

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

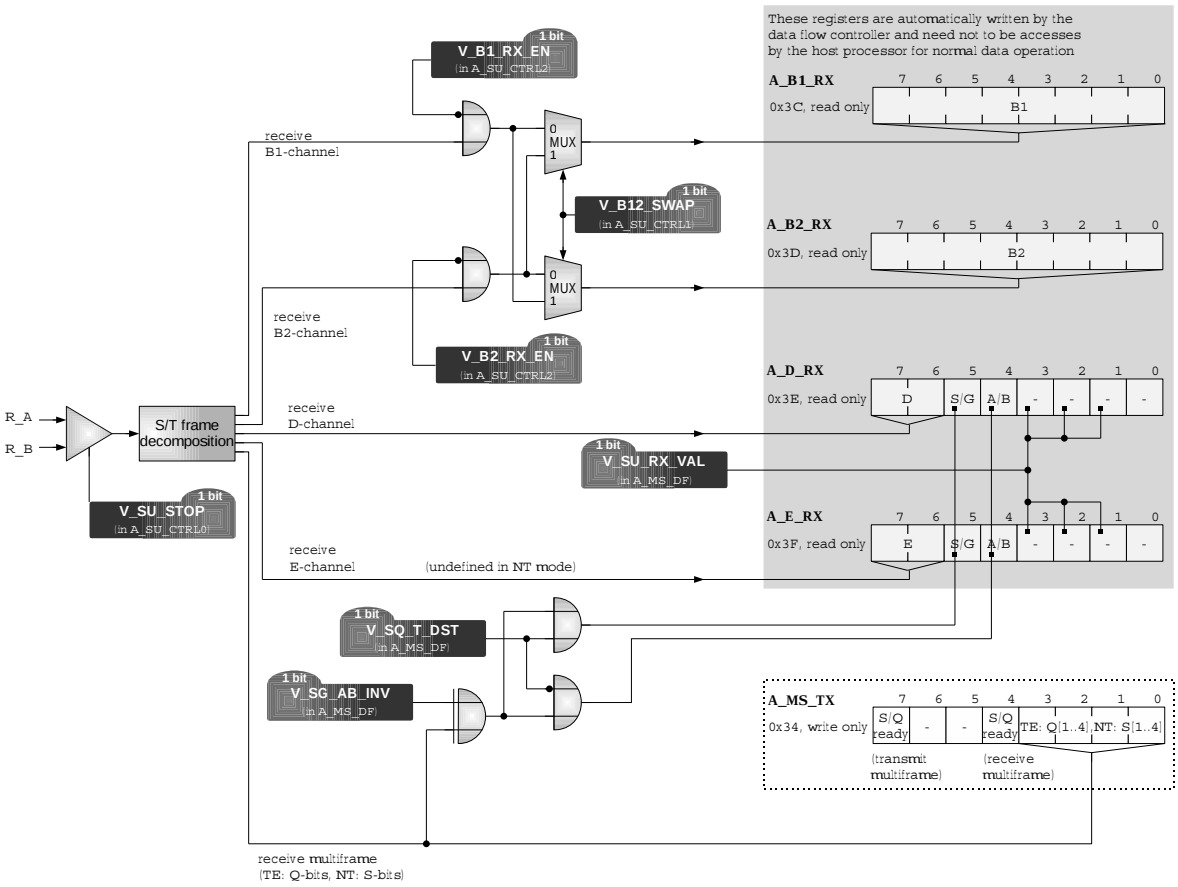


Figure 5.4: *S/T frame decomposition for B1-, B2-, D-, E-channel and multiframe bits (S/T interface mode, receive direction)*

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 [9]. So the current Fx or Gx state can be read out of register A_SU_RD_STA. However, it is possible to overwrite the current state by setting bit V_SU_LD_STA in register A_SU_WR_STA.

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

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

> **Important !**
>
> The S/T state machine is stuck at F0 or G0 after a reset. The interface sends no signal on the S/T line (INFO 0) and is not able to activate it by incoming INFO x in this state. Writing '0' into bit V_SU_LD_STA of register A_SU_WR_STA starts the state machine.
>
> **NT mode:** The NT state machine does not change automatically from G2 to G3 if the TE side sends INFO 3 frames. This transition must be activated each time by V_G2_G3 in register A_SU_RD_STA or it can permanently be enabled by setting bit V_G2_G3_EN in register A_SU_CTRL1.

Incoming INFO 0 at state F 6 cause a state change to F 3 normally. Sometimes an intermediate state F 7 occurs which stops timer T3. In this case another state change to F 3 comes up within 1 ms and F 7 can be ignored. V_SU_INFO0 in register A_SU_RD_STA should be checked with every state change from F 6 to F 7. When this bit is set, the state should be checked again after about 1 ms. When it is F 3, the intermediate state F 7 has to be ignored. [3]

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

[3] It might be useful to start a timer of approximately 1 ms to detect the F 6 – F 7 – F 3 state changes.

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

State name:	Reset	Deactivated	Pending activation	Active	Pending deactivation
State number:	G 0	G 1	G 2	G 3	G 4
INFO sent:	INFO 0	INFO 0	INFO 2	INFO 4	INFO 0
Event:					
State machine release [*3]	G 1	|	|	|	|
Activate request	start T 1 [*1] G 2	start T 1 [*1] G 2	|	|	start T 1 [*1] G 2
Deactivate request	—	|	start T 2 G 4	start T 2 G 4	|
Expiry T 1 [*1]	—	—	start T 2 G 4	/	—
Expiry T 2 [*2]	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1	—	start T 1 [*1] G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 [*1,4] G 3	—	—
Lost framing	—	/	/	G 2	—

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

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

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

[*3]: After reset the state machine is fixed to G 0.

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

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

State name:	Reset	Sensing	Deactivated	Awaiting signal	Identifying input	Synchronized	Activated	Lost framing
State number:	F 0	F 2	F 3	F 4	F 5	F 6	F 7	F 8
INFO sent:	INFO 0	INFO 0	INFO 0	INFO 1	INFO 0	INFO 3	INFO 3	INFO 0
Event:								
State machine release [*1]	F 2	/	/	/	/	/	/	/
Activate request, receiving any signal	—	|	F 5	|	|	—	|	—
Activate request, receiving INFO 0	—	|	start T3 [*5] F 4	|	|	—	|	—
Expiry T 3 [*5]	—	/	—	F 3	F 3	—	—	F 3
Receiving INFO 0	—	F 3	—	—	—	F 3	F 3	F 3
Receiving any signal [*2]	—	—	—	F 5	—	/	/	—
Receiving INFO 2 [*3]	—	F 6	F 6	F 6	F 6	—	F 6	F 6
Receiving INFO 4 [*3]	—	F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	—	stop T3 [*5] F 7
Lost framing [*4]	—	/	/	/	/	F 8	F 8	—

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

[*1]: After reset the state machine is fixed to F 0.

[*2]: This event reflects the case where a signal is received and the TE has not (yet) determined wether it is INFO 2 or INFO 4.

[*3]: Bit and frame synchronization achieved.

[*4]: Loss of Bit or frame synchronization.

[*5]: Timer T 3 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_A3 and R_B0 .. R_B3 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 231).

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 (multiframe / 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 237.

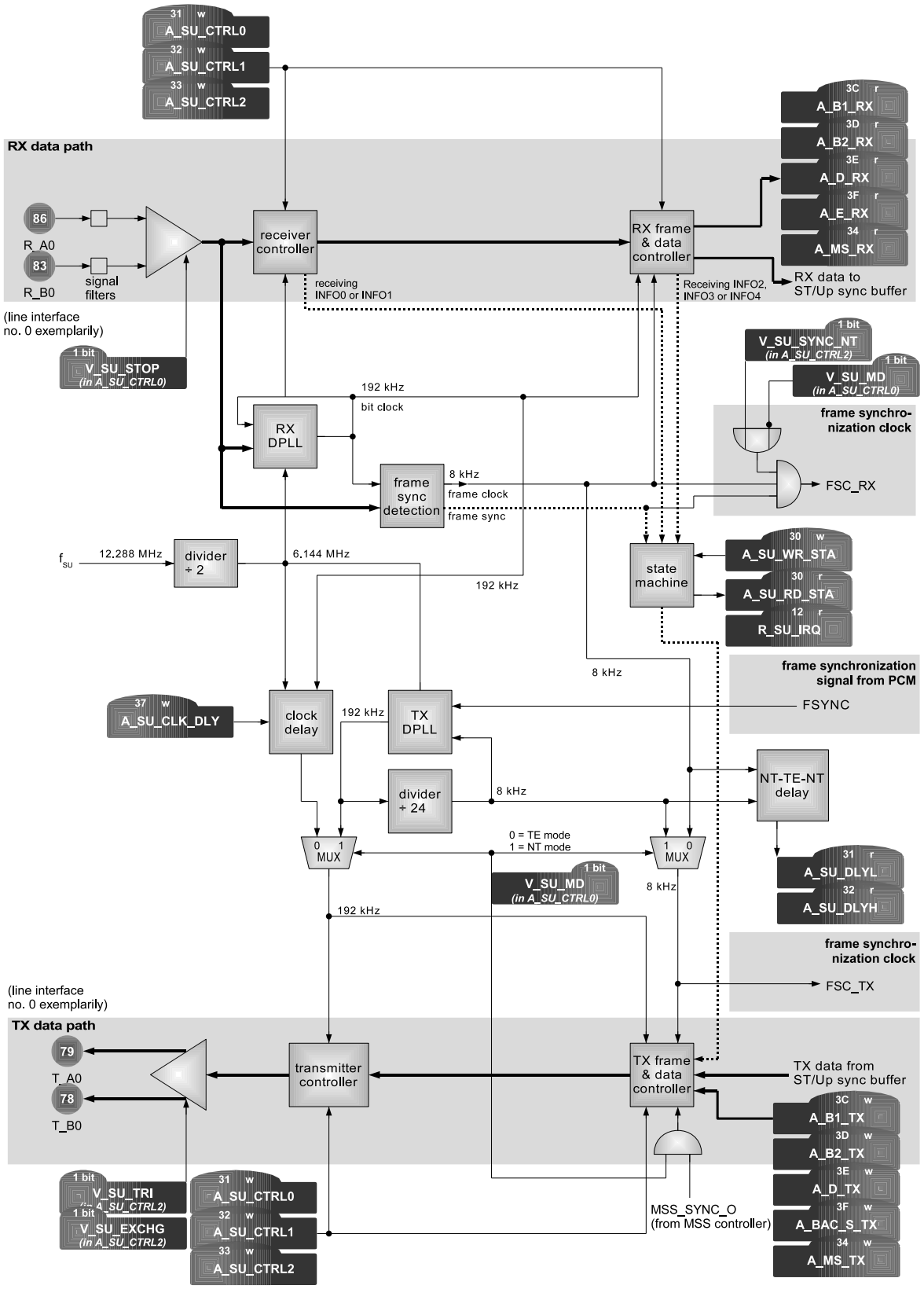


Figure 5.5: *S/T clock synchronization*

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-2S4U / 4SU receiver, it is allowed to swap the pairs R_A0 / L_A0 and R_B0 / L_B0 .

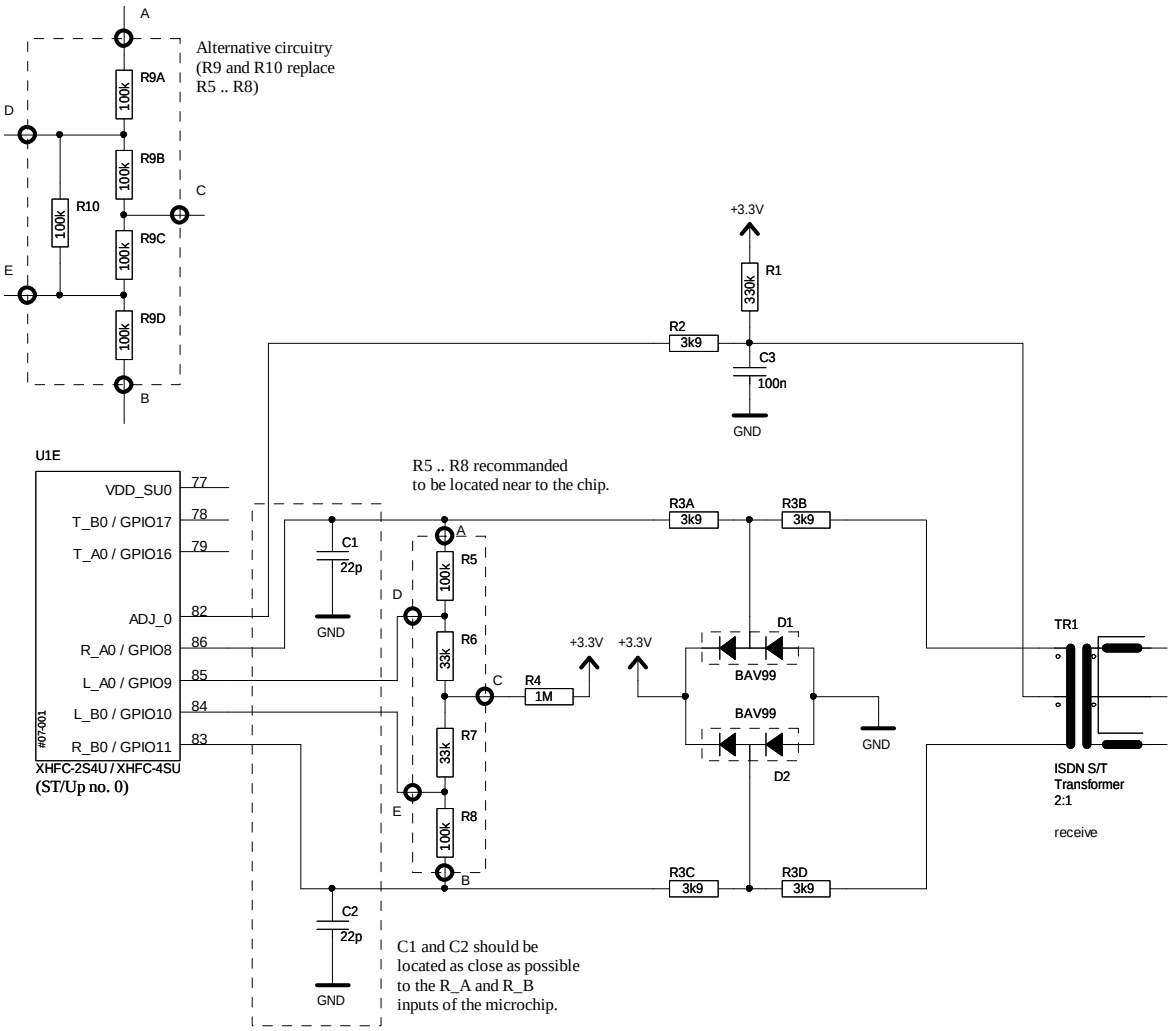


Figure 5.6: *External S/T receive circuitry for TE and NT mode*

XHFC-2S4U has two S/T interfaces while XHFC-4SU has four S/T interfaces. For all not used S/T interfaces, the level adjustment pins ADJ_0 . . ADJ_3 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.

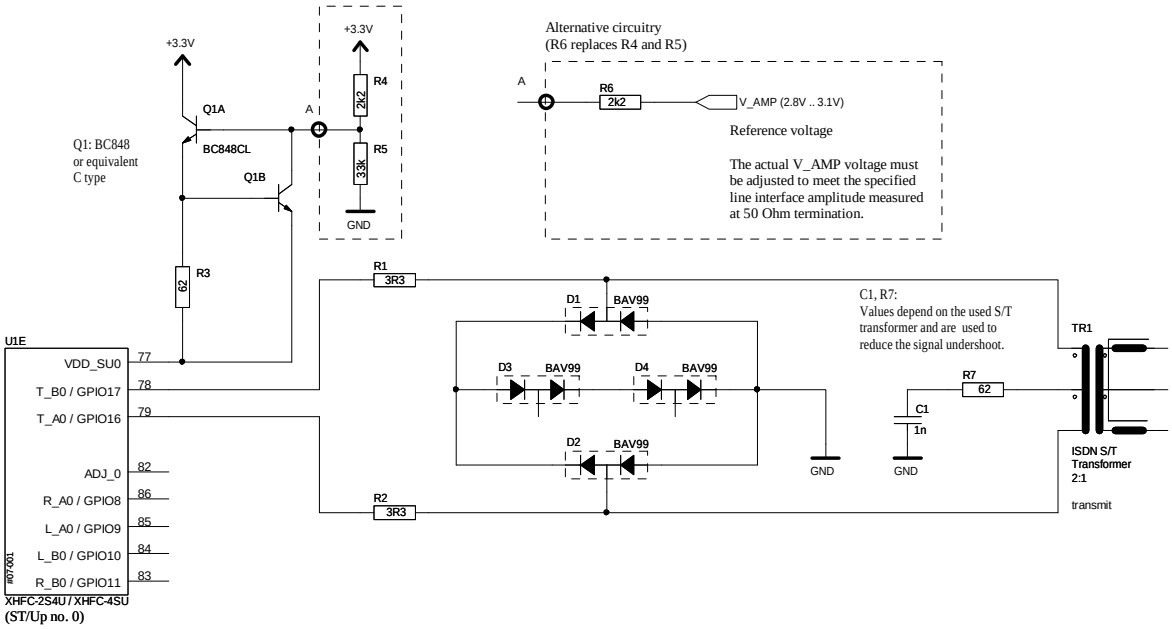


Figure 5.7: *External S/T transmit circuitry for TE and NT mode*

The signal level of the transmit circuitry has to be adjusted by VDD_SU0 .. VDD_SU3 . 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 [4]. The termination resistors R1 and R2 are optional.

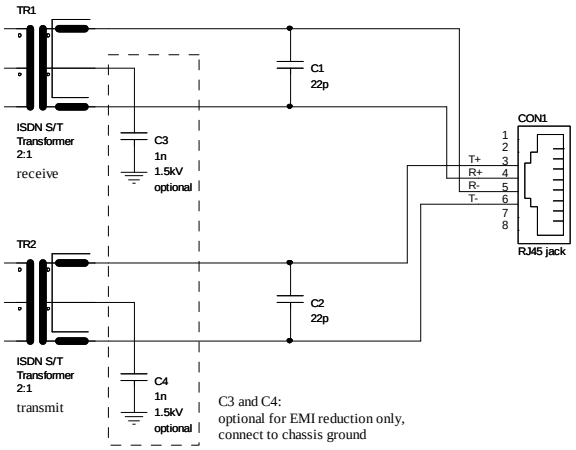


Figure 5.8: *Transformer and connector circuitry in TE mode*

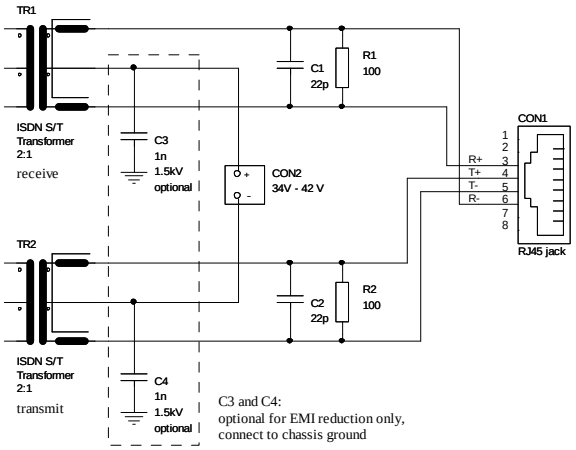
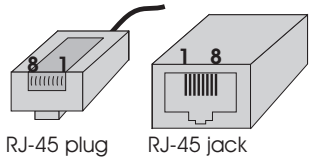


Figure 5.9: *Transformer and connector circuitry in NT mode (shown with optional 100Ω termination, whole bus termination must be 50Ω)*

[4] 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*

Bel Fuse Inc., United Kingdom, http://www.belfuse.com

Type	Device
Dual Transformer Module without choke:	
	2798B (SMD)
	2798C (SMD)
	2798D (SMD)
Dual Transformer Module with choke:	
	APC 48301
Hybrid Transformer Module with choke:	
	S803-6200-01

Pulse Engineering, Inc., United States, http://www.pulseeng.com

Type	Device
Single transformer:	
	T5003 (SMD, PCMCIA)
	T5020 (SMD)
	T5023 (SMD)
	T5024 (SMD, 3 kV)
	T5033 (SMD)
	T5035 (3 kV)
	T5036 (SMD, 3 kV)
	PE-64995
	PE-64999
	PE-68992
	ST-5069 (SMD)
Dual Transformer Module without choke:	
	T5006 (SMD)
	T5007 (SMD)
	T5042 (SMD, 3 kV)
	PE-65495
	PE-65499
	PE-65795 (SMD)
	PE-65799 (SMD)
Dual Transformer Module with choke:	
	T5012
	T5034 (SMD)
	T5038 (SMD)

(continued on next page)

Table 5.6: *S/T transformer part numbers and manufacturers*

(continued from previous page)

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

Type	Device
Single Transformer:	
	ISF-140B1
	ISV-140B1 (3 kV)
	ISHF-240B1 (3 kV)
	SHJ-240B (SMD, 3 kV)
	SMJ-140B (SMD)
	SWJ-140B (SMD)
Dual Transformer Module without choke:	
	MUJ-103A-000 (SMD)
	MAJ-403-000 (SMD)
	MSJ-403A-000 (SMD)
	MHJ-240B1-000 (SMD, 3 kV)

Type	Device
Dual Transformer Module with choke:	
	HVM-140C1
	ISM-140C1
	MUJ-103A-500(SMD)
	MUJ-103A-101(SMD)
	MUJ-103A-501(SMD)
	MUJ-103A-502(SMD)
	MAJ-403A-470 (SMD)
	MAJ-403A-101 (SMD)
	MAJ-403A-501 (SMD)
	MAJ-403A-502 (SMD)
	MSJ-403A-470 (SMD)
	MSJ-403A-101 (SMD)
	MSJ-403A-501 (SMD)
	MSJ-403A-502 (SMD)
	MHJ-240B1-470 (SMD, 3 kV)
	MHJ-240B1-101 (SMD, 3 kV)
	MHJ-240B1-501 (SMD, 3 kV)
	MHJ-240B1-502 (SMD, 3 kV)
	MHJ-240B1-123 (SMD, 3 kV)

(continued on next page)

Table 5.6: *S/T transformer part numbers and manufacturers*

(continued from previous page)

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

Type	Device
Single transformer:	
	UT 20995
	UT 20999
	UT 21023
	UT 21595
	UT 28166
	UT 28166-TS (SMD)
	UT 28428-TS (SMD)
	UT 28729 (4 kV)

Type	Device
Dual Transformer Module without choke:	
	UT 20495-TS (SMD)
	UT 20499-00TS (SMD)
	UT 20765-00 (SMD, 3 kV)
	UT 20795-00TS (SMD)
	UT 21624
	UT 21624 TS (SMD)
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

Type	Device
Single transformer:	
	3-L4021-X066
	3-L4025-X095
	3-L4031-X001
	3-L4097-X029 (3 kV)
	3-L5024-X028 (SMD)
	3-L5032-X040 (SMD, 3 kV)

Type	Device
Dual Transformer Module without choke:	
	7-M4035-X001
	7-M5014-X001 (SMD)
	7-M5026-X001 (SMD)
	7-M5026-X002 (SMD, 3 kV)
	7-M5054-X001 (SMD)
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)

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

Type	Device
Single transformer:	
	503 05 901 00 (SMD)
	503 10 009 00 (SMD)
	503 12 001 00 (PCMCIA)
	503 20 010 00 (SMD)
	503 20 019 00 (SMD, 4 kV)

Type	Device
Dual Transformer Module without choke:	
	503 16 504 00 (SMD)
	503 16 513 00 (SMD, 4 kV)
	503 20 981 00 (SMD)
	503 74 003 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_p interface description

5.3.1 Overview

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

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

A specification conform state machine for TE and LT mode is implemented (see Section 5.3.6). This is very similar to the S/T state machine (see Section 5.2.6 on page 165).

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

5.3.2 Frame Structure

The U_p frame structure has a length of 250 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.

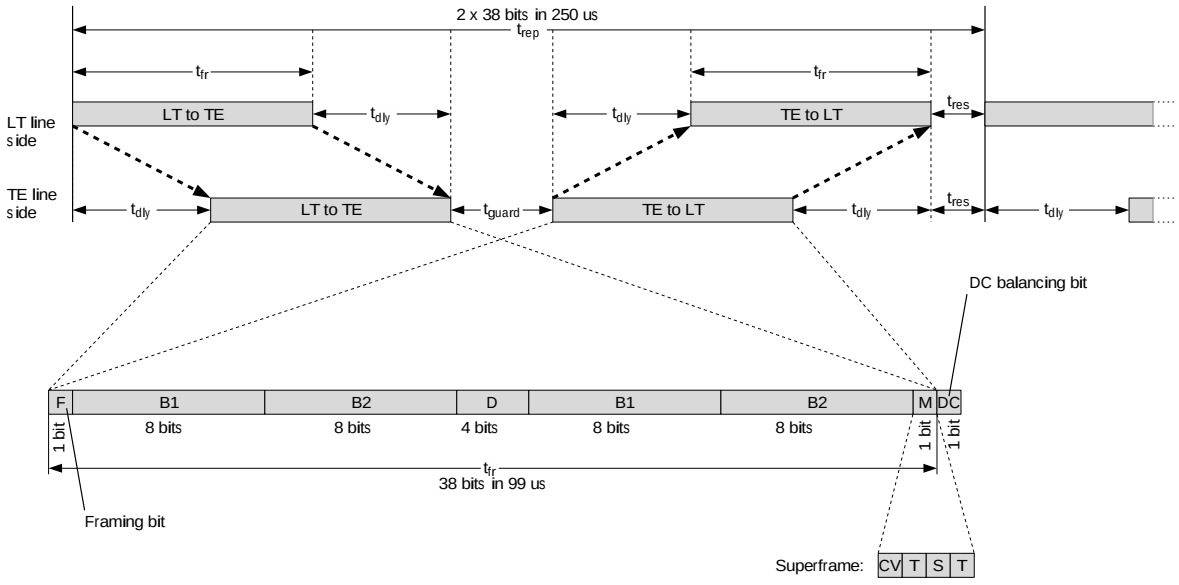


Figure 5.10: *U_p interface frame structure*

5.3.3 Superframe structure

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

Table 5.7: *Legend for Figure 5.10*

Symbol	min / µs	typ / µs	max / µs	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. Due to the fact that the bit scramblers are functionally arranged in front of the B-channel enable / disable logic (V_B1_TX_EN and V_B2_TX_EN), it is important always to enable the B-channels during U_p operation even if no data is send.

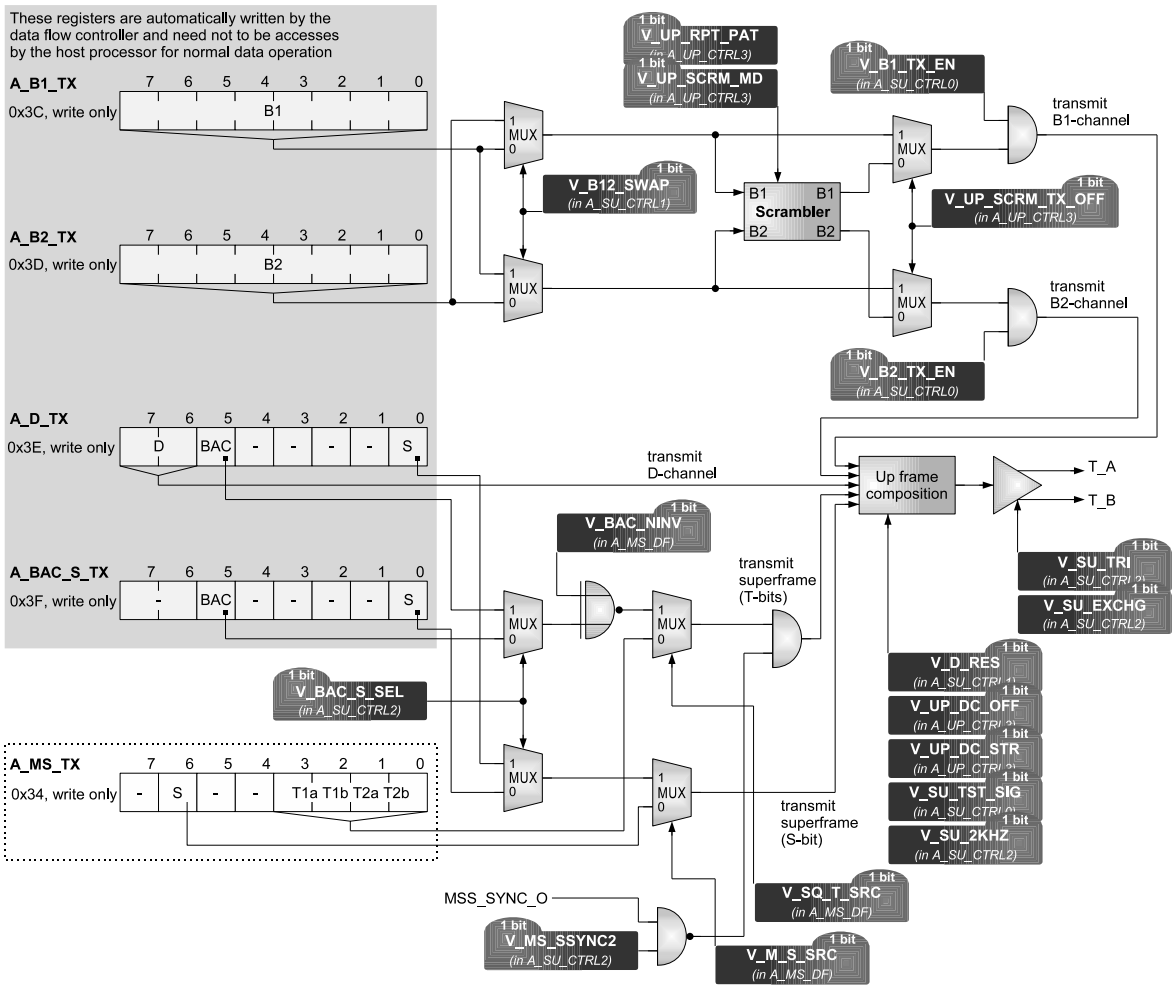


Figure 5.11: *U_p frame composition for B1-, B2-, D-channel and superframe bits (U_p interface mode, transmit direction)*

Figures 5.11 and 5.12 show the composition and decomposition of the U_p frames. B1-, B2- and D-channel data is normally handled by the data flow controller. The HDLC controller as well as the PCM interface deliver data to the U_p interface and receive data from the U_p interface. For this reason, registers A_B1_TX, A_B2_TX, A_D_TX and A_BAC_S_TX as well as A_B1_RX, A_B2_RX, A_D_RX and A_E_RX [5] 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] Please note, that A_E_RX does only contain E-channel bits in S/T interface mode.

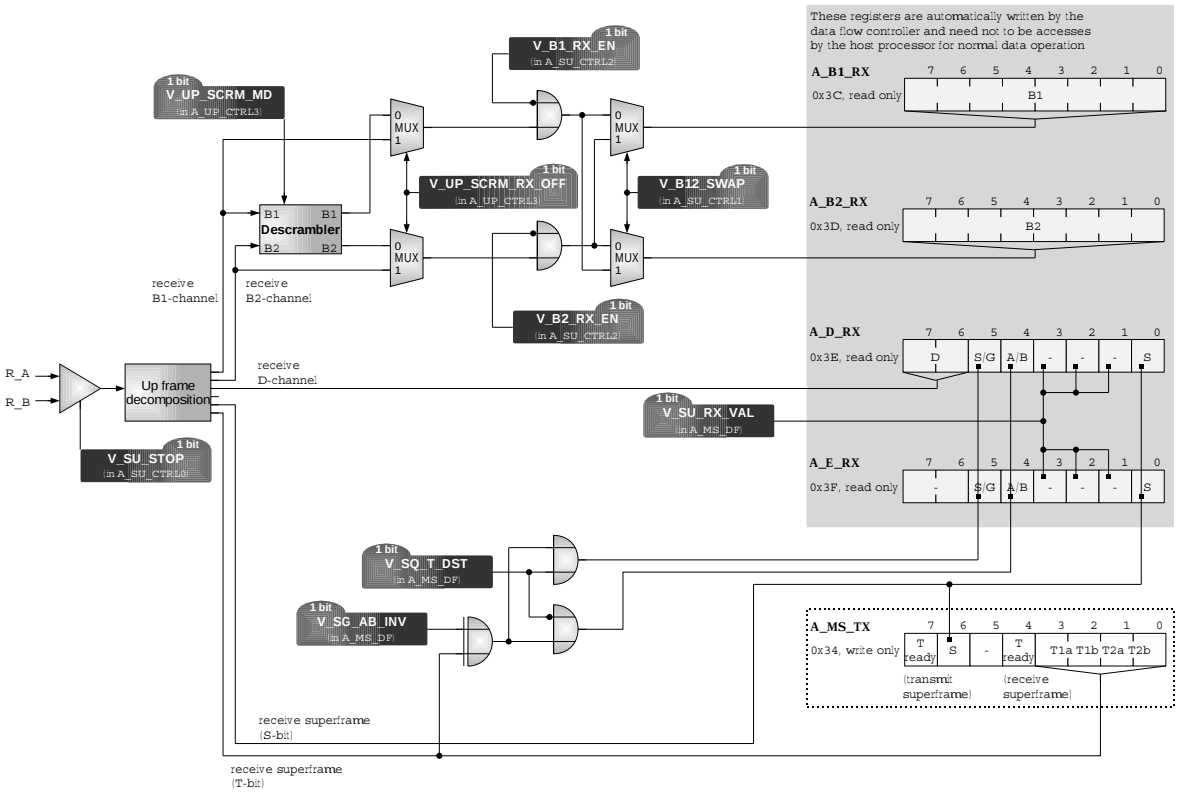


Figure 5.12: *U_p frame decomposition for B1-, B2-, D-channel and superframe bits (U_p interface mode, receive direction)*

5.3.5 INFO signals

Signals which are transmitted on the interface line are called *INFO signals*. INFO 0 is defined for upstream (TE-to-LT) and downstream (LT-to-TE) direction. All other INFO signals are either for upstream (INFO 1W, INFO 1, INFO 3) or downstream (INFO 2, INFO 4) direction. The INFO signals are defined as follows [3]:

INFO 0: No signal on line.

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

'F 00010001 00010001 0001 01010100 01011111 1'

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

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

'F 00010001 00010001 0001 01010100 01011111 M DC'

with code violation in the framing bit F.

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

'F 00010001 00010001 0001 01010100 01011111 M'

with code violation in the framing bit F.

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

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

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

5.3.6 State machine

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

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

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

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

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

State name:	Reset	Deactivated	Pending activation	Active	Pending deactivation
State number:	G 0	G 1	G 2	G 3	G 4
INFO sent:	INFO 0	INFO 0	INFO 2	INFO 4	INFO 0
Event:					
State machine release [*3]	G 1	|	|	|	|
Activate request	start T 1 [*1] G 2	start T 1 [*1] G 2	|	|	start T 1 [*1] G 2
Deactivate request	—	|	start T 2 G 4	start T 2 G 4	|
Expiry T 1 [*1]	—	—	start T 2 G 4	/	—
Expiry T 2 [*2]	—	—	—	—	G 1
Receiving INFO 0	—	—	—	G 2	G 1
Receiving INFO 1 or INFO 1W	—	start T 1 [*1] G 2	—	/	—
Receiving INFO 3	—	/	stop T 1 [*1,4] G 3	—	—
Lost framing	—	/	/	G 2	—

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

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

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

[*3]: After reset the state machine is fixed to G 0.

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

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

State name:	Reset	Sensing	Deactivated	Awaiting signal	Identifying input	Synchronized	Activated	Lost framing
State number:	F 0	F 2	F 3	F 4	F 5	F 6	F 7	F 8
INFO sent:	INFO 0	INFO 0	INFO 0	INFO 1W	INFO 0	INFO 1	INFO 3	INFO 0
Event:								
State machine release [*1]	F 2	/	/	/	/	/	/	/
Activate request,								
receiving any signal	—	|	F 5	|	|	—	|	—
receiving INFO 0	—	|	start T3 [*5] F 4	|	|	—	|	—
Expiry T 3 [*5]	—	/	—	F 3	F 3	—	—	F 3
Receiving INFO 0	—	F 3	—	—	—	F 3	F 3	F 3
Receiving any signal [*2]	—	—	—	F 5	—	/	/	—
Receiving INFO 2 [*3]	—	F 6	F 6	F 6	F 6	—	F 6	F 6
Receiving INFO 4 [*3]	—	F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	stop T3 [*5] F 7	—	stop T3 [*5] F 7
Lost framing [*4]	—	/	/	/	/	F 8	F 8	—

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

[*1]: After reset the state machine is fixed to F 0.

[*2]: This event reflects the case where a signal is received and the TE has not (yet) determined wether it is INFO 2 or INFO 4.

[*3]: Bit and frame synchronization achieved.

[*4]: Loss of Bit or frame synchronization.

[*5]: Timer T 3 is not implemented and must be implemented in software.

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

Important !

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

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

5.3.7 Clock synchronization

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

Received data from the pins R_A0 .. R_A3 and R_B0 .. R_B3 is passed through the RX data path to the switching buffer. A bit clock and a frame clock are derived from the received data steam. These clocks are used to synchronize the RX data path timing to the incoming data stream. The frame clock can be passed for synchronization purposes to the TX data path and the PCM timing control as well.

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

LT mode: The 384 kHz bit clock as well as the 8 kHz frame clock are derived from FSYNC in NT mode. This signal is either F0IO input or F1_7 (see register R_SL_SEL7 and Figure 6.5 on page 231).

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 pathes. The scramblers can be switched off with V_UP_SCRM_RX_OFF = '1' (for the receive data path) and V_UP_SCRM_TX_OFF = '1' (for the transmit data path) in register A_UP_CTRL3. Bit scramblers should only be disabled for test purposes. They shall mandatorily be enabled for normal U_p operation.

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

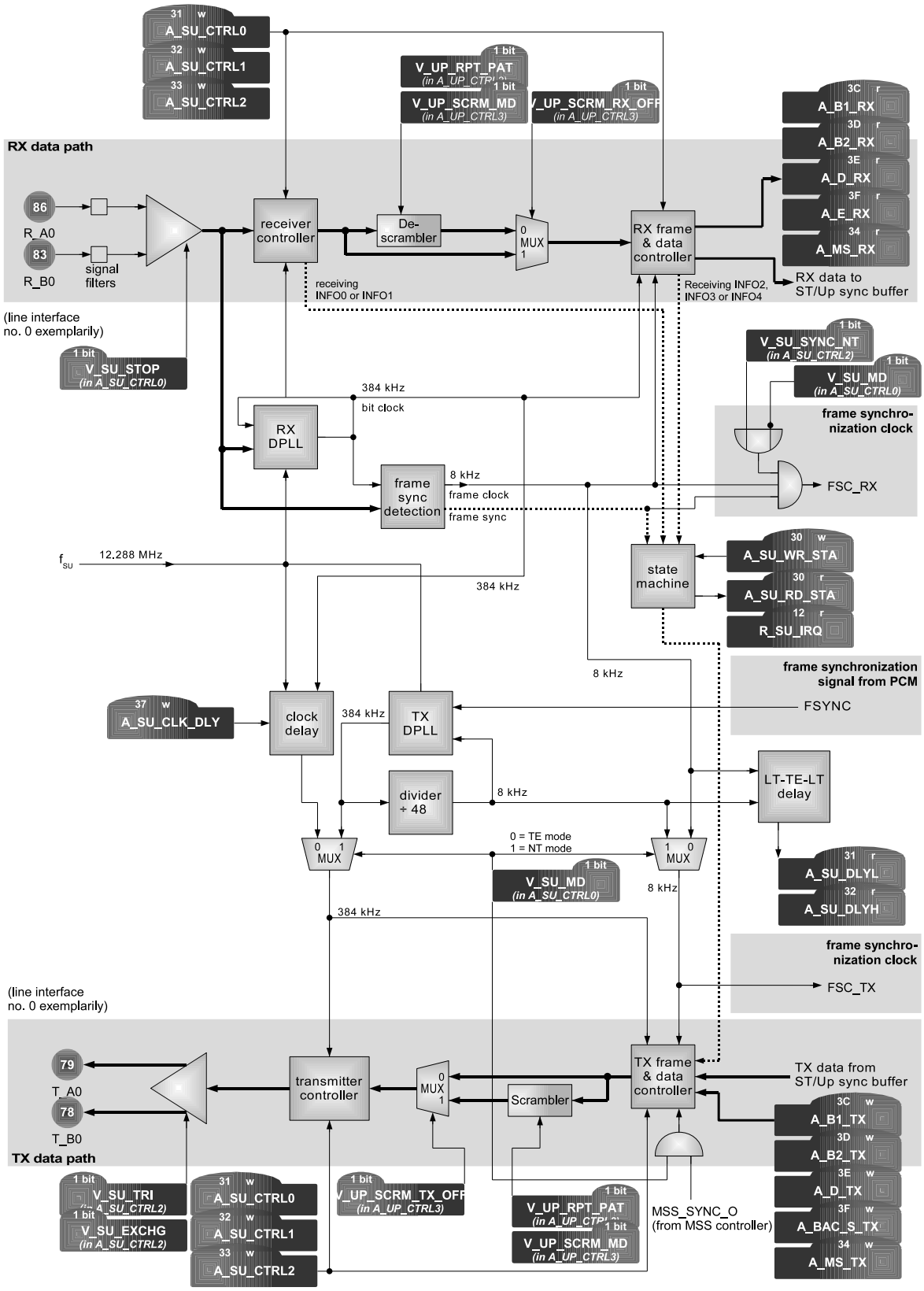


Figure 5.13: *U_p clock synchronization*

5.3.8 External circuitry

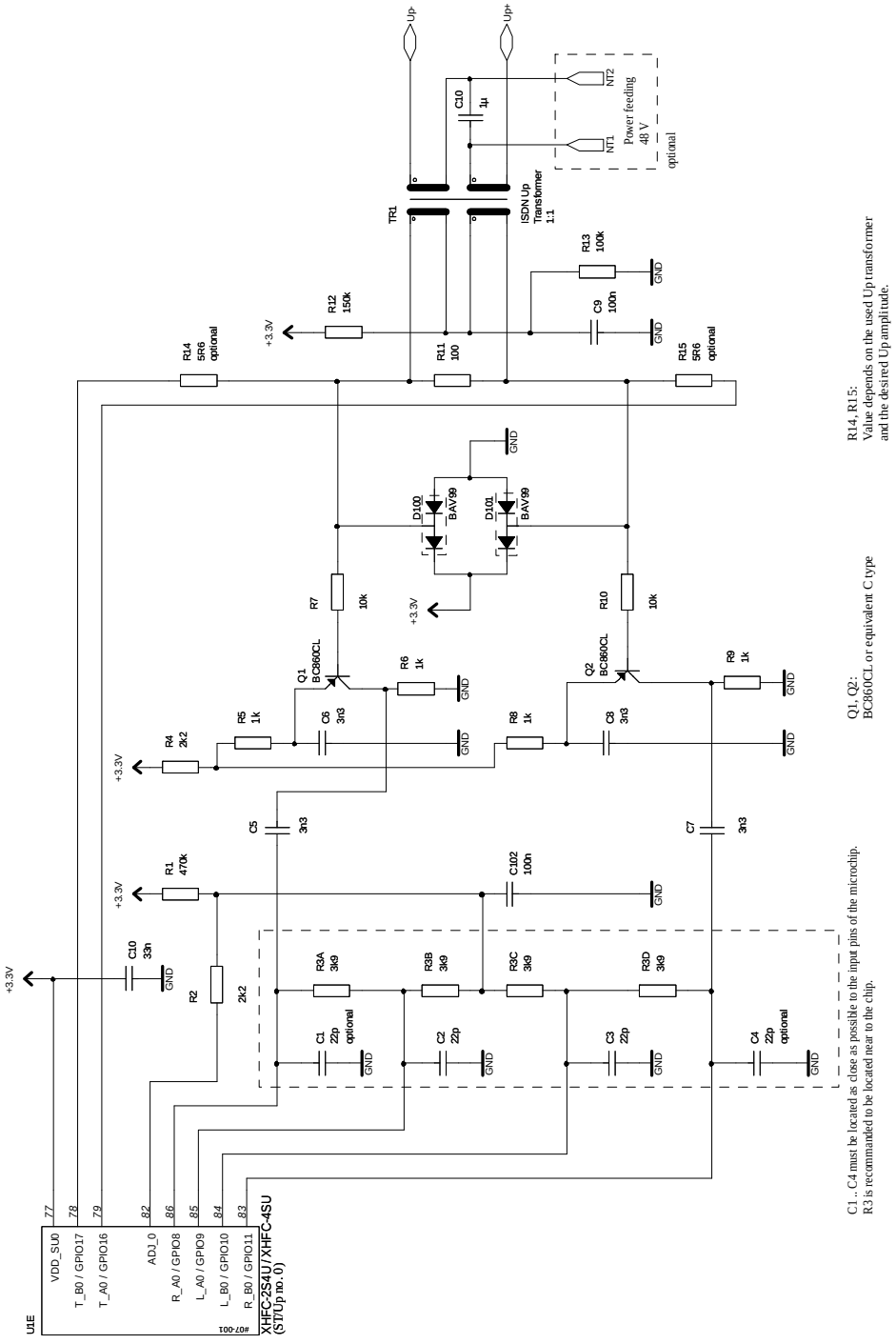


Figure 5.14: *External U_p circuitry for TE and LT mode*

5.3.9 U_p transformers

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

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

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

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

Table 5.11: *Up transformer part numbers and manufacturers*

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

Type	Device
Single transformer:	UT 21434A-TS (SMD)

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

Type	Device
Single transformer:	3-M5024-X008 (SMD)

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

Type	Device
Single transformer	503 10 903 00 (SMD)

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

5.4 Common features of the S/T and U_p interfaces

5.4.1 Direct data access for test purposes

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

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

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

Interrupt capability

Every line interface can cause an interrupt event when a state change occurs. State changes can be read from V_SU0_IRQ . . V_SU3_IRQ in register R_SU_IRQ for every line interface separately. All bits in R_SU_IRQ are cleared after reading the register.

Any '1' of these interrupt status bits cause an interrupt signal if the associated mask bit V_SU0_IRQMSK . . V_SU3_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_SU3_IRQMSK = '0000' 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

Several TEs of the XHFC-2S4U / 4SU 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 szenario is illustrated in Figure 5.15. XHFC-2S4U / 4SU is able to synchronize all line interfaces as it is described in this section.

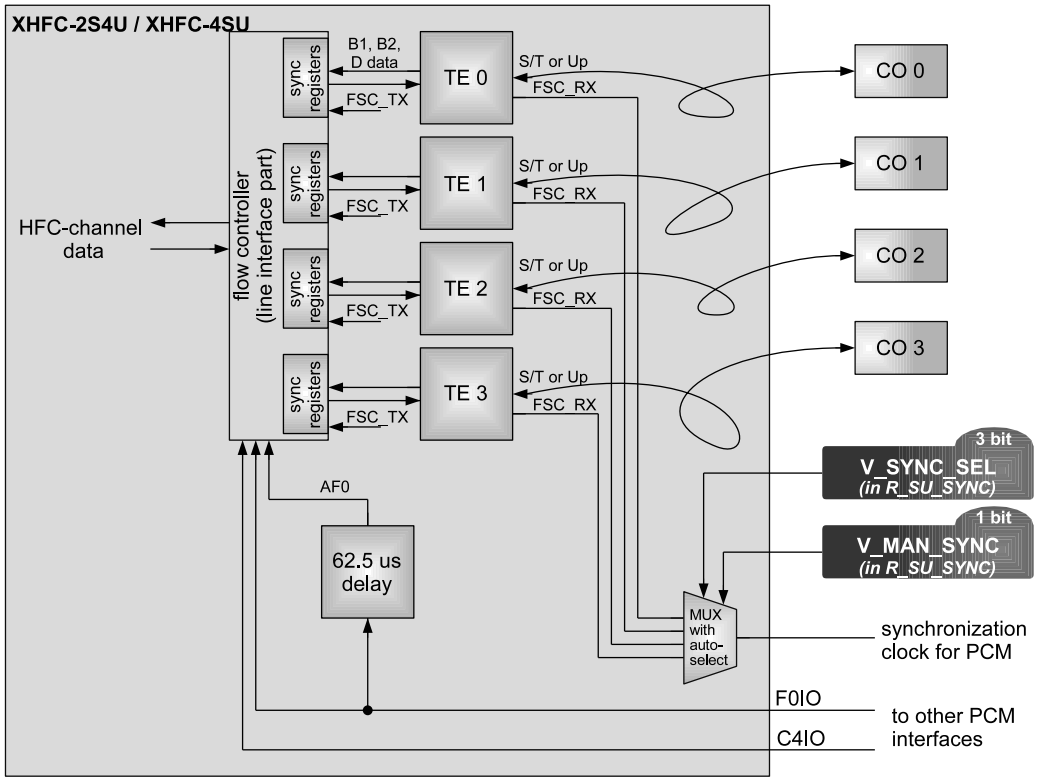


Figure 5.15: *Synchronization scenario with TEs connected to several central office switches*

The sychronization 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-2S4U / 4SU 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 it's output signal when the phase offset between FSC_TX and F0 is smaller than approximately 25 µs (guard window). So the phase offset between FSC_TX and F0 is always 25 µs . . 100 µs.

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

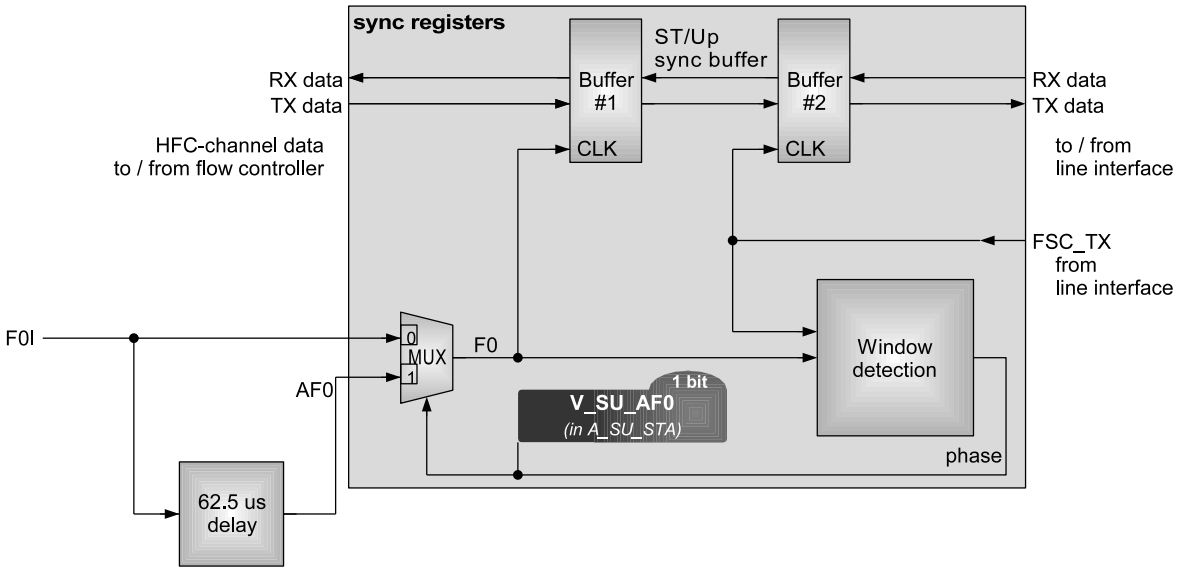


Figure 5.16: *Synchronization registers (detail of Figure 5.15)*

between FSC_TX_0 and F0IO is 62.5 µs (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 µs.

Table 5.12: *Symbols of Figures 5.17*

Symbol	Characteristic
t_{PLL}	PLL-generated frame pulse offset between FSC_TX_0 and F0IO (62.5 μs)
t_{delay}	Frame pulse delay between F0IO and AF0 (62.5 μs)
t_{phase}	Frame offset between interface TE 0 and TE 1
$TX_{data\_F0\_0}$	Time from transmit data valid to next F0_0 pulse
$TX_{F0\_0\_FSC0}$	Time from F0_0 pulse to next FSC_TX_0 pulse
$RX_{FSC0\_F0\_0}$	Time from FSC_TX_0 pulse to next F0_0 pulse
$RX_{F0\_0\_data}$	Time from F0_0 pulse to receive data valid
$TX_{data\_F0\_1}$	Time from transmit data valid to next F0_1 pulse
$TX_{F0\_1\_FSC1}$	Time from F0_1 pulse to next FSC_TX_1 pulse
$RX_{FSC1\_F0\_1}$	Time from FSC_TX_1 pulse to next F0_1 pulse
$RX_{F0\_1\_data}$	Time from F0_1 pulse to receive data valid

Figure 5.17 is divided into three parts. The upper and lower part show the line interface oriented signals of TE 0 and TE 1 respectively. In the middle part, HFC-channel oriented signals are shown 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 µs. AF0 has a fixed 62.5 µs 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

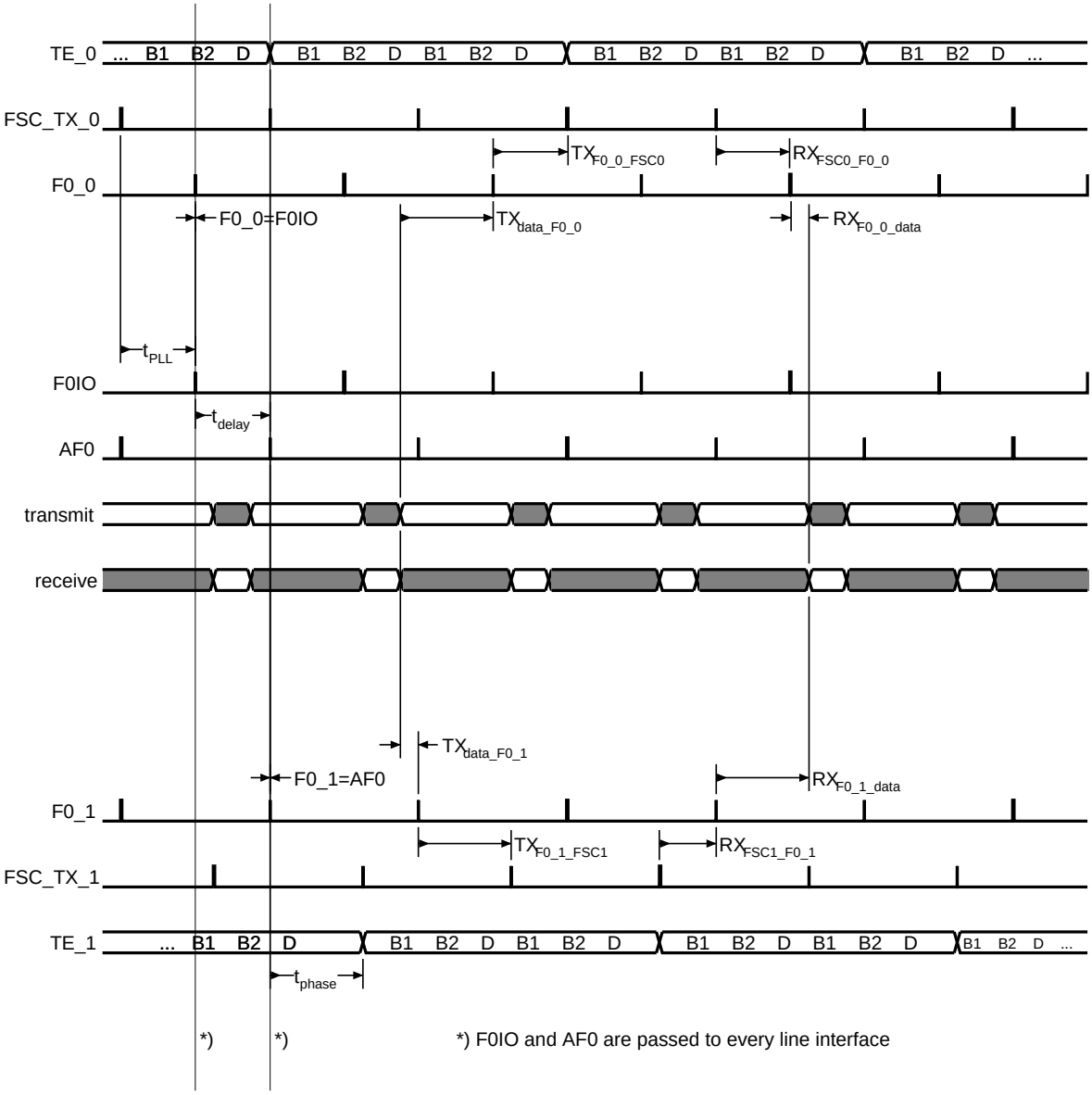


Figure 5.17: *Transmit and receive data transmission examples of two TEs with phase offset (see explanation in the text)*

#1 in transmit direction and output data from buffer #1 in receive direction.

As TE 0 is the synchronization source in this example, FSC_TX_0 is the reference signal for the PLL to generate the F0IO signal. A data transmission from 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_TX_0 pulse.

In receive direction, the incoming data byte is stored in buffer #2 first with the FSC_TX_0 pulse. After $RX_{\text{FSC0\_F0\_0}}$, 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 $TX_{\text{data\_F0\_1}}$ after data became valid. The internal data transfer of each ST / U_p interface is controlled either by F0IO or by AF0. In this example F0_1 = AF0 is shown. $TX_{\text{F0\_1\_FSC1}}$ after F0_1, the FSC_TX_1 pulse stores the data in buffer #2 so that the data byte is available at the line interface.

Received data is first stored in buffer #2 with the FSC_TX_1 pulse. After $RX_{\text{FSC1\_F0\_1}}$ buffer #1 takes the data byte and it receives the HFC-channel.

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

Timing with frequency drift

When there is a frequency drift between FSC_TX_0 and FSC_TX_1, the window detection block changes 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 $f_{\text{FSC\_TX\_1}} > f_{\text{FSC\_TX\_0}}$ in transmit data direction. FSC_TX_0 is assumed to be the synchronization signal which is the source for F0IO . F0_1 is either F0IO or AF0. In this case FSC_TX_1 is too fast which leads to a *byte doubling* in case of transmission error.

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

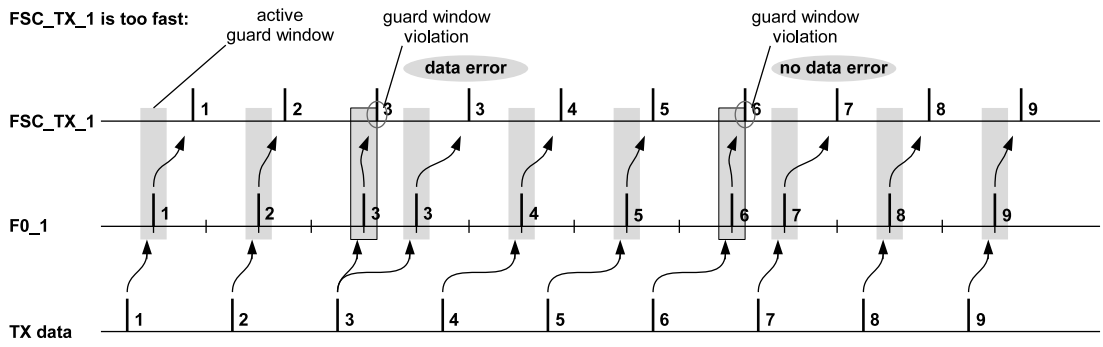


Figure 5.18: *Data transmission with $f_{\text{FSC\_TX\_1}} > f_{\text{FSC\_TX\_0}}$ (i.e. too fast FSC_TX from unsynchronized TE)*

Figure 5.19 shows the synchronization process for $f_{\text{FSC\_TX\_1}} < f_{\text{FSC\_TX\_0}}$ in transmit data direction. Again, FSC_TX_0 is assumed to be the synchronization signal. In this case FSC_TX_1 is too slow which leads to a *byte skip* in case of transmission error.

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

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

The time between two errors is given by

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

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

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

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.

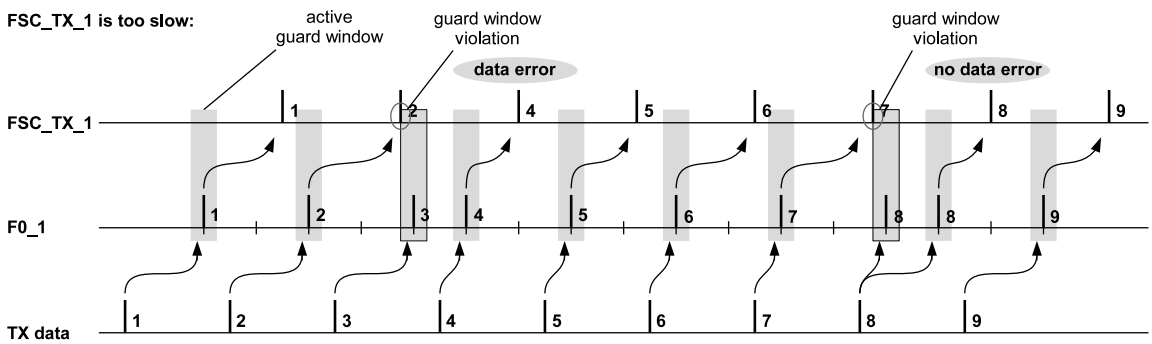


Figure 5.19: *Data transmission with $f_{FSC\_TX\_1} < f_{FSC\_TX\_0}$ (i.e. too slow FSC_TX from unsynchronized TE)*

5.4.3 Combined S/T and U_p circuitry

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

5.5 Register description

> **Please note !**
>
> The name fragment SU of registers and bitmaps indicates those registers and bitmaps which are valid in both S/T and U_p interface mode. SU means 'ST / U_p'.
>
> The name fragments ST or UP are used when a register or bitmap is either valid in S/T or U_p interface mode.

5.5.1 Write only registers

R_SU_SEL (w) (Reset group: H, 0, 3) **0x16**

S/T or Up interface selection register

This register is used to select an S/T or Up interface. Before a line interface array register can be accessed, this index register must specify the desired line interface number.

Bits	Reset value	Name	Description
1 . . 0	0	**V_SU_SEL**	**Single line interface selection** '000' = ST/Up interface 0 '001' = ST/Up interface 1 '010' = ST/Up interface 2 '011' = ST/Up interface 3
2	0	(reserved)	Must be '0'.
3	0	**V_MULT_SU**	**multiple line interface selection** All line interfaces are selected together. This is only useful for write access. '0' = interface selection by V_SU_SEL '1' = select all line interfaces for write accesses
7 . . 4	0	(reserved)	Must be '0000'.

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

ST / Up state machine register

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

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

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

A_SU_CTRL0 [ST/Up] **(w)** (Reset group: H, 0, 3) **0x31**

Control register of the selected line interface, register 0

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

Bits	Reset value	Name	Description
0	0	**V_B1_TX_EN**	**B1-channel transmit** '0' = B1 send data disabled (permanent '1's sent when the line interface is activated) '1' = B1 send data enabled
1	0	**V_B2_TX_EN**	**B2-channel transmit** '0' = B2 send data disabled (permanent '1's sent when the line interface is activated) '1' = B2 send data enabled
2	0	**V_SU_MD**	**Line interface mode** '0' = TE mode '1' = NT/LT mode
3	0	**V_ST_D_LPRIO**	**D-channel priority** '0' = high priority 8/9 '1' = low priority 10/11 **Note:** This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
4	0	**V_ST_SQ_EN**	**S/Q bits transmission** '0' = S/Q bits disabled '1' = S/Q bits (multiframe) enabled **Note:** This bit is only used for line interfaces in S/T mode. It is ignored in Up interface mode.
5	0	**V_SU_TST_SIG**	**Send test signal** '0' = normal operation '1' = send test signal The test signal depends on the selected line interface mode and V_SU_2KHZ in register A_SU_CTRL2: **S/T interface mode:** Test signal is 96 kHz (alternating '0's) or 2 kHz (one alternating '0' per frame) **Up interface mode:** Test signal is 192 kHz (alternating '1's) or 2 kHz (one alternating '1' per frame)

(continued on next page)

(continued from previous page)

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

A_SU_CTRL1 [ST/Up] **(w)** (Reset group: H, 0, 3) **0x32**

Control register of the selected line interface, register 1

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

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

A_SU_CTRL2 [ST/Up] **(w)** (Reset group: H, 0, 3) **0x33**

Control register of the selected line interface, register 2

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

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

(continued on next page)

(continued from previous page)

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

A_MS_TX [ST/Up]	**(w)**	(Reset group: H, 0, 3)	**0x34**

Multiframe / superframe transmit register

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

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

A_ST_CTRL3 [ST/Up]	**(w)**	(Reset group: H, 0, 3)	**0x35**

Control register of the selected line interface, register 3

This register is an array register *and* a multi-register.
Before writing this array register the line interface must be selected by register R_SU_SEL. Afterwards the multi-register selection is required with bit V_ST_SEL = '0' in this register.

Note: 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.

Bits	Reset value	Name	Description
0	0	**V_ST_SEL**	**Line interface mode selection** This bit selects either the S/T or Up mode of the line interface. '0' = line interface is in S/T mode and multi-register A_ST_CTRL3 is selected '1' = line interface is in Up mode and multi-register A_UP_CTRL3 is selected **Note:** This bit should only be changed after reset.
7..1	0	**V_ST_PULSE**	**End of pulse control** 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)**	(Reset group: H, 0, 3)	**0x35**

Control register of the selected line interface, register 3

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

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

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

(continued on next page)

(continued from previous page)

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

A_MS_DF [ST/Up]	**(w)**	(Reset group: H, 0, 3)	**0x36**

Multiframe / superframe data flow configuration register

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

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

A_SU_CLK_DLY [ST/Up]	**(w)**	(Reset group: H, 0, 3)	**0x37**

Clock control register of the line interface module

This register is not initialized after reset. It must be initialized before activating the ST / Up state machine.

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

Bits	Reset value	Name	Description
3..0	0	**V_SU_CLK_DLY**	**Line interface clock delay** **TE mode (S/T or Up):** 4 bit delay value to adjust the 2 bit delay between receive and transmit frame. The delay of the external line interface circuitry can be compensated. The lower the value the smaller the delay between receive and transmit direction. The suitable value is 0xE (S/T interface mode) or 0xF (Up interface mode) for normal external circuitries. **NT mode (for S/T interface mode only):** Data sample point. The lower the value the earlier the input data is sampled. The normal operation value is 0xC. **LT mode (for Up interface mode only):** This bitmap is not used when the line interface operates in Up mode. The steps are 163 ns for line interfaces in S/T mode and 81 ns for line interfaces in Up mode.
6..4	0	**V_ST_SMPL**	**Early edge input data shaping** (NT mode only) Low pass characteristic of extended bus configurations can be compensated. The lower the value the earlier input data pulses are sampled. The default value is 6 (’110’) which means that no compensation is carried out. Step size is 163 ns. **Note:** This bit is only used for line interfaces in S/T mode. It must be ’0’ in Up interface mode.
7	0	(reserved)	Must be ’0’.

A_B1_TX [ST/Up]	(w)	(Reset group: –)	**0x3C**

Transmit register for the B1-channel data

This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.

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

Bits	Reset value	Name	Description
7..0		**V_B1_TX**	**B1-channel data byte** Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

A_B2_TX [ST/Up]	(w)	(Reset group: –)	**0x3D**

Transmit register for the B2-channel data

This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.

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

Bits	Reset value	Name	Description
7..0		**V_B2_TX**	**B2-channel data byte** Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

A_D_TX [ST/Up]	(**w**)	(Reset group: –)	**0x3E**

Transmit register for the D-channel data

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_D_TX_S**	**S-bit of the Up superframe** The transmitted S-bit of the Up superframe is either taken from V_D_TX_S, V_S_TX or V_MS_TX. When the line interface is in S/T mode, this bit can be used as multiframing bit M.
4 . . 1		(reserved)	Must be '0000' when written.
5		**V_D_TX_BAC**	**BAC-bit** The BAC-bit can be used as S/Q-bit (S/T multiframe) or T-bit (Up superframe). Alternatively, bitmap V_MS_TX in register A_MS_TX can be used for the multiframe / superframe transmission.
7 . . 6		**V_D_TX**	**D-channel data bits**

A_BAC_S_TX [ST/Up]	(**w**)	(Reset group: –)	**0x3F**

BAC-bit and S-bit for multiframe / superframe transmission

This register is written automatically by the flow controller and need not be accessed by the user. FIFOs should be used to write data instead.

Before writing this array register the line interface must be selected by register R_SU_SEL. Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

Bits	Reset value	Name	Description
0		**V_S_TX**	**S-bit of the Up superframe** The transmitted S-bit of the Up superframe is either be taken from V_D_TX_S, V_S_TX or V_D_TX_S. When the line interface is in S/T mode, this bit can be used as multiframing M-bit.
4 . . 1		(reserved)	Must be ’0000’ when written.
5		**V_BAC_TX**	**BAC-bit** The BAC-bit can be used as S/Q-bit (S/T multiframe) or T-bit (Up superframe). Alternatively, bitmap V_MS_TX in register A_MS_TX can be used for the multiframe / superframe transmission.
7 . . 6		(reserved)	Must be ’00’ when written.

5.5.2 Read only registers

R_AF0_OVIEW	(r)	(Reset group: H, 0, 3)	0x13

Alternate frame synchronization signal overview register

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).

Bits	Reset value	Name	Description
0	0	**V_SU0_AF0**	**FSC selection for line interface 0** This bit is equal to A_SU_STA with line interface 0 being selected.
1	0	**V_SU1_AF0**	**FSC selection for line interface 1** This bit is equal to A_SU_STA with line interface 1 being selected.
2	0	**V_SU2_AF0**	**FSC selection for line interface 2** This bit is equal to A_SU_STA with line interface 2 being selected.
3	0	**V_SU3_AF0**	**FSC selection for line interface 3** This bit is equal to A_SU_STA with line interface 3 being selected.
7..4		(reserved)	

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

ST / Up state machine register

This register is used to read the current state. A new state can be set with register A_SU_WR_STA.

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

Bits	Reset value	Name	Description
3..0	0	**V_SU_STA**	**ST / Up state** Binary value of current state (NT/LT: Gx, TE: Fx)
4	0	**V_SU_FR_SYNC**	**Frame synchronization** '0' = not synchronized '1' = synchronized
5	0	**V_SU_T2_EXP**	**Timer T2 expired** '1' = timer T2 expired (NT/LT mode only)
6	0	**V_SU_INFO0**	**INFO 0** '1' = receiving INFO 0
7	0	**V_G2_G3**	**G2 to G3 transition allowed** '0' = no operation '1' = allows transition from G2 to G3 in NT/LT mode This bit is automatically cleared after the transition and has no function in TE mode.

A_SU_DLYL [ST/Up]	**(r)**	(Reset group: –)	**0x31**

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

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

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

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

In both line interface modes, the round trip delay is measured within a 10 bit range. The lower 5 bits can be read from this register, the upper 5 bits are stored in register A_SU_DLYH.

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

Bits	Reset value	Name	Description
4..0		**V_SU_DLYL**	**Lower part of the round trip delay** The lower part of the delay time $t_{dly,l} = t_0 \cdot$ V_SU_DLYL. Register A_SU_DLYH must also be read to obtain the whole delay $t_{dly} = t_{dly,l} + t_{dly,h}$. **S/T interface mode:** 0 = delay $t_{dly,l} = 0\,\text{ns}$ 1 = delay $t_{dly,l} = 162.8\,\text{ns}$ 2 = delay $t_{dly,l} = 325.5\,\text{ns}$ 3 = delay $t_{dly,l} = 488.3\,\text{ns}$ ... 31 = delay $t_{dly,l} = 5045.6\,\text{ns}$ **Up interface mode:** 0 = delay $t_{dly,l} = 0\,\text{ns}$ 1 = delay $t_{dly,l} = 81.4\,\text{ns}$ 2 = delay $t_{dly,l} = 162.8\,\text{ns}$ 3 = delay $t_{dly,l} = 244.1\,\text{ns}$ ... 31 = delay $t_{dly,l} = 2522.8\,\text{ns}$
7..5		(reserved)	

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

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

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

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

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

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

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

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

A_MS_RX [ST/Up] **(r)** (Reset group: H, 0, 3) **0x34**

Multiframe / superframe receive register

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

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

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

Status register of the line interface

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

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

Bits	Reset value	Name	Description
0	0	**V_ST_D_HPRIO9**	**Current D-channel priority state if high priority is selected** 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. '0' = 8 bits '1' '1' = 9 bits '1' **Note:** This bit is only valid in S/T interface mode.
1	0	**V_ST_D_LPRIO11**	**Current D-channel priority state if low priority is selected** 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. '0' = 10 bits '1' '1' = 11 bits '1' **Note:** This bit is only valid in S/T interface mode.
2	0	**V_ST_D_CONT**	**D-channel contention** This bit reports a difference between D- and E-channel. '0' = no contention occured '1' = contention occured in the last frame **Note:** This bit is only valid in S/T interface mode.
3	0	**V_ST_D_ACT**	**D-channel active** This bit has the value '1' when the D-channel is just transmitting data. **Note:** This bit is only valid in S/T interface mode.
6..4		(reserved)	

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	**V_SU_AF0**	**FSC selection** 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 This bit is also available in the overview register R_AF0_OVIEW.

A_B1_RX [ST/Up] **(r)** (Reset group: –) **0x3C**

Receive register for the B1-channel data

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

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

Bits	Reset value	Name	Description
7..0		**V_B1_RX**	**B1-channel data byte** In activated state data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS). Register value is 0xFF in deactivated state.

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

Receive register of the B2-channel data

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

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

Bits	Reset value	Name	Description
7..0		**V_B2_RX**	**B2-channel data byte** In activated state data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS). Register value is 0xFF in deactivated state.

A_D_RX [ST/Up]	**(r)**	(Reset group: –)	**0x3E**

Receive register for the D-channel data

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

Before reading this array register the line interface must be selected by register R_SU_SEL. Data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).

Bits	Reset value	Name	Description
0		**V_D_RX_S**	**S-bit of the Up superframe** Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_E_RX and A_MS_RX. **Note:** This bit is only used for line interfaces in Up mode. It is undefined in S/T interface mode.
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**	**A/B-bit** When V_SQ_T_DST = '1', V_D_RX_AB contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_D_RX_AB is constant '1' when V_SQ_T_DST = '0'. The A/B-bit can also be read from register A_E_RX.
5		**V_D_RX_SG**	**S/G-bit** When V_SQ_T_DST = '0', V_D_RX_SG contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_D_RX_SG is constant '1' when V_SQ_T_DST = '1'. The S/G-bit can also be read from register A_E_RX.
7 . . 6		**V_D_RX**	**D-channel data bits** Data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).

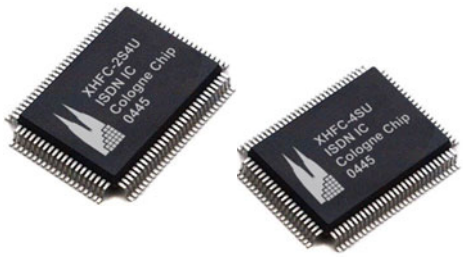
A_E_RX [ST/Up]	**(r)**	(Reset group: –)	**0x3F**

Receive register for the E-channel data

This register is read automatically by the flow controller and need not be accessed by the user. FIFOs should be used to read data instead.

Before reading this array register the line interface must be selected by register R_SU_SEL. Data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).

Bits	Reset value	Name	Description
0		**V_E_RX_S**	**S-bit of the Up superframe** Last received S-bit of the superframe (M-channel). The S-bit is also stored in registers A_D_RX and A_MS_RX. **Note:** This bit is only used for line interfaces in Up mode. It is undefined in S/T interface mode.
3 . . 1		(reserved)	A read access to these unused bits returns the value of V_SU_RX_VAL in register A_MS_DF.
4		**V_E_RX_AB**	**A/B-bit** When V_SQ_T_DST = '1', V_E_RX_AB contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_E_RX_AB is constant '1' when V_SQ_T_DST = '0'. The A/B-bit can also be read from register A_D_RX.
5		**V_E_RX_SG**	**S/G-bit** When V_SQ_T_DST = '0', V_E_RX_SG contains either the last received S/Q-bit of the S/T multiframe or the last received T-bit of the Up superframe. V_E_RX_SG is constant '1' when V_SQ_T_DST = '1'. The S/G-bit can also be read from register A_D_RX.
7 . . 6		**V_E_RX**	**E-channel data bits** Data is valid after a F0IO or AF0 pulse and can be read during the non-processing phase (see V_PROC in register R_STATUS).



Chapter 6

PCM interface

Table 6.1: *Overview of the XHFC-2S4U / 4SU PCM interface registers*

Write only registers:

Address	Name	Page
0x10	R_SLOT	255
0x14	R_PCM_MD0	256
0x15	R_SL_SEL0	257
0x15	R_SL_SEL1	258
0x15	R_SL_SEL7	258
0x15	R_MSS0	259
0x15	R_PCM_MD1	261
0x15	R_PCM_MD2	263
0x15	R_MSS1	265
0x15	R_SH0L	266
0x15	R_SH0H	266
0x15	R_SH1L	266
0x15	R_SH1H	267
0x17	R_SU_SYNC	268
0x28	R_CI_TX	269
0x29	R_GCI_CFG0	270
0x2A	R_GCI_CFG1	272
0x2B	R_MON_TX	272

Read only registers:

Address	Name	Page
0x18	R_F0_CNTL	273
0x19	R_F0_CNTH	273
0x1D	R_SL_MAX	273
0x28	R_CI_RX	274
0x29	R_GCI_STA	275
0x2A	R_MON_RX	275

Read / write registers:

Address	Name	Page
0xD0	A_SL_CFG	276

6.1 PCM interface function

XHFC-2S4U / 4SU 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

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

is allowed and leads to n PCM time slots. F0IO must always have a frequency of 8 kHz.

XHFC-2S4U / 4SU 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.

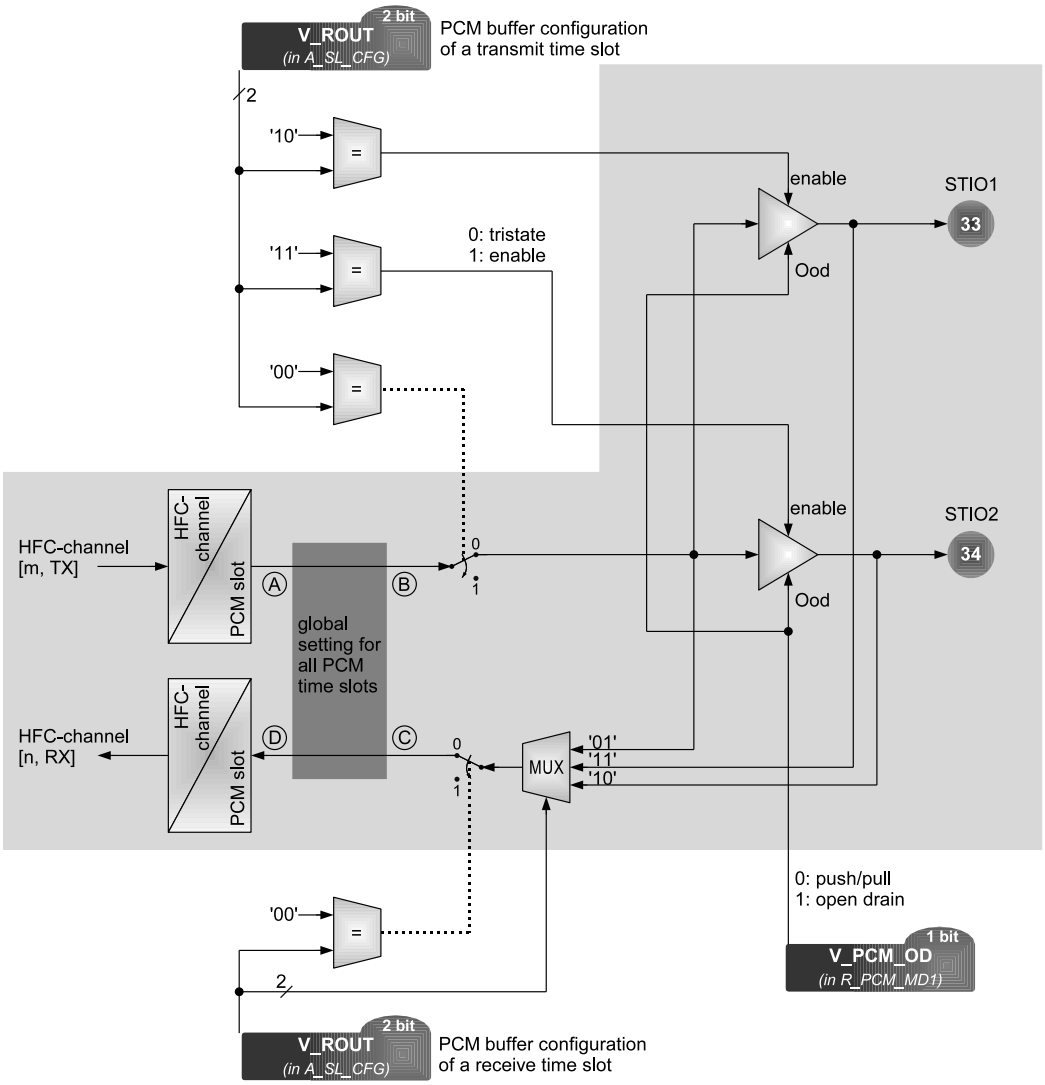


Figure 6.1: *PCM data flow for transmit and receive time slots (see Figure 6.2 for additional setting of all PCM time slots between Ⓐ..Ⓓ)*

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.

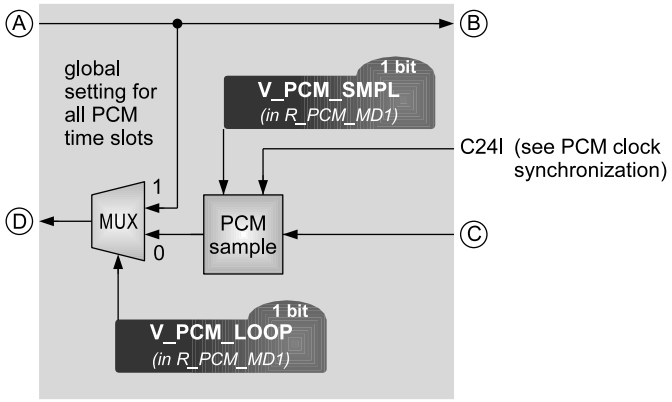


Figure 6.2: *Global setting for all PCM time slots to set up an internal PCM loop (detail of Figure 6.1), normally used for test loop setup*

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 avaliable between Ⓐ .. Ⓓ as shown in Figure 6.2. When V_PCM_LOOP = '1' in register R_PCM_MD1, the PCM data is looped internally.

Please note, that it is not allowed to set V_PCM_OD = '1' in register R_PCM_MD1 when an internal PCM loop is activated with either V_ROUT = '01' in register A_SL_CFG or V_PCM_LOOP = '1' in register R_PCM_MD1.

6.3 PCM initialization

After hardware, global software or PCM reset the PCM interface starts an initialization sequence to set all A_SL_CFG registers of the PCM time slots to the reset value 0. This is even done if no valid C4IO and F0IO signals exist, which might occur in PCM slave mode. C4IO and F0IO input signals are ignored during PCM initialization.

The initialization process is indicated with V_PCM_INIT = '1' in register R_STATUS. This bit changes to '0' when the initialization sequence is finished.

6.4 PCM timing

6.4.1 Mode selection

The PCM interface of XHFC-2S4U / 4SU can operate either in slave mode or master mode. Slave mode is the default selection after XHFC-2S4U / 4SU reset.

To configure XHFC-2S4U / 4SU 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-2S4U / 4SU in this case and both pins have output characteristic.

Slave mode is selected with V_PCM_MD = '0'. C4IO and F0IO are input pins in slave mode. External clocks must be connected to F0IO and also to either C4IO (when V_C2I_EN = '0' in register R_PCM_MD2) or C2IO (when V_C2I_EN = '1') in PCM slave mode because these signals are used from the ST / U_p interface and the flow controller as well (see Figure 6.5 on page 231, signal C24I is required).

6.4.2 Master mode

Figure 6.3 shows the timing diagram for PCM master mode. The timing characteristics are specified in Table 6.4. STIO1 is shown as data output and STIO2 as data input of XHFC-2S4U / 4SU. 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 240).

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-2S4U / 4SU. However, both pins can change their I/O characteristic with every PCM time slot.

The F0IO pulse is expected to be one C4IO pulse long with the default value V_F0_LEN = '0' in register R_PCM_MD0. F0IO is expected to start one C4IO clock earlier if bit V_F0_LEN = '1'.

If the ST / U_p interfaces are synchronized from C4IO in NT / LT mode, the frequency stability must be at least $\pm 10^{-4}$.

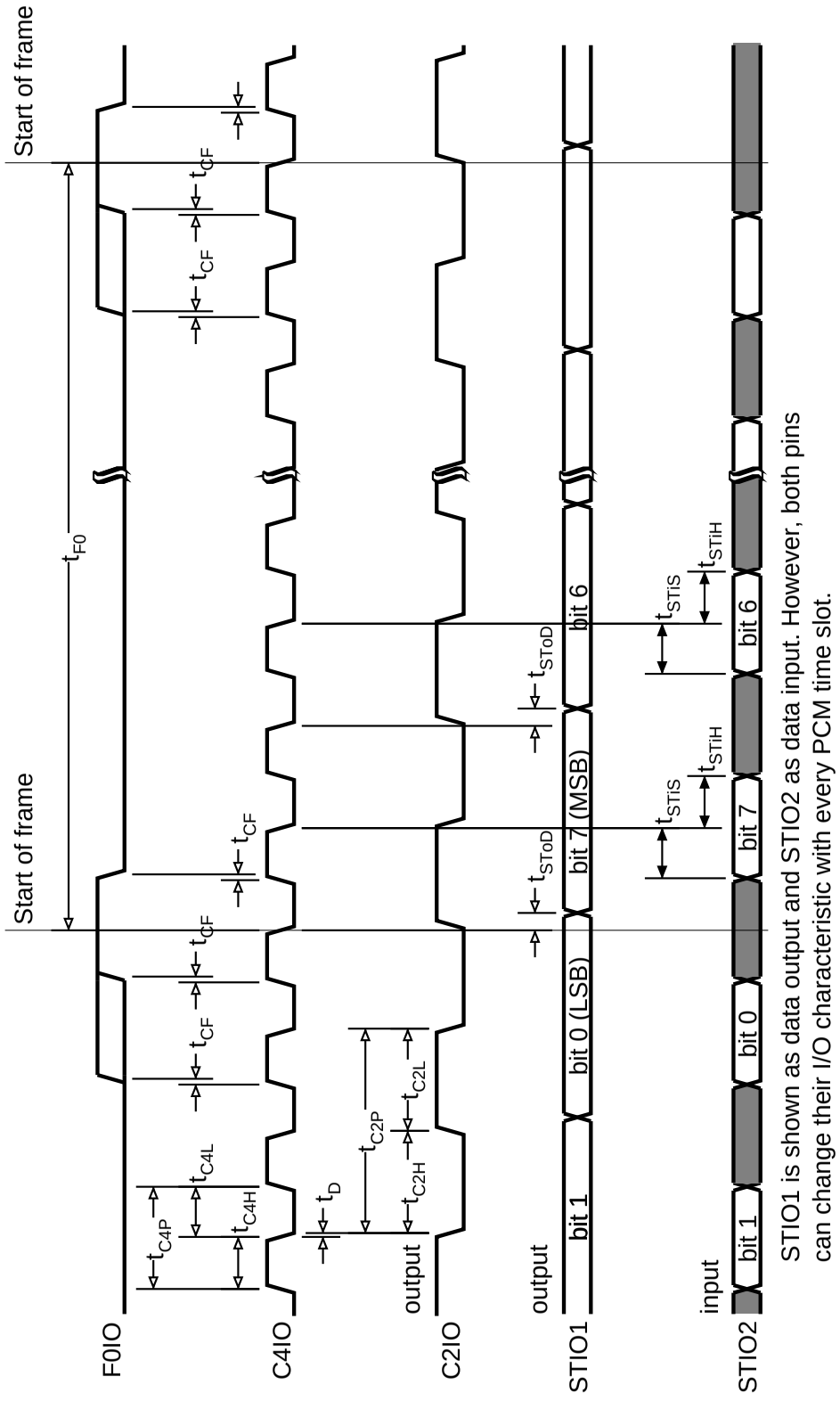


Figure 6.3: *PCM timing for master mode*

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				Basic C4IO pulse width (not shown in the timing diagram)
		122.070		4.096 MHz C4IO clock for 2 MB/s
		61.035		8.192 MHz C4IO clock for 4 MB/s
		30.518		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 ⎿ to C2IO ⌜ delay
t_{CF}	0.5		8	C4IO ⌜ to F0IO ⌜ or C4IO ⌜ to F0IO ⎿
t_{STiS}	10			Data valid to C4IO ⎿ setup time
t_{STiH}	10			Data valid to C4IO ⎿ hold time
t_{SToD}	2		10	STIO output delay from C4IO ⎿

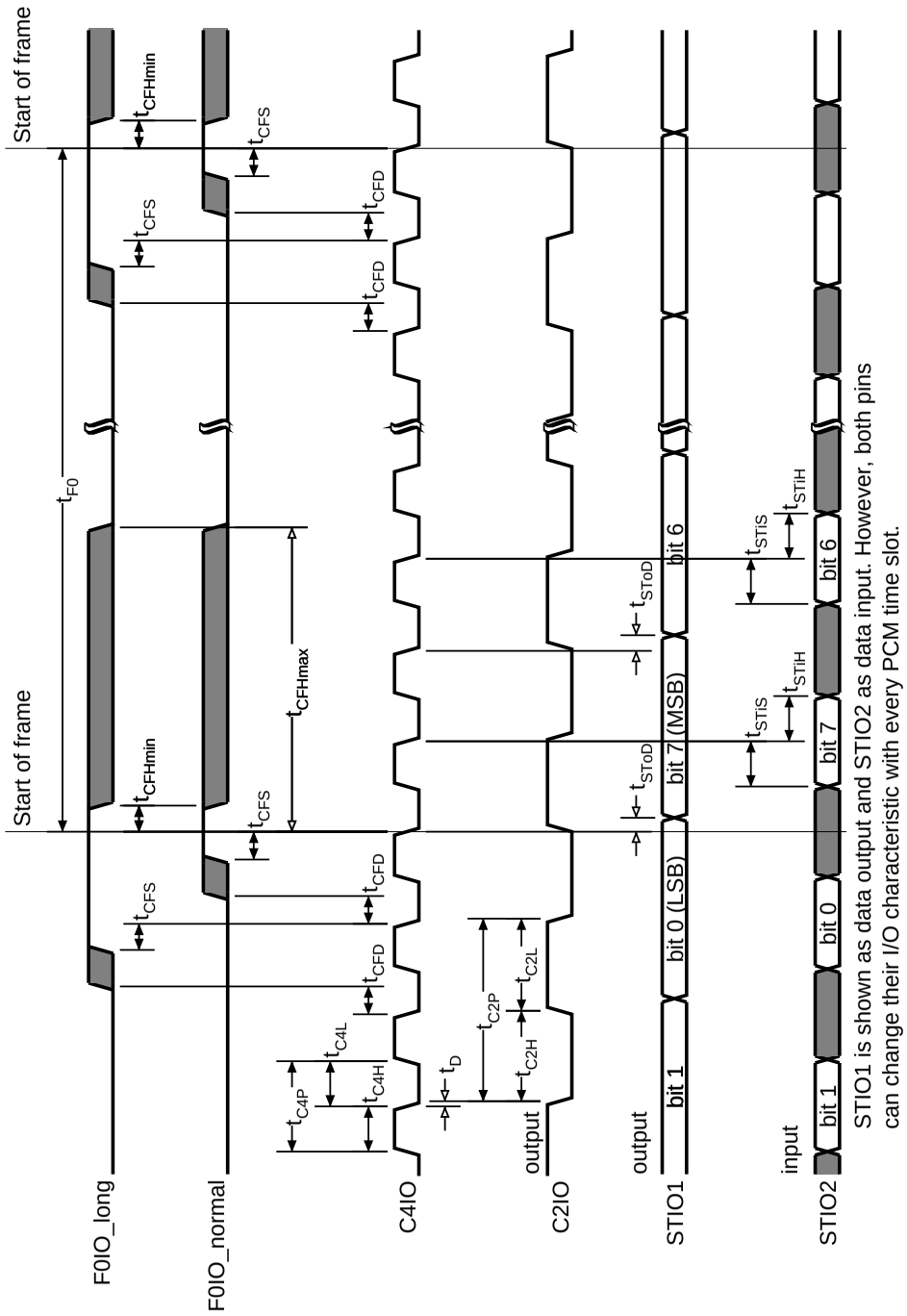


Figure 6.4: *PCM timing for slave mode*

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				Basic C4IO pulse width (not shown in the timing diagram)
		122.070		4.096 MHz C4IO clock for 2 MB/s
		61.035		8.192 MHz C4IO clock for 4 MB/s
		30.518		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 ↓ to C2IO ↑ delay
t_{CFS}	15	t_C		F0IO ↑ to C4IO ↓ setup time
t_{CFHmin}	15	t_C		F0IO ↓ to C4IO ↓ hold time
t_{CFHmax}	15	t_C	100000	F0IO high time after start of frame
t_{CFD}	15	t_C		C4IO ↓ to F0IO ↑ delay
t_{STiS}	10			Data valid to C4IO ↓ setup time
t_{STiH}	10			Data valid to C4IO ↓ hold time
t_{SToD}	2		10	STIO output delay from C4IO ↓

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 169) and 5.13 (page 185) and with the MSS controller (shown in Figure 6.10 on page 238).

6.5.2 Manual or automatic synchronization source selection

Any line interface in TE mode can be chosen as synchronization source with an appropriate value in bitmap V_SYNC_SEL of register R_SU_SYNC. Alternatively, an automatic selection can be enabled with V_MAN_SYNC = '0' in the same register (reset default). The actual synchronization source can be read from V_RD_SYNC_SRC in register R_BERT_STA. If synchronization is lost on this TE, the next interface in TE mode with active synchronization is automatically selected.

When the automatic selection doesn't find a synchronization source, the SYNC_I pulse can alternatively be taken as synchronization source. This must be enabled with V_AUTO_SYNCI = '1' in register R_SU_SYNC.

6.5.3 PLL programming for F0IO generation

C4IO is adjusted from the PCM DPLL (see Figure 6.5) during the last PCM time slot to synchronize the PCM interface with the ST / U_p interface. [1] The maximum number of edge adjustments during one 125 µs cycle can be configured in the range 1 . . 4 by the bitmap value V_PLL_ADJ in register R_PCM_MD1. This automatic adjustment is enabled with V_PLL_MAN = '0' in register R_PCM_MD2.

V_PLL_MAN = '1' switches into manual adjustment mode. In this case, the adjustment direction is specified in V_PLL_ICR of register R_PCM_MD2. The number of edge adjustments which is specified in V_PLL_ADJ is carried out within the last PCM time slot every 125 µs. This manual adjustment does not stop before V_PLL_MAN is reset to automatic mode. By default, the C4IO clock is adjusted four times for one half clock cycle. This can be reduced to one adjustment of a half clock cycle (see R_PCM_MD1 register). This is useful if a non XHFC series ISDN controller is connected as slave in NT mode to the PCM bus. The synchronization source can be selected by the R_PCM_MD2 register settings.

6.5.4 Manual PLL adjustment

In normal operation mode, the synchronization input signal is passed from the V_SYNC_SRC controlled multiplexer to the PCM DPLL as shown in Figure 6.5. For this V_PLL_MAN has to be '0' in register R_PCM_MD2 .

The PLL output frequency can manually be adjusted if no synchronization source is available. This software controlled PLL adjustment is enabled with V_PLL_MAN = '1'. The V_SYNC_SRC controlled multiplexer must feed a 8 kHz signal in any case.

[1] C4IO adjustment is only in operation when the PCM DPLL receives both 8 kHz reference clocks.

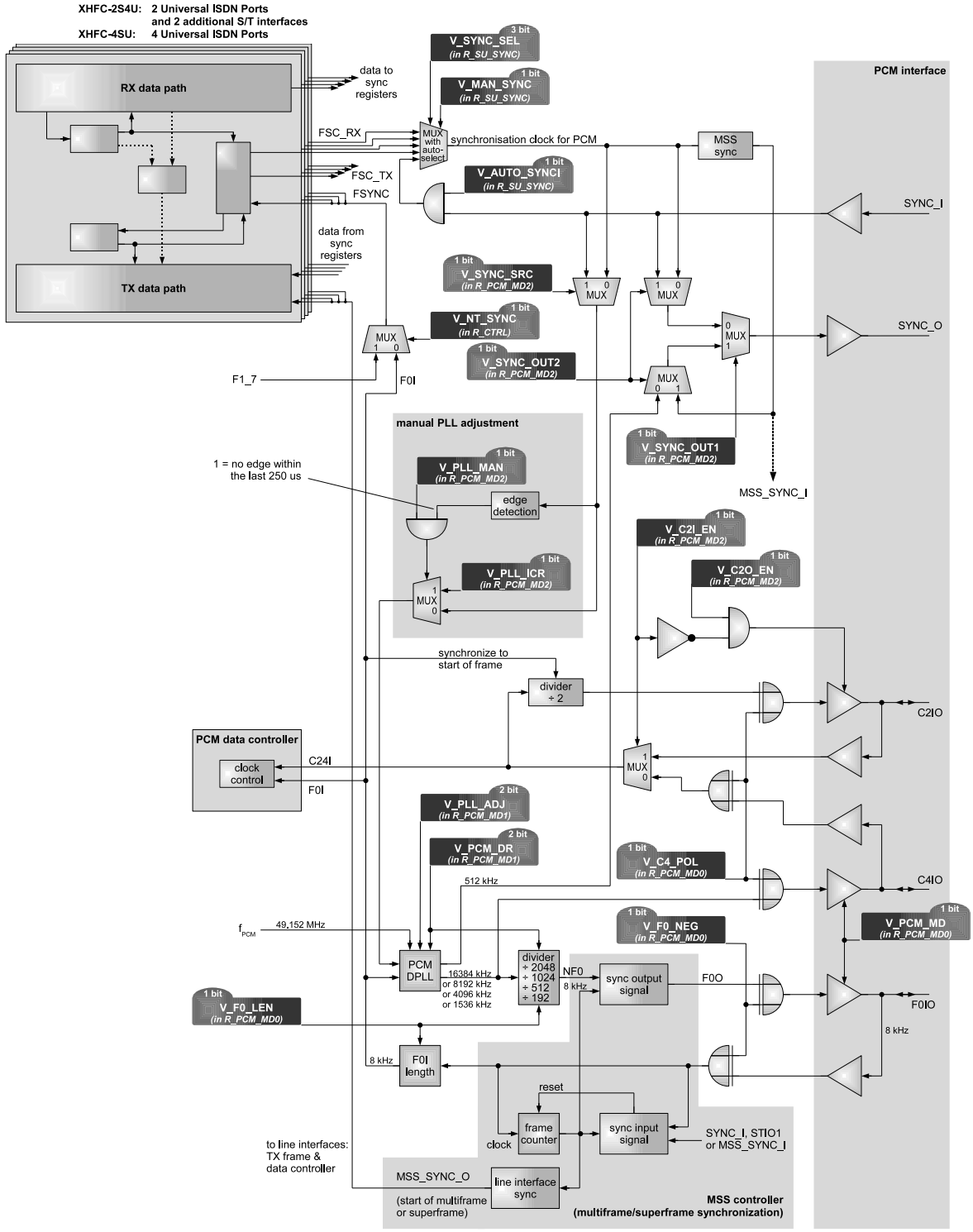


Figure 6.5: *PCM clock synchronization (For details on the Universal ISDN Port see Figures 5.1, 5.5 and 5.13, for details on the MSS controller see Figure 6.10)*

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 30) can either be configured to input or output characteristic.

C2IO is used as an output signal when V_C2O_EN = '1' and V_C2I_EN = '0' in register R_PCM_MD2.

C2IO output signal is derived from C4IO by a frequency divider. In fact, there is an additional building block (not shown in Figure 6.5) which ensures a specific phase relation at the start of a frame. According to the timing diagrams in Figures 6.3 and 6.4, C2IO has its rising edge at the start of a frame. From this follows that C4IO has a falling edge with every edge of C2IO .

6.5.6 SYNC_O and FSC_RX synchronization signals

The 'frame synchronization detection' blocks shown in Figures 5.5 (page 169) and 5.13 (page 185) deliver a 8 kHz frame clock FSC_RX which can be routed to the SYNC_O synchronization output pin. FSC_RX signal is available when a frame synchonization is detected (INFO 2 or INFO 4) and it does not depend on the state machine's condition.

The normal FSC_RX pulse is high for one bit length. This is 5.208 µs in S/T interface mode or 2.604 µs in U_p interface mode. For S/T multiframe [2] synchonization or U_p superframe [3] synchronization, the start of every multiframe or superframe is indicated with a lengthened pulse with 7.5 bit width. This is 39 µs every 40th pulse in S/T interface mode or 19.5 µs every 8th pulse in U_p interface mode.

The rising edges of FSC_RX have a distance of 125 µs and are adjusted once every 250 µs with ±163 ns. The rising edge is stable as long as the chosen TE port receives valid INFO 2 or INFO 4 even if the S/T or U_p state machine is forced into a deactivated state F0, F2, F3, F4, F5, G0, G12, or G2. FSC_RX is low otherwise.

During XHFC-2S4U / 4SU reset, SYNC_O pin drives an active low signal.

6.5.7 Synchronous 512 kHz output signal

SYNC_O frequency can either be 8kHz, which is mostly used, or 512kHz. The 512kHz clock is synchronous to the frame clock as it is generated from the DPLL as shown in Figure 6.5.

When XHFC-2S4U / 4SU operates in PCM slave mode, which means that F0IO is used as clock input, SYNC_O should not issue 512 kHz. This would lead to a closed loop with the internal DPLL and an external PCM master PLL. A closed loop with two PLLs might be instable and can lose the nominal frequency.

[2] A complete S/T multiframe takes 20 S/T frames with 250 µs each. Therefore, a multiframe has a length of 5 ms in total.
[3] A complete U_p superframe takes 4 U_p frames with 250 µs each. Therefore, a superframe has a length of 1 ms in total.

6.5.8 Application examples for XHFC-2S4U / 4SU synchronization schemes

Flexible synchronization schemes can be implemented with the clock pins SYNC_I and SYNC_O of XHFC-2S4U / 4SU. 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-2S4U / 4SU are used within a system, all ST / U_p and PCM clocks must be synchronized to a single synchronization source. When an ST interface operates in TE mode and is connected to the central office, the synchronization source is obtained from the central office. A U_p interface in TE mode is typically synchronized to a PBX clock.

6.5.8.1 An existing system with internal PCM bus has to be expanded by ISDN interfaces

When an existing system has to be expanded by ISDN interfaces, there are two solutions as shown in Figures 6.6 and 6.7.

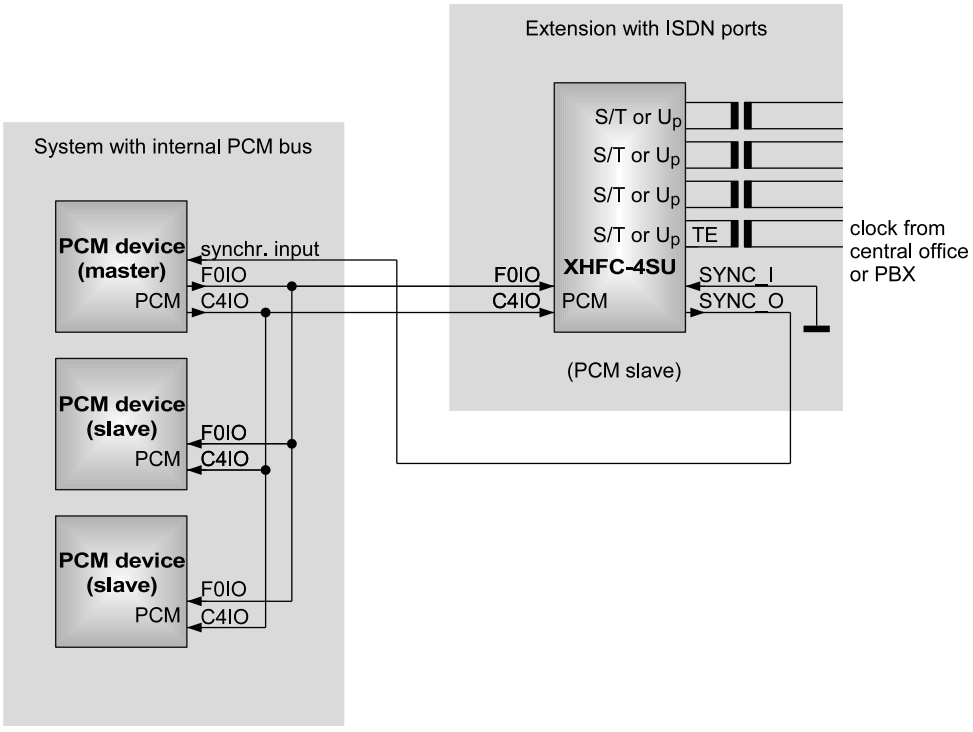


Figure 6.6: *Expanding an existing system with ISDN ports (XHFC-2S4U / 4SU as PCM slave)*

The existing system can keep the PCM master when there is a synchronization input (Figure 6.6). This must be connected to pin SYNC_O of XHFC-2S4U / 4SU to synchronize the whole system to the clock derived from the received ISDN signal.

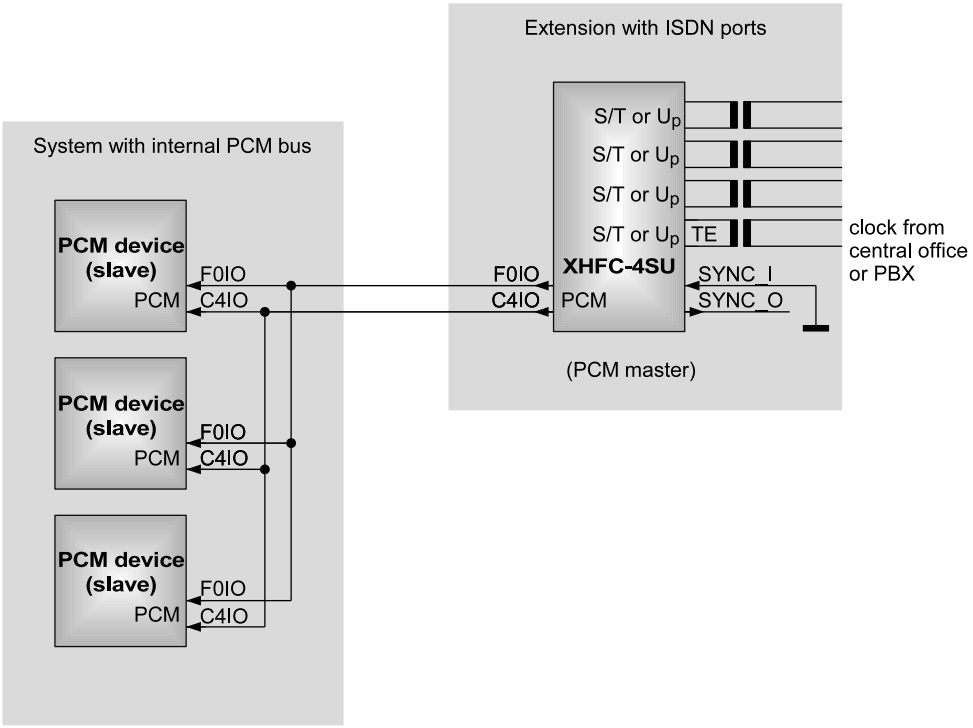


Figure 6.7: *Expanding an existing system with ISDN ports (XHFC-2S4U/4SU as PCM master)*

SYNC_O is either derived from the central office or the DPLL of XHFC-2S4U/4SU is free-running. In both cases, XHFC-2S4U/4SU is the synchronization source for the whole system.

When there is no synchronization input available at the existing system, XHFC-2S4U/4SU must operate as PCM master and all other PCM devices in the system must be PCM slaves (Figure 6.7). Again, XHFC-2S4U/4SU is the synchronization source for the whole system.

6.5.8.2 Application with multiple XHFC-2S4U/4SU

Multiple XHFC-2S4U/4SU can be interconnected to build up a multi ISDN port application. The SYNC_O /SYNC_I pins must be connected to a daisy chain. Two solutions are shown in Figures 6.8 and 6.9, depending on the PCM master requirements.

Figure 6.8 shows an example where any XHFC-2S4U/4SU can be in TE mode. The synchronization signals are daisy chained. The last XHFC-2S4U/4SU 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-2S4U/4SU 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-2S4U/4SU can be in PCM master mode.

Nevertheless, it is recommended to interconnect all devices with the F0IO /C4IO clocks for more flexible application capability.

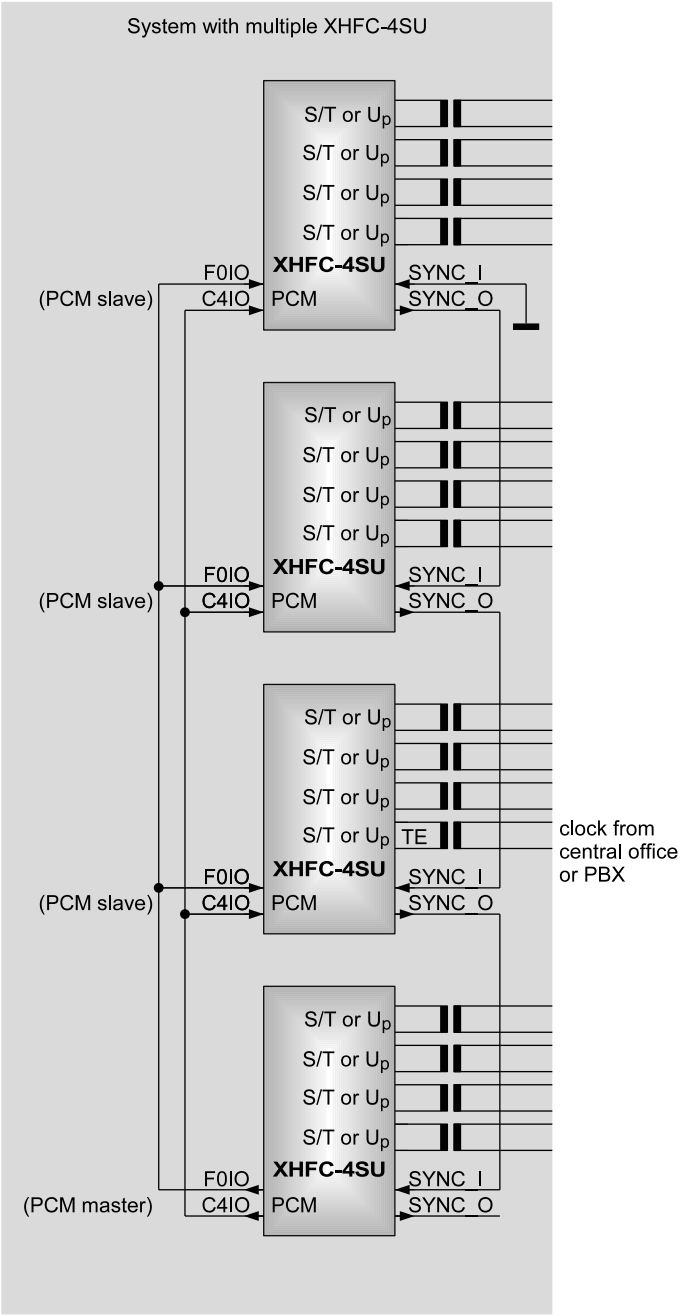


Figure 6.8: *Multiple XHFC-2S4U/4SU synchronized with an open loop of SYNC_O /SYNC_I*

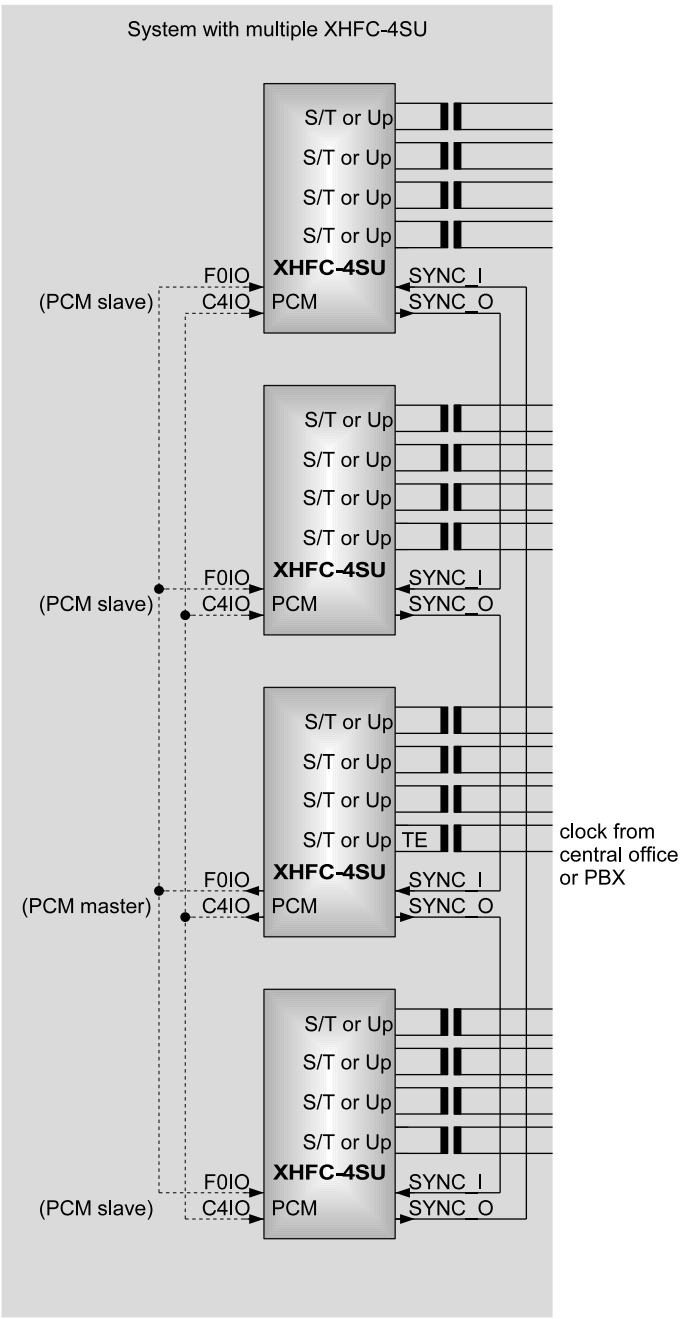


Figure 6.9: *Multiple XHFC-2S4U/4SU synchronized with an closed loop of SYNC_O /SYNC_I*

6.6 Multiframe / superframe synchronization to the PCM interface

6.6.1 Overview

The multiframes (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 169 and Figure 5.13 on page 185)

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.

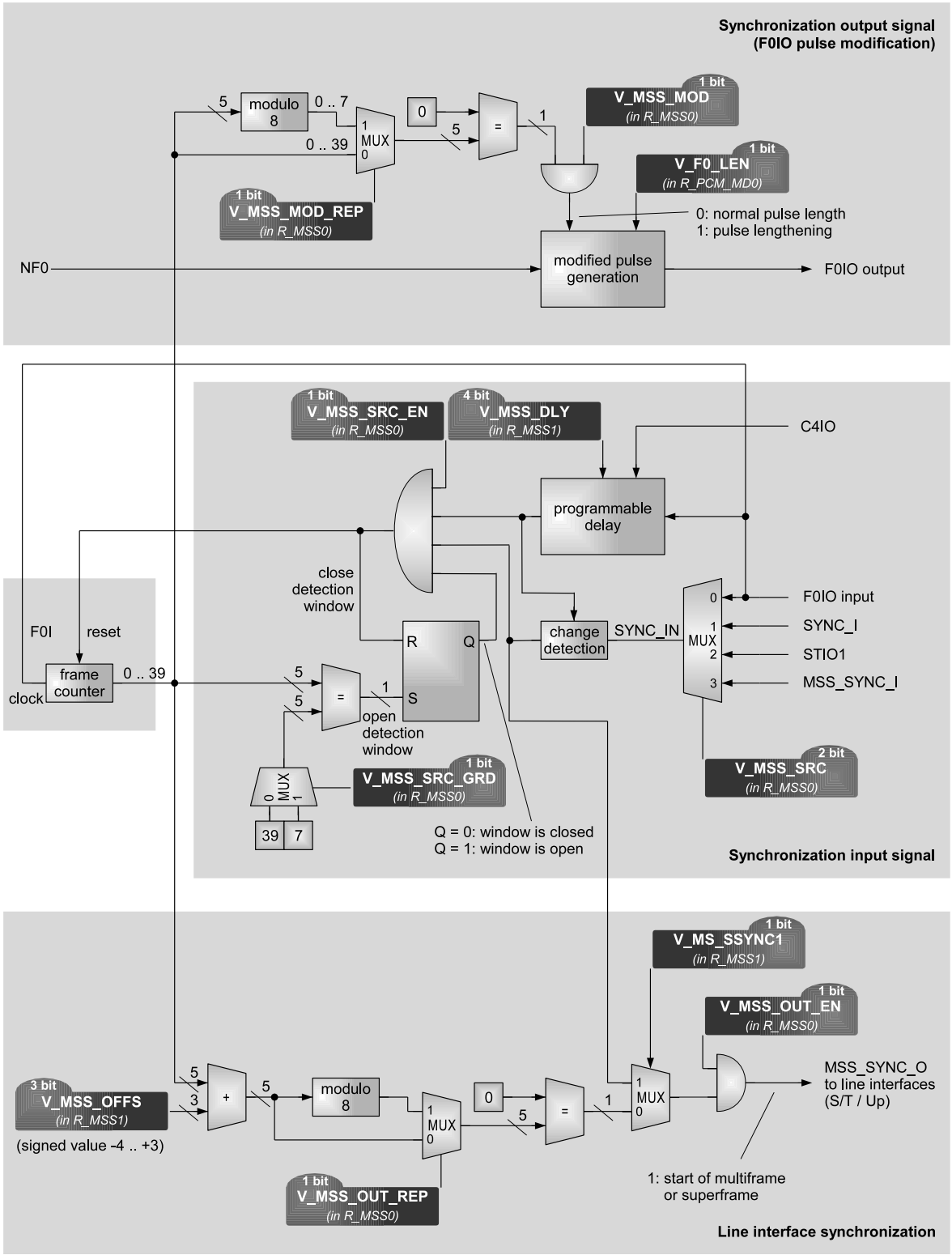


Figure 6.10: *Multiframe / superframe synchronization to the PCM interface*

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 oneine interface operates in S/T mode. The value V_MSS_SRC_GRD = '1' is recommended when all line interfaces operate in U_p mode.

The frame counter is synchronized to the synchronization input signal when the following conditions are fulfilled at the same time:

1. The synchronization detection is enabled, i.e. V_MSS_SRC_EN = '1'.
2. The detection windows is open, i.e. Q = '1'.
3. The programmable delay is passed.
4. The selected input signal has changed from the previous frame.

The delay time t_{DLY} is specified in Figure 6.11 and Table 6.6. Its maximum value is $t_{DLY,max} = 16 \cdot t_{C4P}$.

If the synchronization pulse is received on the data input line STIO1 , it can be any bit within the first PCM time slot. V_MSS_DLY must have an even value in this case to ensure that the *change detection* time matches with the bit cells. V_MSS_DLY = 0 selects the first bit of the time slot 0 while V_MSS_DLY = 14 selects the last bit.

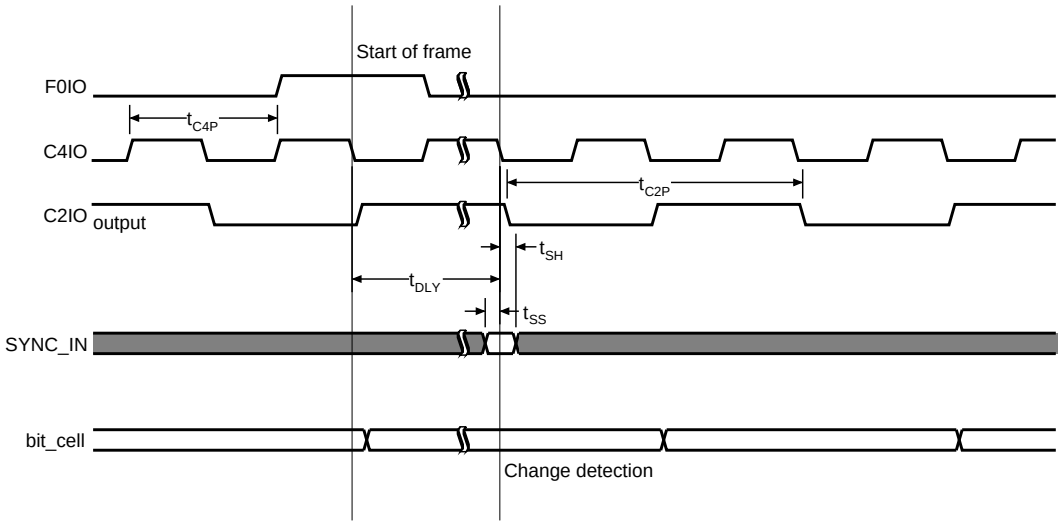


Figure 6.11: *Timing specification of the SYNC_IN change detection*

Table 6.6: *Symbols of the SYNC_IN change detection timing specification in Figure 6.11*

Symbol	min / ns	max / ns	Characteristic
t_{C4P}			C4IO clock period (depends on the selected PCM data rate)
t_{C2P}			C2IO clock period
t_{DLY}			Programmable synchronization pulse delay from start of frame $t_{DLY} = (V\_MSS\_DLY + 1) \cdot t_{C4P}$
t_{SS}	10		Synchronization pulse valid to F_SYNC ⌐ setup time
t_{SH}	10		Synchronization pulse valid to F_SYNC ¬ hold time

6.6.4 Synchronization output signal (F0IO pulse modification)

The start of a multiframe or superframe is indicated by a F0IO pulse which has a different length than usual.

In principle, the length of the F0IO pulse is arbitrary. The F0IO signal is used for PCM frame synchronization only on the first falling edge of C4IO . The F0IO high time has a typical length of one C4IO pulse (V_F0_LEN = '0') or two C4IO pulses (V_F0_LEN = '1') as described in Section 6.4.2.

The F0IO high time can be lengthen by another C4IO pulse width to indicate the multiframe / superframe synchronization signal. This must be enabled with V_MSS_MOD = '1' in register R_MSS0. The repetition rate of the synchronization pulse is once every 40th PCM frame if V_MSS_MOD_REP = '0' in register R_MSS0. V_MSS_MOD_REP = '1' selects a repetition rate of once every 8th PCM frame.

Figure 6.12 shows all different F0IO pulses in PCM master mode. Table 6.7 summarizes the bit values which must be set to achieve these pulses.

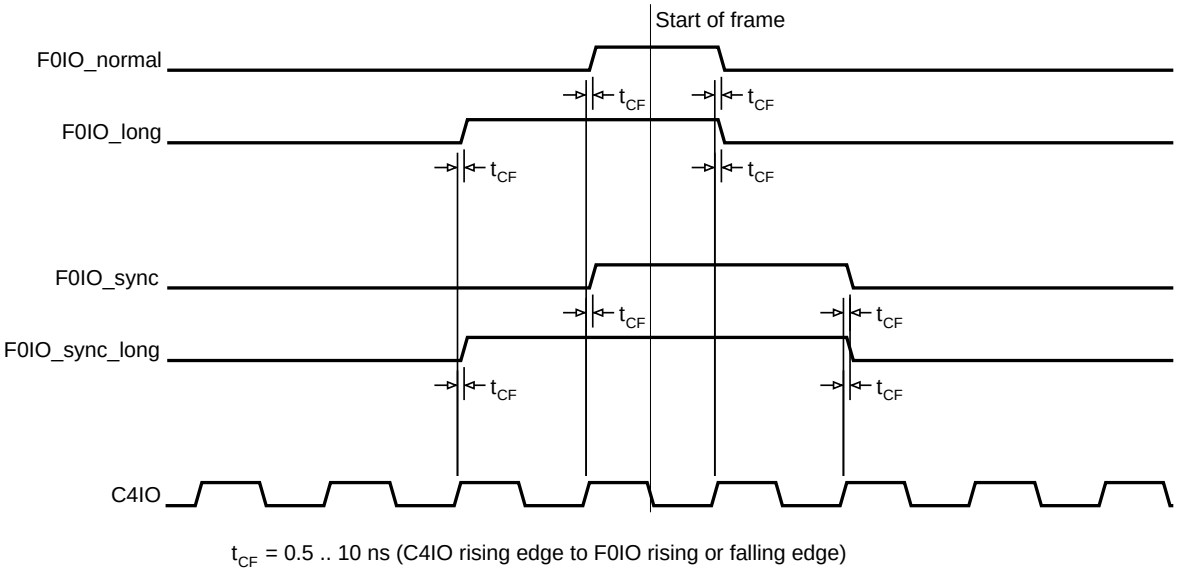


Figure 6.12: *F0IO pulse modification*

Table 6.7: *Selection of the F0IO pulse characteristic (see also Figure 6.12)*

Pulse	V_F0_LEN in register R_PCM_MD0	V_MSS_MOD in register R_MSS0	Description
F0IO_normal	'0'	'0'	normal F0IO pulse
F0IO_long	'1'	'0'	F0IO starts one C4IO period earlier (early leading edge)
F0IO_sync	'0'	'1'	F0IO pulse is lengthened by one C4IO pulse every 8th or 40th frame, F0IO_normal otherwise (delayed trailing edge)
F0IO_sync_long	'1'	'1'	F0IO starts one C4IO period earlier and is lengthened by one C4IO pulse every 8th or 40th frame, F0IO_long otherwise (combined early leading edge and delayed trailing edge)

6.6.5 Line interface synchronization

The signal MSS_SYNC_O for the multiframe / superframe synchronization is delivered to the ST / U_p interfaces. It must be enabled with V_MSS_OUT_EN = '1' in register R_MSS0. V_MSS_OUT_EN = '0' leads to an unsynchronized transmission of the multiframe / superframe structure.

The repetition rate can be specified to be either every 40th or 8th PCM frame with bit V_MSS_OUT_REP in register R_MSS0. Only line interfaces in U_p mode can be feed with a 8 PCM frame repetition rate. The repetition rate of once every 40 PCM frames must be chosen in S/T mode. This is a suitable selection for a line interface in U_p mode as well.

As there is a delay between F0IO and the line interface output, it is possible to adjust the line interface synchronization with a signed offset of $-4 .. +3$ F0IO pulses. This can also be used to fulfill requirements which might occur with a special application where the chip is connected to another ISDN device.

6.7 External CODECs

Up to two external CODECs can be connected to the PCM interface. XHFC-2S4U/4SU has two CODEC enable signals F1_0 and F1_1 . An external CODEC has to be assigned to a PCM time slot via bitmaps V_SL_SEL1 and V_SL_SEL0 in registers R_SL_SEL1 and R_SL_SEL0.

The shape signals can be programmed. The last bit determines the inactive level by which non-inverted and inverted shape signals can be programmed. Every external CODEC can choose one of the two shape signals with bits V_SH_SEL1 and V_SH_SEL0 in registers R_SL_SEL1 and R_SL_SEL0.

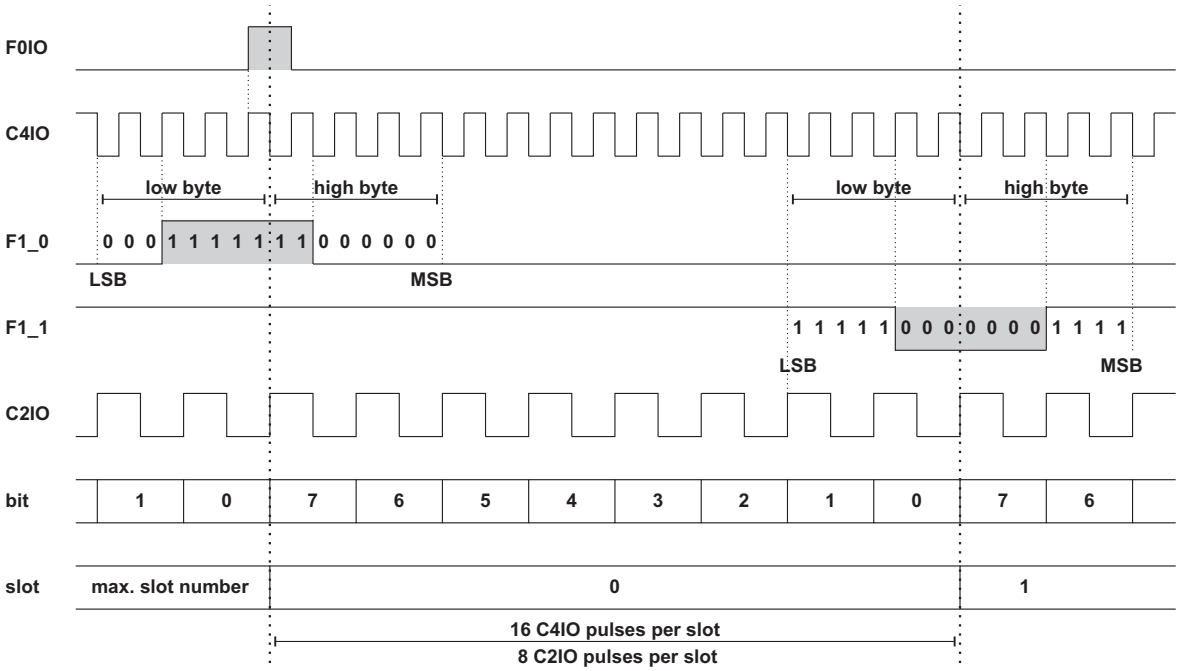


Figure 6.13: *Example for two CODEC enable signal shapes*

Figure 6.13 shows an example with two external CODECs. Time slot 0 starts with the F0IO pulse. In this example – assuming that PCM30 is configured – F1_0 enables the first CODEC on time slot 0 and shape bytes on R_SH0L and R_SH0H with the following register settings.

Register setup:

R_PCM_MD0	: V_PCM_IDX = 0		(R_SL_SEL0 register accessible)
R_SL_SEL0	: V_SL_SEL0 = 0x1F		(time slot #0)
	: V_SH_SEL0 = 0		(shape bytes R_SH0L and R_SH0H)

The second CODEC on time slot 1 and shape bytes on R_SH1L and R_SH1H must be configured as shown below.

Register setup:

R_PCM_MD0	: V_PCM_IDX	= 1		(R_SL_SEL1 register accessible)
R_SL_SEL1	: V_SL_SEL1	= 0		(time slot #1)
	: V_SH_SEL1	= 1		(shape bytes R_SH1L and R_SH1H)

The shown shape signals have to be programmed in reverse bit order by the following register settings.

Register setup:

R_PCM_MD0	: V_PCM_IDX	= 0xC		(R_SH0L register accessible)
R_SH0L	: V_SH0L	= 0xF8		(0xF8 = '1111 1000' $\overset{reverse}{\longrightarrow}$ '0001 1111')
R_PCM_MD0	: V_PCM_IDX	= 0xD		(R_SH0H register accessible)
R_SH0H	: V_SH0H	= 0x03		(0x03 = '0000 0011' $\overset{reverse}{\longrightarrow}$ '1100 0000')
R_PCM_MD0	: V_PCM_IDX	= 0xE		(R_SH1L register accessible)
R_SH1L	: V_SH1L	= 0x1F		(0x1F = '0001 1111' $\overset{reverse}{\longrightarrow}$ '1111 1000')
R_PCM_MD0	: V_PCM_IDX	= 0xF		(R_SH1H register accessible)
R_SH1H	: V_SH1H	= 0xF0		(0xF0 = '1111 0000' $\overset{reverse}{\longrightarrow}$ '0000 1111')

6.8 GCI / IOM-2 mode

6.8.1 Overview

XHFC-2S4U / 4SU is equipped with a simple GCI controller [4] (also known as IOM™-2) [5] to support interconnection to U-chips, external CODECs or DSPs.

The IOM™-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 [6]. The GCI [7] functionality has been implemented in respect to the IOM™-2 specification [7].

The interconnection between XHFC-2S4U / 4SU and a GCI device uses four wires, typically:

- C4IO : Double bit rate clock
- F0IO : 8 kHz frame signal
- STIO1 : Data from XHFC-2S4U / 4SU to the GCI device (can be swapped with STIO2)
- STIO2 : Data from the GCI device to XHFC-2S4U / 4SU (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.

[4] GCI = General Circuit Interface
[5] IOM™-2= ISDN Oriented Modular revision 2, trademark of Infineon Technologies AG
[6] Alcatel (France), Siemens (Germany), Italtel (Italy) and Plessey (UK)
[7] The term GCI is always used in this document and includes IOM™-2 functionality as well.

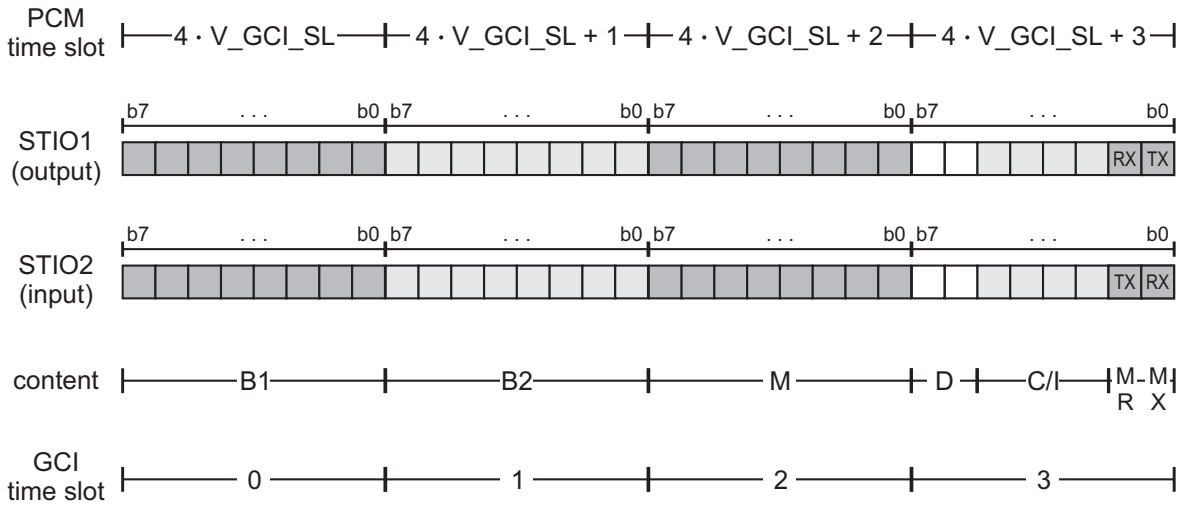


Figure 6.14: *Single channel GCI format*

Table 6.8: *Legend of figure 6.14*

Name	Description
B1	B1-channel data
B2	B2-channel data
M	Monitor channel data
D	D-channel data
C/I	Command/indication bits for controlling activation/deactivation and for additional control functions
MR	Handshake bit for the monitor channel (MR_{RX} on STIO1 output and MR_{TX} on STIO2 input)
MX	Handshake bit for the monitor channel (MX_{TX} on STIO1 output and MX_{RX} on STIO2 input)

6.8.3 GCI register programming

6.8.3.1 Enable CGI functionality

The GCI functionality is disabled after XHFC-2S4U / 4SU 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 $4 \cdot V\_GCI\_SL + 2$ and $4 \cdot V\_GCI\_SL + 3$ 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-2S4U / 4SU 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-2S4U / 4SU. 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-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU is both PCM master and GCI master (most application cases, standard configuration). This means:

- XHFC-2S4U / 4SU 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 slotwhen GCI is enabled. C/I-channel and handshake bits MX and MR are also automatically assigned to their time slot.

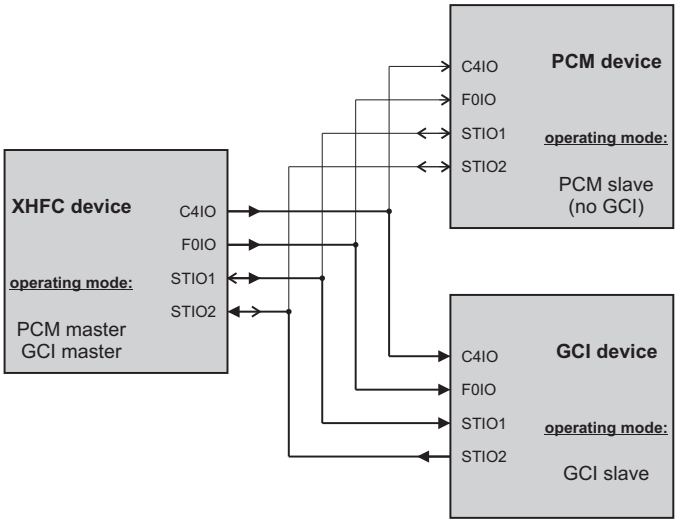


Figure 6.15: *GCI application example 1 (XHFC-2S4U / 4SU is PCM master and GCI master)*

The second application example shown in Figure 6.16 has only one difference to the first example: XHFC-2S4U / 4SU 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-2S4U / 4SU operating in GCI slave mode. XHFC-2S4U / 4SU 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-2S4U / 4SU microprocessor interface even if XHFC-2S4U / 4SU 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-2S4U/4SU 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

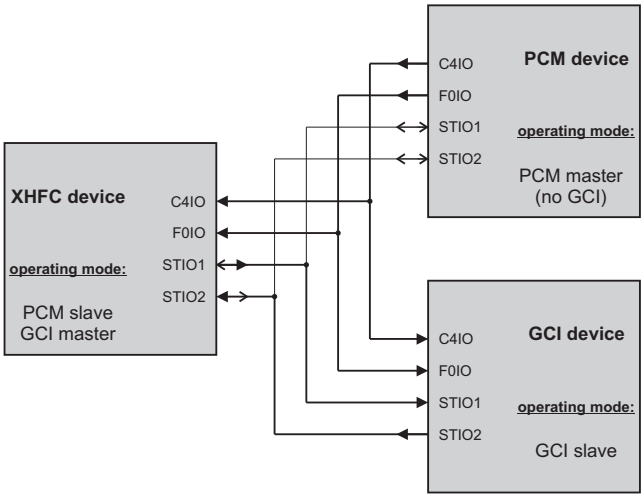


Figure 6.16: *GCI application example 2 (XHFC-2S4U/4SU is PCM slave and GCI master)*

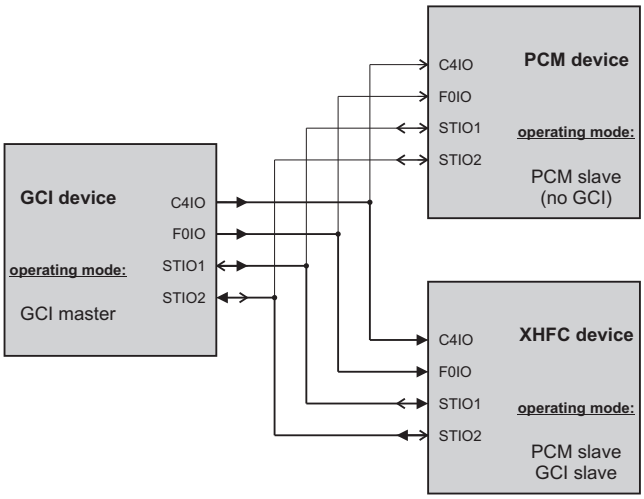


Figure 6.17: *GCI application example 3 (XHFC-2S4U/4SU is PCM slave and GCI slave)*

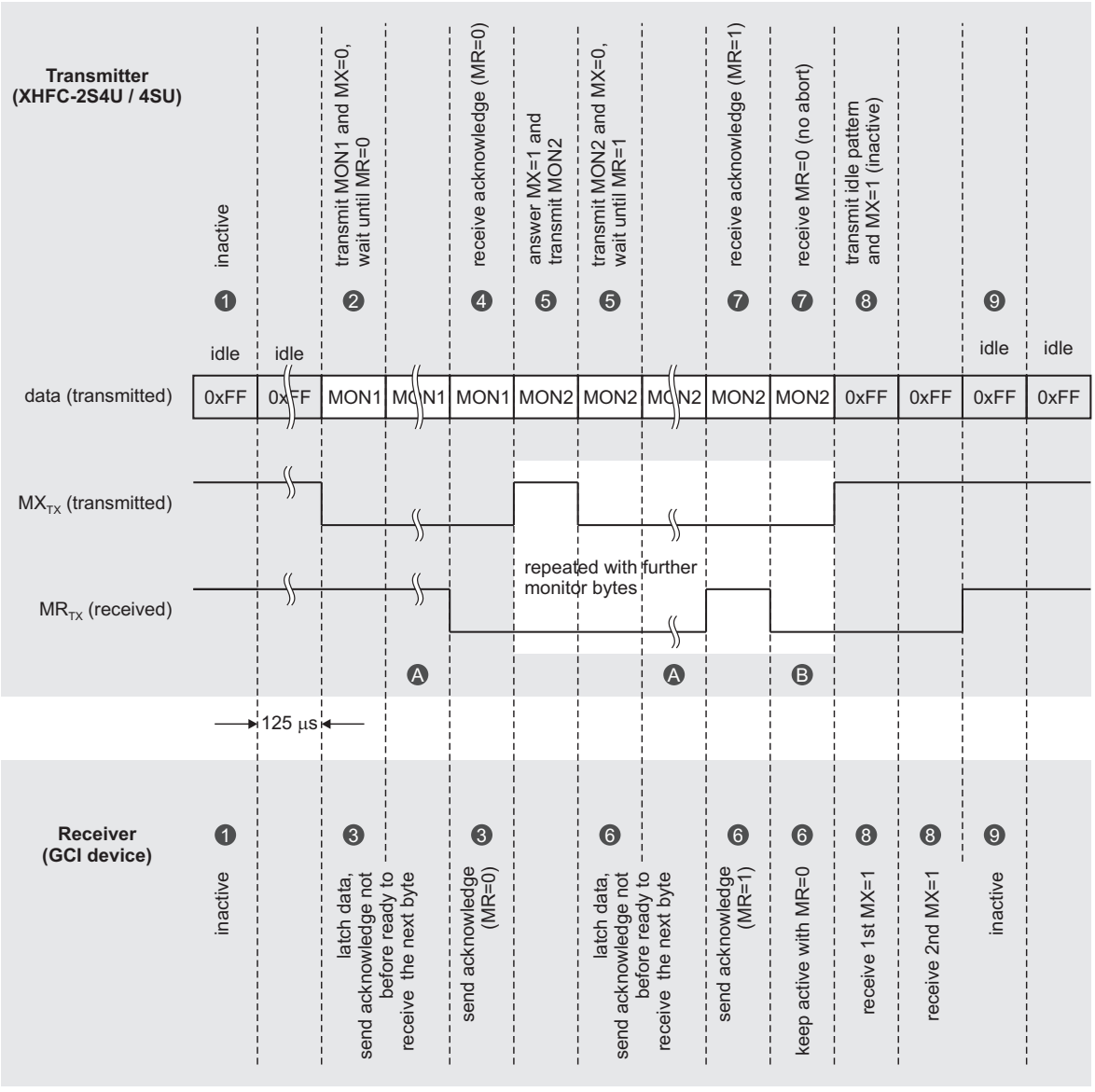
Table 6.10: *Programming procedure for application example 3 according to Figure 6.17 (XHFC-2S4U/4SU is PCM slave and GCI slave)*

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-2S4U / 4SU 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.



Ⓐ omitted, if the receiver sends acknowledge immediately

Ⓑ omitted, if V_MON_SLOW = 0 (high transmission speed)

Figure 6.18: *GCI protocol for monitor bytes transmission*

❶ 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.

❷ 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.

❸ The receiver latches the first monitor byte. This can take an arbitrary number of 125 µs cycles. An acknowledge is send when the receiver is ready for the next byte.

❹ The transmitter gets acknowledge with MR_{TX} = '0'.

❺ XHFC-2S4U / 4SU 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.

❻ The receiver latches the second monitor byte. This can take an arbitrary number of 125 µs cycles. An acknowledge signal MR_{TX} = '1' is send for one cycle when the receiver is ready for the next byte.

❼ The transmitter receives acknowledge with MR_{TX} = '1' for one cycle.

❽ The procedure shown with the second monitor byte (❺..❼) 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.

❾ 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-2S4U / 4SU transmits a monitor byte sequence.

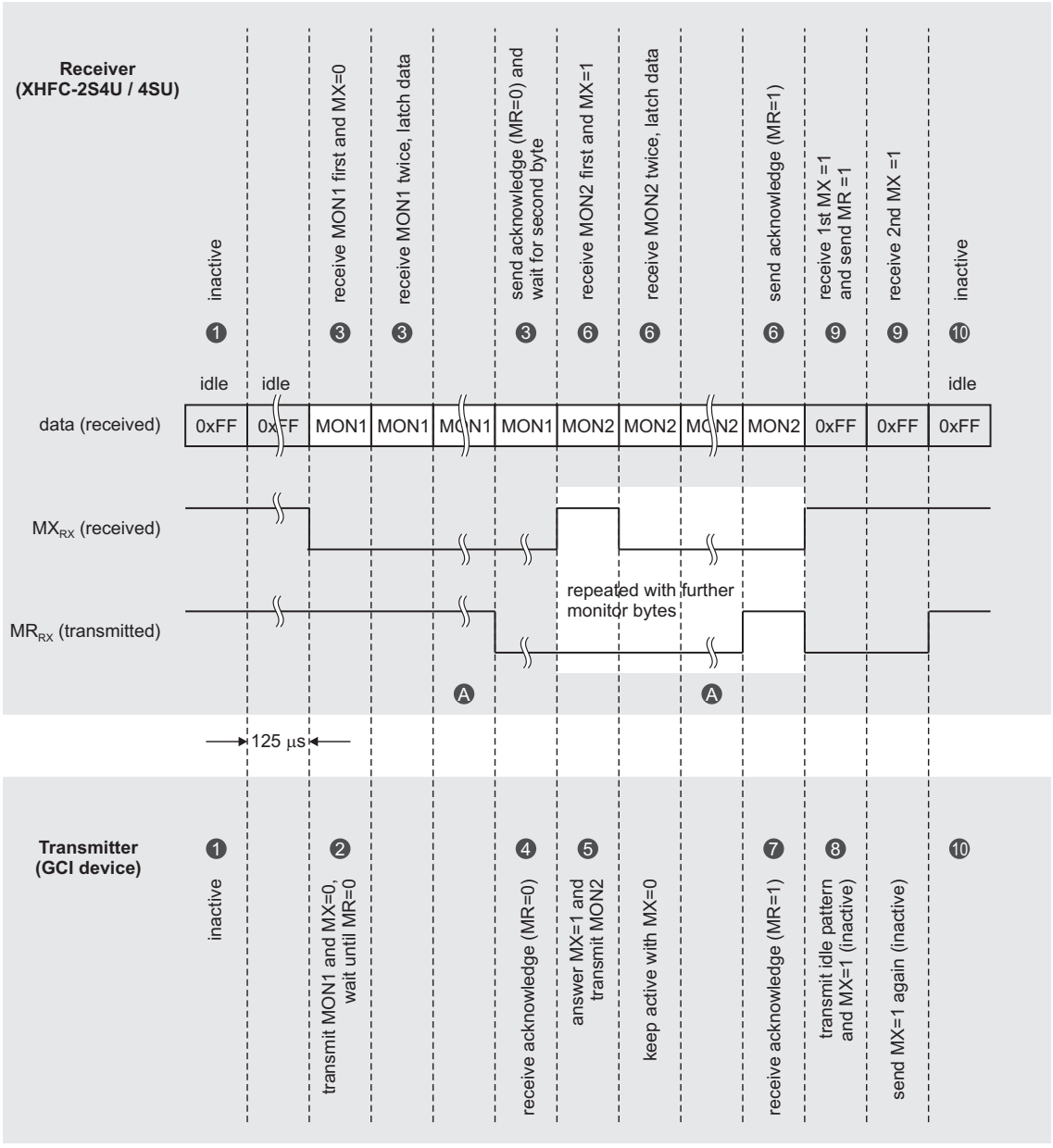
Table 6.11: *Rules for the handshake signals MX_{TX} and MR_{TX} when XHFC-2S4U / 4SU transmits a monitor byte sequence*

MX_{TX}	MR_{TX}	Monitor channel state
'1'	'1'	Idle
'0'	'0'	Transmitter and receiver are both active
'0'	'1' (once)	Acknowledge of the 2nd and following monitor bytes
'0'	'1' (repeated)	Transmitter waits for acknowledge to the 1st monitor byte
'1' (once)	'0'	Transmitter answers to the acknowledge
'1' (repeated)	'0'	Transmitter terminates the transmission

6.8.4.2 XHFC-2S4U / 4SU receive procedure

The monitor channel is full duplex and operates similar in the opposite direction. XHFC-2S4U / 4SU 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 µs 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 µs 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..❼) 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-2S4U / 4SU 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-2S4U / 4SU 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-2S4U / 4SU 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 1st 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-2S4U / 4SU 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-2S4U / 4SU never aborts any message when it is receiver of the monitor channel.

6.9 Register description

6.9.1 Write only registers

R_SLOT	(w)	(Reset group: H, 0, 2)	0x10

PCM time slot selection

This register is used to select a PCM time slot. Before a PCM slot array register can be accessed, this index register must specify the desired slot number and data direction. Depending on the V_PCM_DR value in register R_PCM_MD1, either 32, 64 or 128 time slots are available for each data direction.

Bits	Reset value	Name	Description
0	0	**V_SL_DIR**	**PCM time slot data direction** '0' = transmit PCM data '1' = receive PCM data
7..1	0x00	**V_SL_NUM**	**PCM time slot number**

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PCM_MD0	(w)	(Reset group: H, 0, 2)	0x14

PCM mode, register 0

Bits	Reset value	Name	Description
0	0	**V_PCM_MD**	**PCM bus mode** '0' = slave (pins C4IO and F0IO are inputs) '1' = master (pins C4IO and F0IO are outputs) If no external C4IO and F0IO signal is provided this bit must be set for operation.
1	0	**V_C4_POL**	**Polarity of C4IO clock** '0' = pin F0IO is sampled on negative clock transition of C4IO '1' = pin F0IO is sampled on positive clock transition of C4IO
2	0	**V_F0_NEG**	**Polarity of F0IO signal** '0' = positive pulse '1' = negative pulse
3	0	**V_F0_LEN**	**Duration of F0IO signal** '0' = active for one C4IO clock (244 ns at 4 MHz) '1' = active for two C4IO clocks (488 ns at 4 MHz, early leading edge) The specified signal duration is generated in PCM master mode and it is expected in PCM slave mode.
7..4	0	**V_PCM_IDX**	**Index value to select the register at address 15** At address 15 a so-called multi-register is accessible. 0 = R_SL_SEL0 register accessible 1 = R_SL_SEL1 register accessible 2 = not used 3 = not used 4 = not used 5 = not used 6 = not used 7 = R_SL_SEL7 register accessible 8 = R_MSS0 register accessible 9 = R_PCM_MD1 register accessible 0xA = R_PCM_MD2 register accessible 0xB = R_MSS1 register accessible 0xC = R_SH0L register accessible 0xD = R_SH0H register accessible 0xE = R_SH1L register accessible 0xF = R_SH1H register accessible

R_SL_SEL0	(w)	(Reset group: H, 0, 2)	0x15

Slot selection register for pin F1_0

This multi-register is selected with bitmap V_PCM_IDX = 0 in register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_0 is disabled.

Bits	Reset value	Name	Description
6..0	0x7F	**V_SL_SEL0**	**PCM time slot selection** The selected slot number is V_SL_SEL0 +1 for F1_0 . Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL0**	**Shape selection** '0' = use shape 0 set by registers R_SH0L and R_SH0H '1' = use shape 1 set by registers R_SH1L and R_SH1H

Important !

For selecting slot 0, the value that has to be written into bitmaps V_SL_SEL0 and V_SL_SEL1 of registers R_SL_SEL0 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_SEL0 = '0' and V_SH_SEL1 = '0' (SHAPE 0) in registers R_SL_SEL0 and R_SL_SEL1.

R_SL_SEL1	(w)	(Reset group: H, 0, 2)	0x15

Slot selection register for pin F1_1

This multi-register is selected with bitmap V_PCM_IDX = 1 in register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_1 is disabled.

Bits	Reset value	Name	Description
6..0	0x7F	**V_SL_SEL1**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_1 . Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL1**	**Shape selection** '0' = use shape 0 set by registers R_SH0L and R_SH0H '1' = use shape 1 set by registers R_SH1L and R_SH1H

R_SL_SEL7	(w)	(Reset group: H, 0, 2)	0x15

Slot selection register for signal F1_7

This multi-register is selected with bitmap V_PCM_IDX = 7 in register R_PCM_MD0.

The F1_7 signal is only internally available. It can be used to shift the synchronization signal for Universal ISDN Ports in NT mode.

Bits	Reset value	Name	Description
6..0	0x7F	**V_SL_SEL7**	**PCM time slot selection** The selected slot number is V_SL_SEL7 +1 for the F1_7 signal. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	0	(reserved)	Must be '0'.

R_MSS0	(w)	(Reset group: H, 0, 2)	0x15

PCM multiframe / superframe synchronization mode, register 0

This multi-register is selected with bitmap V_PCM_IDX = 8 in register R_PCM_MD0.

Bits	Reset value	Name	Description
0	0	**V_MSS_MOD**	**F0IO pulse modification** The F0IO pulse duration can be changed to generate the multiframe / superframe synchronization signal. '0' = normal operation '1' = F0IO pulse is lengthened by one C4IO clock to indicate the start of the multiframe / superframe (delayed trailing edge)
1	0	**V_MSS_MOD_REP**	**F0IO modification repetition rate** The repetition rate of the modified F0IO signal can be selected. '0' = once every 40 PCM frames '1' = once every 8 PCM frames
2	0	**V_MSS_SRC_EN**	**Enable external multiframe / superframe synchronization signal** '0' = disabled '1' = enabled
3	0	**V_MSS_SRC_GRD**	**Synchronization guard count** The multiframe / superframe synchronization signal is detected not before 7 or 39 PCM frames. '0' = not before 39 PCM frames '1' = not before 7 PCM frames
4	0	**V_MSS_OUT_EN**	**Enable line interface synchronization signal for the multiframe / superframe** '0' = the synchronization signal is disabled '1' = the synchronization signal is passed to the line interfaces
5	0	**V_MSS_OUT_REP**	**Repetition rate of the line interface synchronization signal for the multiframe / superframe** The line interface synchronization signal is generated either every 8th or 40th PCM frame. '0' = every 40th PCM frame '1' = every 8th PCM frame (can only be used in pure Up environments)

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7..6	0	**V_MSS_SRC**	**Multiframe / superframe synchronization source** One of the following signals can be chosen as synchronization source. '00' = pin F0IO '01' = SYNC_I '10' = STIO1 '11' = internal signal MSS_SYNC_I

R_PCM_MD1	(w)	(Reset group: H, 0, 2)	0x15

PCM mode, register 1

This multi-register is selected with bitmap V_PCM_IDX = 9 in register R_PCM_MD0.

Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	**V_PCM_OD**	**Characteristic of the PCM output lines** '0' = STIO1 and STIO2 have push / pull characteristic '1' = STIO1 and STIO2 have open drain characteristic (pull up resistor required)
3..2	0	**V_PLL_ADJ**	**DPLL adjust speed** '00' = C4IO clock is adjusted in the last time slot of the PCM frame 4 times by one clock cycle of the PCM clock f_{PCM} (81.4 ns all 125 µs, 651 ppm) '01' = C4IO clock is adjusted in the last time slot of the PCM frame 3 times by one clock cycle of the PCM clock f_{PCM} (61.0 ns all 125 µs, 489 ppm) '10' = C4IO clock is adjusted in the last time slot of the PCM frame twice by one clock cycle of the PCM clock f_{PCM} (40.7 ns all 125 µs, 326 ppm) '11' = C4IO clock is adjusted in the last time slot of the PCM frame once by one clock cycle of the PCM clock f_{PCM} (20.3 ns all 125 µs, 163 ppm)
5..4	0	**V_PCM_DR**	**PCM data rate** '00' = 2 MBit/s (C4IO is 4.096 MHz, 32 time slots) '01' = 4 MBit/s (C4IO is 8.192 MHz, 64 time slots) '10' = 8 MBit/s (C4IO is 16.384 MHz, 128 time slots) '11' = 0.75 MBit/s (C4IO is 1.536 MHz, 12 time slots) Every time slot exists in transmit and receive data direction.
6	0	**V_PCM_LOOP**	**PCM test loop** When this bit is set, the PCM output data is looped to the PCM input data internally for all PCM time slots. **Note:** When this bit is set (internal PCM loop), it is not allowed to set bit V_PCM_OD in register R_PCM_MD1 as well.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	**V_PCM_SMPL**	**PCM receive sample point** '0' = sample point at middle of PCM bit cell (normal operation) '1' = sample point at 3/4 of PCM bit cell

R_PCM_MD2	(w)	(Reset group: H, 0, 2)	0x15

PCM mode, register 2

This multi-register is selected with bitmap V_PCM_IDX = 0xA in register R_PCM_MD0.

Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	**V_SYNC_OUT1**	**SYNC_O output signal selection** V_SYNC_OUT2 is also used for synchronization selection. '0' = SYNC_O signal is either SYNC_I or the received synchronization pulse FSC_RX (see register R_SU_SYNC for synchronization source selection) '1' = SYNC_O signal is either 512 kHz from the PLL or the received multiframe / superframe synchronization pulse
2	0	**V_SYNC_SRC**	**PCM PLL synchronization source selection** '0' = line interface (see R_SU_SYNC for further synchronization configuration) '1' = SYNC_I input (8 kHz)
3	0	**V_SYNC_OUT2**	**SYNC_O output signal selection** V_SYNC_OUT1 is also used for synchronization selection. '0' = SYNC_O signal is either the received synchronization pulse FSC_RX (see register R_SU_SYNC for synchronization source selection) or 512 kHz from the PLL '1' = SYNC_O signal is either SYNC_I or the received multiframe / superframe synchronization pulse
4	0	**V_C2O_EN**	**Enable C2IO output signal** '0' = C2IO output is disabled '1' = C2IO output is enabled when also V_C2I_EN = '0'
5	0	**V_C2I_EN**	Single Clock on C2IO is used as PCM clock (master and slave mode) '0' = PCM data controller gets C4IO input clock '1' = PCM data controller gets C2IO input clock
6	0	**V_PLL_ICR**	**Increase PCM frame time** This bit is only valid if V_PLL_MAN is set. '0' = PCM frame time is reduced as selected by bitmap V_PLL_ADJ in register R_PCM_MD1 '1' = PCM frame time is increased as selected by bitmap V_PLL_ADJ in register R_PCM_MD1

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	**V_PLL_MAN**	**Manual PLL adjustment** '0' = PCM PLL is automatically adjusted to the SYNC_I signal or to the line interface synchronization pulse (depending on V_SYNC_OUT2 setting) '1' = PCM PLL is manually adjusted according to V_PLL_ICR. This can be used to make synchronization by software if no synchronization source is available. **Note:** Manual PLL adjustment is automatically disabled when a synchronization pulse is available.

R_MSS1	(w)	(Reset group: H, 0, 2)	0x15

PCM multiframe / superframe synchronization mode, register 1

This multi-register is selected with bitmap V_PCM_IDX = 0xB in register R_PCM_MD0.

Bits	Reset value	Name	Description
2..0	0	**V_MSS_OFFS**	**Synchronization source offset** The offset between the synchronization frame counter and the generated synchronization signal is specified in number of PCM frames. The value can be chosen in the range -4 . . . +3. '000' = no offset '001' = 1 PCM frame '010' = 2 PCM frames '011' = 3 PCM frames '100' = -4 PCM frames '101' = -3 PCM frames '110' = -2 PCM frames '111' = -1 PCM frames
3	0	**V_MS_SSYNC1**	**Multiframe / superframe single synchronization pulse** '0' = a repetitive synchronization signal is generated every 8 or 40 PCM frames '1' = the multiframe / superframe synchronization input signal is directly used (this is normally used to generate a single MSS_SYNC_O pulse) **Note:** When this bit is set to '1', usually also V_MS_SSYNC2 has to be set.
7..4	0	**V_MSS_DLY**	**Multiframe / superframe delay** The position of multiframe / superframe data is always related to the regular F0IO signal. This bitmap specifies the delay in number of C4IO clock pulses. 0 = 1 C4IO pulse delay 1 = 2 C4IO pulses delay . . . 15 = 16 C4IO pulses delay

R_SH0L	(w)	(Reset group: H, 0, 2)	0x15

CODEC enable signal F1_0 , low byte

This multi-register is selected with bitmap V_PCM_IDX = 0xC in register R_PCM_MD0.

Bits	Reset value	Name	Description
7..0	0x00	**V_SH0L**	**Shape bits 7..0** Every bit is used for 1/2 C4IO clock cycle.

R_SH0H	(w)	(Reset group: H, 0, 2)	0x15

CODEC enable signal F1_0 , high byte

This multi-register is selected with bitmap V_PCM_IDX = 0xD in register R_PCM_MD0.

Bits	Reset value	Name	Description
7..0	0x00	**V_SH0H**	**Shape bits 15..8** Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH0H defines the value for the rest of the period.

R_SH1L	(w)	(Reset group: H, 0, 2)	0x15

CODEC enable signal F1_1 , low byte

This multi-register is selected with bitmap V_PCM_IDX = 0xE in register R_PCM_MD0.

Bits	Reset value	Name	Description
7..0	0x00	**V_SH1L**	**Shape bits 7..0** Every bit is used for 1/2 C4IO clock cycle.

R_SH1H	(w)	(Reset group: H, 0, 2)	0x15

CODEC enable signal F1_1 , high byte

This multi-register is selected with bitmap V_PCM_IDX = 0xF in register R_PCM_MD0.

Bits	Reset value	Name	Description
7..0	0x00	**V_SH1H**	**Shape bits 15..8** Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH1H defines the value for the rest of the period.

R_SU_SYNC	(w)	(Reset group: H, 0, 3)	0x17

ST/Up synchronization source

This register selects the synchronization source for the internal or external PCM clock PLL.

Bits	Reset value	Name	Description
2..0	0	**V_SYNC_SEL**	**Synchronization source selection** Any line interface or the SYNC_I input signal can be selected as synchronization source. A line interface can be used as synchronization source only if it is in TE mode. '000' = source is line interface 0 '001' = source is line interface 1 '010' = source is line interface 2 '011' = source is line interface 3 '100' = source is SYNC_I signal '101'..'111' = not allowed
3	0	**V_MAN_SYNC**	**Automatically synchronization source selection** '0' = automatic synchronization source selection. A TE which is synchronized to the incoming S/T signal (e.g. state F6 or F7) is chosen as synchronization source and V_SYNC_SEL is ignored. '1' = manual synchronization source selection. V_SYNC_SEL is used for synchronization source. The current synchronization source can be read from V_RD_SYNC_SRC in register R_BERT_STA.
4	0	**V_AUTO_SYNCI**	**Enable SYNC_I as synchronization signal** In addition to the line interface synchronization pulse FSC_RX, SYNC_I can be taken for synchronization. '0' = SYNC_I is not used for synchronization '1' = SYNC_I is automatically used for synchronization if no FSC_RX pulse is detected
5	0	**V_D_MERGE_TX**	All 4 D-channels are taken from one byte in TX direction.
6	0	**V_E_MERGE_RX**	All 4 E-channels are combined into one Byte in RX direction.
7	0	**V_D_MERGE_RX**	All 4 D-channels are combined into one Byte in RX direction.

R_CI_TX	(w)	(Reset group: H, 0, 2)	0x28

C/I-channel of the GCI interface

Bits	Reset value	Name	Description
5..0	0	V_GCI_C	**Command bits of the C/I-channel** These bits are continously send in the C/I-channel. C/I-channel length can either be 4 bit or 6 bit according to V_MON_CI6 setting. Bits [5,4] are ignored when the C/I-channel length is 4 bit.
7..6	0	(reserved)	Must be '00'.

R_GCI_CFG0	(w)	(Reset group: H, 0, 2)	0x29

GCI interface configuration, register 0

Bits	Reset value	Name	Description
0	0	**V_MON_END**	**End of command flag for the monitor channel** The transmitted monitor command ends after the next monitor byte.
1	0	**V_MON_SLOW**	**Transmission speed of the monitor channel** '0' = the next monitor byte is sent immediately after the receive handshake bit gets high '1' = the next monitor byte is sent after a high-to-low transition of the receive handshake bit
2	0	**V_MON_DLL**	**Enable *double last look criterion*** '0' = a received monitor byte is accepted at once '1' = a monitor byte is accepted only if it is received twice
3	0	**V_MON_CI6**	**Expand C/I-channel width to 6 bits** '0' = 4 bit C/I-channel '1' = 6 bit C/I-channel (no D-channel transmission)
4	0	**V_GCI_SWAP_TXHS**	**Swap handshake bits for transmitted monitor bytes** '0' = normal operation (XHFC-2S4U/4SU is GCI master) '1' = handshake bits MR_{TX} and MX_{TX} are swapped (XHFC-2S4U/4SU is GCI slave) **Note:** This bit must be set when XHFC-2S4U/4SU operates in GCI slave mode. Bits V_GCI_SWAP_RXHS and V_GCI_SWAP_STIO in this register must also be set in this case.
5	0	**V_GCI_SWAP_RXHS**	**Swap handshake bits for received monitor bytes** '0' = normal operation (XHFC-2S4U/4SU is GCI master) '1' = handshake bits MR_{RX} and MX_{RX} are swapped (XHFC-2S4U/4SU is GCI slave) **Note:** This bit must be set when XHFC-2S4U/4SU operates in GCI slave mode. Bits V_GCI_SWAP_TXHS and V_GCI_SWAP_STIO in this register must also be set in this case.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
6	0	**V_GCI_SWAP_STIO**	**Swap STIO pins for monitpor bytes and C/I-channel** '0' = normal operation (XHFC-2S4U / 4SU is GCI master) '1' = swap STIO1 and STIO2 for monitor bytes and C/I-channel (XHFC-2S4U / 4SU is GCI slave) **Note:** This bit must be set when XHFC-2S4U / 4SU 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.
7	0	**V_GCI_EN**	**Enable GCI interface function** 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.

R_GCI_CFG1	(w)	(Reset group: H, 0, 2)	0x2A

GCI interface configuration, register 1

Bits	Reset value	Name	Description
4..0	0	**V_GCI_SL**	**Slot group of the GCI interface** The GCI interface allocates four PCM time slots $4 \cdot V\_GCI\_SL..4 \cdot V\_GCI\_SL + 3$. Monitor bytes, C/I-channel and handshake bits are automatically mapped to their dedicated position. B1-, B2- and D-channels must be assigned according to the GCI needs with normal PCM slot assigner programming (registers R_SLOT and A_SL_CFG). 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) **Note:** Normally, GCI requires PCM30 data rate which results in valid bitmap values 0..7. When PCM64 or PCM128 is used with GCI functionality, 0..15 or 0..31 can be chosen.
7..5	0	(reserved)	Must be '000'.

R_MON_TX	(w)	(Reset group: H, 0, 2)	0x2B

Monitor data byte

Bits	Reset value	Name	Description
7..0	0	**V_MON_TX**	**Monitor data byte to be transmitted**

6.9.2 Read only registers

R_F0_CNTL	(r)	(Reset group: H, 0, 1)	0x18

F0IO pulse counter, low byte

Bits	Reset value	Name	Description
7 . . 0	0x00	**V_F0_CNTL**	**Low byte (bits 7 . . 0) of the 125 µs time counter** This register should be read first to latch the value of register R_F0_CNTH.

R_F0_CNTH	(r)	(Reset group: H, 0, 1)	0x19

F0IO pulse counter, high byte

Bits	Reset value	Name	Description
7 . . 0	0x00	**V_F0_CNTH**	**High byte (bits 15 . . 8) of the 125 µs time counter** The low byte should be read first (see register R_F0_CNTL)

R_SL_MAX	(r)	(Reset group: –)	0x1D

Number of PCM time slots

Bits	Reset value	Name	Description
7 . . 0		**V_SL_MAX**	**Number of last PCM time slot** PCM time slots are numbered 0 . . V_SL_MAX. Master mode: Four values are possible due to V_PCM_DR setting in register R_PCM_MD1 (31, 63, 127 or 11) Slave mode: Any value in the range 0 . . 127 is possible due to PCM master configuration

R_CI_RX	(r)	(Reset group: –)	0x28

CI-channel of the GCI interface

Bits	Reset value	Name	Description
5..0		V_GCI_I	**Indication bits of the C/I-channel** These bits are continously 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.
7..6		(reserved)	

R_GCI_STA	(r)	(Reset group: H, 0, 2)	0x29

Status register of the GCI interface

Bits	Reset value	Name	Description
0	0	**V_MON_RXR**	**Monitor receiver ready** A monitor byte has been received and can be read from register R_MON_RX.
1	1	**V_MON_TXR**	**Monitor transmitter ready** A monitor byte has been send and the next byte can be written into register R_MON_TX.
2	1	**V_GCI_MX**	**Status of MX handshake bit** This bit shows the current status of the MX_{RX} handshake bit (normally not used, for test purposes only).
3		**V_GCI_MR**	**Status of MR handshake bit** This bit shows the current status of the MR_{RX} handshake bit (normally not used, for test purposes only).
4	0	**V_GCI_RX**	**Receiving monitor byte** '0' = monitor byte transmission from GCI device to XHFC-2S4U / 4SU is idle '1' = monitor byte transmission from GCI device to XHFC-2S4U / 4SU is currently active
5	0	**V_GCI_ABO**	**Receiver abort** '0' = normal operation '1' = receiver aborted the monitor byte transmission
7 . . 6		(reserved)	

R_MON_RX	(r)	(Reset group: –)	0x2A

Monitor data byte

Bits	Reset value	Name	Description
7 . . 0		**V_MON_RX**	**Received monitor data byte**

6.9.3 Read / write register

A_SL_CFG [SLOT]	**(r*/w)**	(Reset group: H, 0, 2)	**0xD0**

HFC-channel assignment for the selected PCM time slot and PCM output buffer configuration

With this register a HFC-channel can be assigned to the selected PCM time slot. Additionally, the PCM buffers can be configured.

Before writing this array register the PCM time slot must be selected by register R_SLOT.

Bits	Reset value	Name	Description
0	0	**V_CH_SDIR**	**HFC-channel data direction** '0' = HFC-channel for transmit data '1' = HFC-channel for receive data
5..1	0x00	**V_CH_SNUM**	**HFC-channel number** (0..31)
7..6	0	**V_ROUT**	**PCM output buffer configuration** For transmit time slots: '00' = data transmission from HFC-channel disabled, output buffers disabled '01' = loop PCM data internally, output buffers disabled '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled For receive time slots: '00' = data transmission to HFC-channel disabled, input data is ignored '01' = loop PCM data internally '10' = receive data from STIO2 '11' = receive data from STIO1 **Note:** When this bitmap is set to '01' (internal PCM loop), it is not allowed to set bit V_PCM_OD in register R_PCM_MD1 as well.

(See Section 2.2.3.2 on page 47 for details on Read* access.)

Chapter 7

Pulse width modulation (PWM) outputs

Table 7.1: *Overview of the XHFC-2S4U/4SU PWM registers*

Address	Name	Page
0x1E	R_PWM_CFG	279
0x38	R_PWM0	279
0x39	R_PWM1	280
0x46	R_PWM_MD	280

7.1 Overview

XHFC-2S4U / 4SU 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

$$T = 256 \cdot \frac{2^{3 \cdot \text{V\_PWM\_FRQ}}}{f_{\text{SYS}}} \quad .$$

The variable duty cycle

$$\frac{t_{\text{high}}}{t_{\text{low}}} = \frac{i}{256 - i} \quad , i = \text{value of R\_PWM0 or R\_PWM1}$$

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 $1/f_{\text{SYS}}$. 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 ouput signal of the PWM unit can be used for analog settings by using an external RC filter which generates a voltage that can be adapted by changing the PWM register value.

7.3 Alternative PWM usage

The PWM output lines can be programmed to generate a 16 kHz signal. This signal can be used as analog metering pulse for POTS interfaces. Each PWM output line can be switched to 16 kHz signal by setting V_PWM0_16KHZ or V_PWM1_16KHZ in register R_PWM_CFG. In this case the output characteristic is also determined by the R_PWM_MD register settings.

7.4 Register description

R_PWM_CFG	(w)	(Reset group: H, 0)	0x1E

Bits	Reset value	Name	Description
3 . . 0	0	(reserved)	Must be '0000'.
4	0	**V_PWM0_16KHZ**	**16 kHz signal on PWM0** '0' = normal PWM output due to the R_PWM0 register setting '1' = 16 kHz output on PWM0
5	0	**V_PWM1_16KHZ**	**16 kHz signal on PWM1** '0' = normal PWM output due to the R_PWM1 register setting '1' = 16 kHz output on PWM1
7 . . 6	0	**V_PWM_FRQ**	The PWM frequency is derived from the system clock f_{SYS} and can be reduced with this bitmap. '00' = PWM frequency is f_{SYS} '01' = PWM frequency is f_{SYS} / 8 '10' = PWM frequency is f_{SYS} / 64 '11' = PWM frequency is f_{SYS} / 512

R_PWM0	(w)	(Reset group: H, 0)	0x38

Modulator register for pin PWM0

Bits	Reset value	Name	Description
7 . . 0	0x00	**V_PWM0**	**PWM duty cycle** The value specifies the number of clock periods where the output signal of PWM0 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high

R_PWM1	(w)	(Reset group: H, 0)	0x39

Modulator register for pin PWM1

Bits	Reset value	Name	Description
7..0	0x00	**V_PWM1**	**PWM duty cycle** The value specifies the number of clock periods where the output signal of PWM1 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high

R_PWM_MD	(w)	(Reset group: H, 0)	0x46

PWM output mode register

Bits	Reset value	Name	Description
0	0	(reserved)	Must be '0'.
1	0	**V_WAK_EN**	**Enable WAKEUP pin** '0' = disable WAKEUP pin '1' = enable WAKEUP pin
3..2	0	(reserved)	Must be '00'.
5..4	0	**V_PWM0_MD**	**Output buffer configuration for pin PWM0** '00' = PWM output tristate (disable) '01' = PWM push / pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only
7..6	0	**V_PWM1_MD**	**Output buffer configuration for pin PWM1** '00' = PWM output tristate (disable) '01' = PWM push / pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only

Chapter 8

Bit Error Rate Test (BERT)

Table 8.1: *Overview of the XHFC-2S4U/4SU BERT registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x1B	R_BERT_WD_MD	286	0x17	R_BERT_STA	287
			0x1A	R_BERT_ECL	287
			0x1B	R_BERT_ECH	288

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-2S4U / 4SU 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. Continous '0', continous '1' or pseudo random bit patterns with one of 6 selectable sequence length's from $2^9 - 1$ bit to $2^{23} - 1$ bit can be configured with bitmap V_PAT_SEQ in register R_BERT_WD_MD. All bit sequences are defined in the ITU-T O.150 [11] and O.151 [10] specifications.

The BERT patterns are passed through the HFC-channel assigner if V_BERT_EN = '1' in register A_FIFO_CTRL[FIFO]. For this reason, either a FIFO-to-ST / U_p or a FIFO-to-PCM configuration must be selected. Furthermore, the allocated FIFO must be enabled to switch on the data path.

BERT patterns are generated if at least one FIFO has its bit V_BERT_EN set to '1'. When more than one transmit FIFO are using BERT patterns, all these patterns are generated from the same pseudo random generator. They are distributed to the FIFOs in the order of the FIFO processing sequence (see Section 3.2.3 on page 80).

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 synchonization capability. When the incoming bit stream is synchronized with the PRBG, bit V_BERT_SYNC in register R_BERT_STA is set to '1'.

A 16 bit BERT error counter is available in registers R_BERT_ECL and R_BERT_ECH. The low

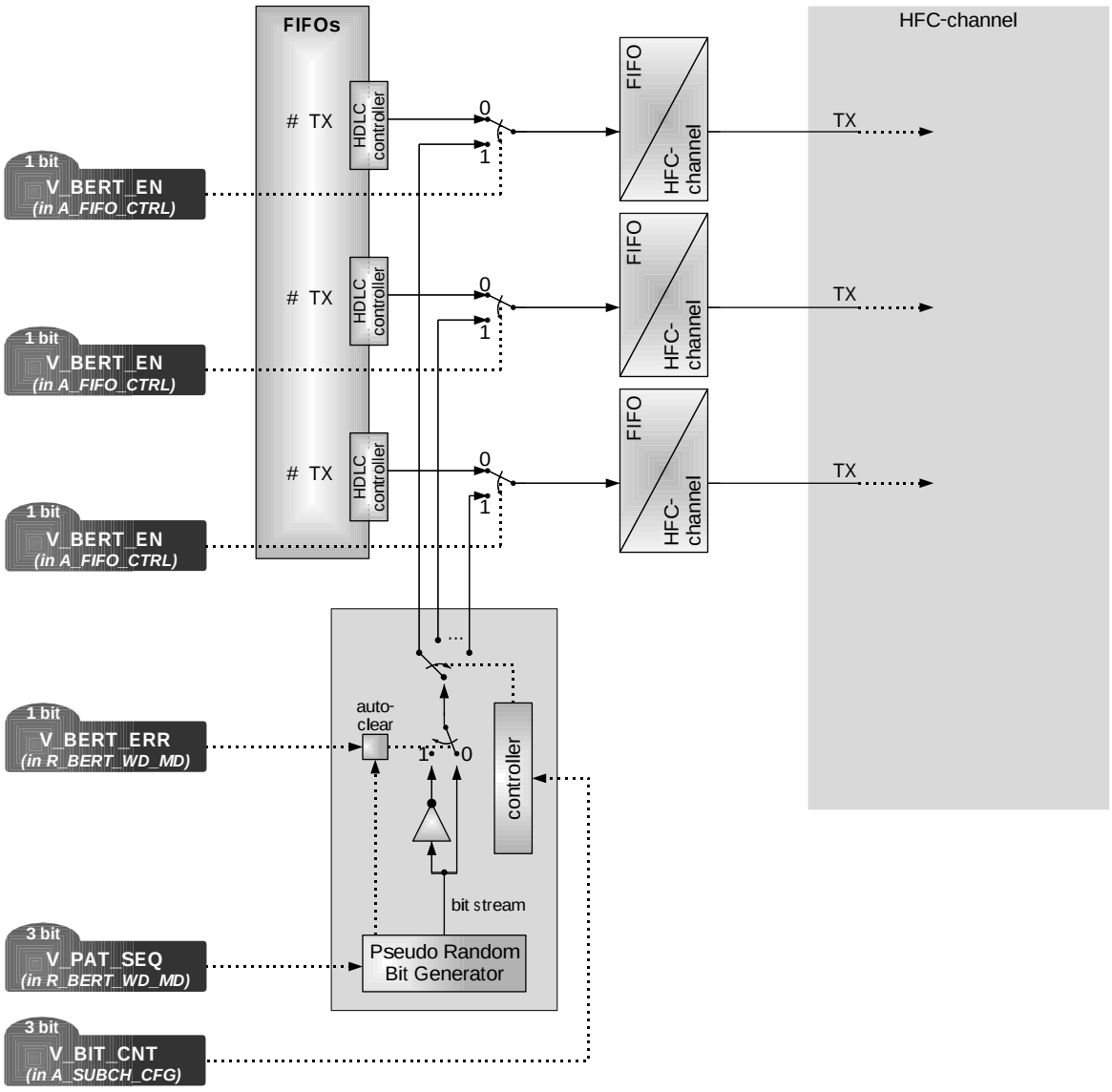


Figure 8.1: *BERT transmitter block diagram*

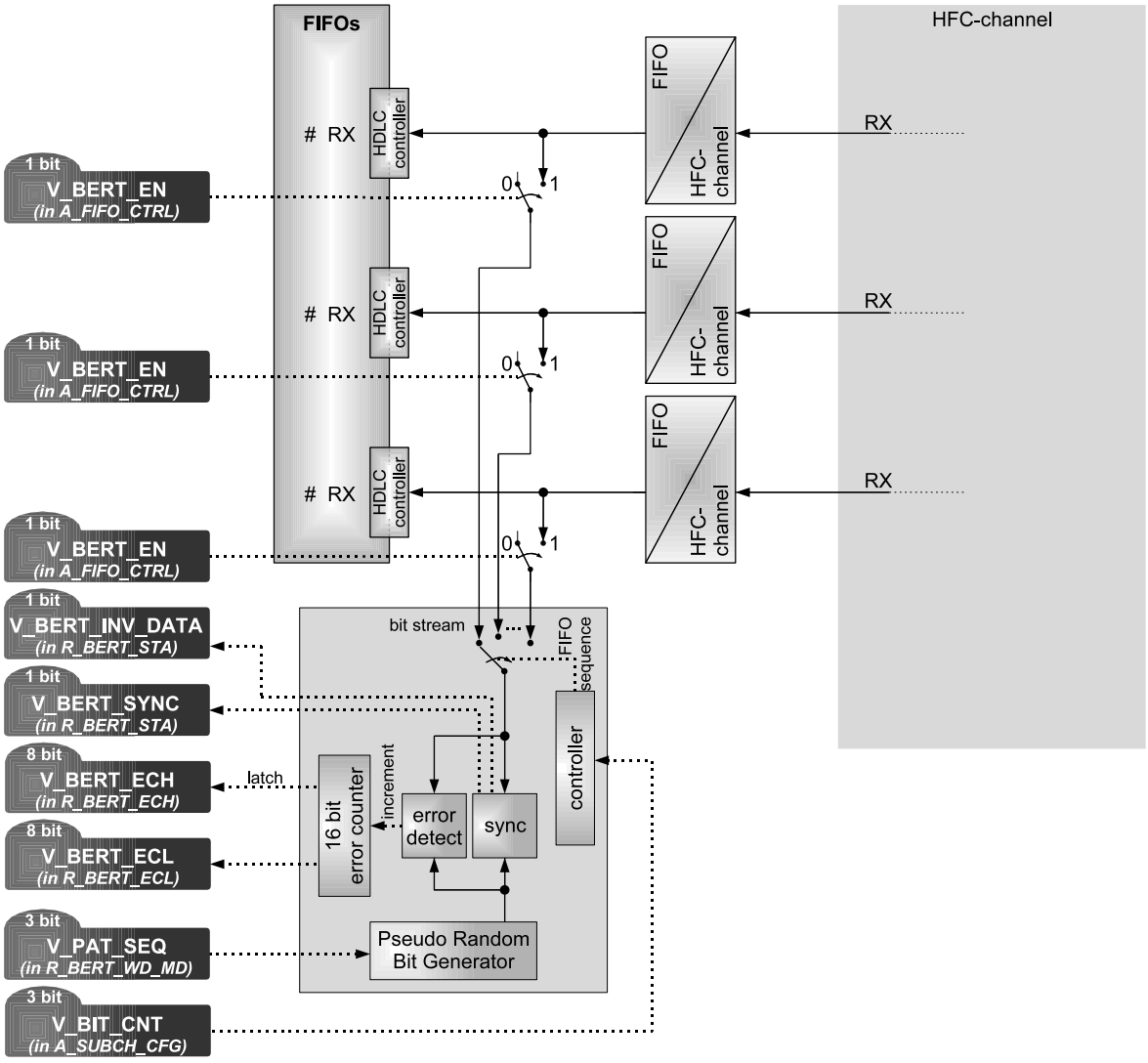


Figure 8.2: *BERT receiver block diagram*

byte R_BERT_ECL should be read first to latch the high byte. Then the high byte can be read from register R_BERT_ECH. A read access to the low byte R_BERT_ECL clears the 16 bit counter.

The BERT procedure should first wait for the synchronization state. After this, the BERT error counter should be cleared by reading register R_BERT_ECL.

Received BERT data is passed through the HFC-channel assigner if V_BERT_EN = '1' in register A_FIFO_CTRL[FIFO]. For this reason, either a FIFO-to-ST / U_p or a FIFO-to-PCM configuration must be selected. Furthermore, the allocated FIFO must be enabled to switch on the data path. Received BERT data is stored in the FIFO but it needs not to be read out. Received BERT data is collected from all FIFOs which have V_BERT_EN = '1' in the order of the FIFO processing sequence (see Section 3.2.3 on page 80).

Subchannel processing can be used together with the *Bit Error Rate Test*. Then V_BIT_CNT bits taken passed to the BERT receiver.

Inverted BERT data is automatically detected and can be checked with V_BERT_INV_DATA in register R_BERT_STA.

The automatic synchronization works only if the bit error rate is less than $4 \cdot 10^{-2}$. Synchronization state will not be achieved with a higher error rate. It is lost when many bit errors occur during a short time period. In this case, the re-synchronization starts automatically and a high bit error counter value indicates that a re-synchronization might has happened.

8.4 Register description

8.4.1 Write only registers

R_BERT_WD_MD	(w)	(Reset group: H, 0)	0x1B

Bit error rate test (BERT) and watchdog mode

Bits	Reset value	Name	Description
2..0	0	V_PAT_SEQ	**Continuous '0' / '1' or pseudo random pattern sequence for BERT** '000' = continuous '0' pattern '001' = continuous '1' pattern '010' = sequence length $2^9 - 1$ bits '011' = sequence length $2^{10} - 1$ bits '100' = sequence length $2^{15} - 1$ bits '101' = sequence length $2^{20} - 1$ bits '110' = sequence length $2^{20} - 1$ bits, but maximal 14 bits are zero '111' = pseudo random pattern seq. $2^{23} - 1$ **Note:** These sequences are defined in ITU-T O.150 and O.151 specifications.
3	0	V_BERT_ERR	**BERT error** Generates one error bit in the BERT data stream '0' = no error generation '1' = generates one error bit This bit is automatically cleared.
4	0	(reserved)	Must be '0'.
5	0	V_AUTO_WD_RES	**Automatic watchdog timer reset** '0' = watchdog is only reset by V_WD_RES '1' = watchdog is reset after every access to the chip
6	0	V_WD_EN	**Watchdog timer enable** '0' = watchdog timer is disabled '1' = watchdog timer is enabled
7	0	V_WD_RES	**Watchdog timer reset** '0' = no action '1' = manual watchdog timer reset This bit is automatically cleared.

8.4.2 Read only registers

R_BERT_STA	(r)	(Reset group: H, 0, 1)	0x17

Bit error rate test status

Bits	Reset value	Name	Description
2..0	0	V_RD_SYNC_SRC	**Synchronization source selection** Reports which line interface is used as synchronization source. Every line interface can be either in S/T or Up mode. The SYNC_I signal can be used as synchronization source, alternatively. '000' = line interface 0 '001' = line interface 1 '010' = line interface 2 '011' = line interface 3 '100' = SYNC_I signal '101'..'111' = not used
3		(reserved)	
4	0	V_BERT_SYNC	**BERT synchronization status** '0' = BERT not synchronized to input data '1' = BERT synchronized to input data
5	0	V_BERT_INV_DATA	**BERT data inversion** '0' = BERT receives normal data '1' = BERT receives inverted data
7..6		(reserved)	

R_BERT_ECL	(r)	(Reset group: H, 0, 1)	0x1A

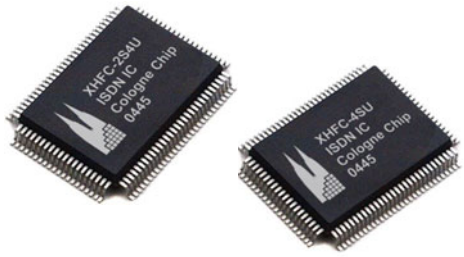
BERT error counter, low byte

Bits	Reset value	Name	Description
7..0	0x00	V_BERT_ECL	**Bits 7..0 of the BERT error counter** This register should be read first to latch the value of register R_BERT_ECH. **Note:** The BERT counter is cleared after reading this register.

R_BERT_ECH	(r)	(Reset group: H, 0, 1)	0x1B

BERT error counter, high byte

Bits	Reset value	Name	Description
7..0	0x00	**V_BERT_ECH**	**Bits 15..8 of the BERT error counter** **Note:** Low byte should be read first (see register R_BERT_ECL).



Chapter 9

Clock, PLL, reset, interrupt, timer and watchdog

Table 9.1: *Overview of clock, PLL, reset, timer and watchdog registers*

Write only registers:			Read only registers:			Read / write registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**	**Address**	**Name**	**Page**
0x00	R_CIRM	308	0x10	R_IRQ_OVIEW	316	0x51	R_PLL_P	325
0x02	R_CLK_CFG	310	0x11	R_MISC_IRQ	318	0x52	R_PLL_N	325
0x11	R_MISC_IRQMSK	311	0x12	R_SU_IRQ	319	0x53	R_PLL_S	325
0x12	R_SU_IRQMSK	312	0x1C	R_STATUS	320			
0x13	R_IRQ_CTRL	313	0x20	R_FIFO_BL0_IRQ	321			
0x1A	R_TI_WD	314	0x21	R_FIFO_BL1_IRQ	322			
0x50	R_PLL_CTRL	315	0x22	R_FIFO_BL2_IRQ	323			
			0x23	R_FIFO_BL3_IRQ	324			
			0x50	R_PLL_STA	324			

9.1 Clock

9.1.1 Clock output

XHFC-2S4U / 4SU supply 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-2S4U / 4SU is used can be clocked. The left part of Figure 9.1 shows the block diagram of CLK_OUT generation.

After reset, the clock output frequency

$$f(\text{CLK\_OUT}) = \frac{1}{8} f(\text{OSC\_OUT})$$

is available at pin CLK_OUT .

Two clock sources are available for f_{SRC}. V_CLK_F1 = '0' in register R_CLK_CFG selects the clock oscillator. Alternatively, pin F1_1 is used as clock source when V_CLK_F1 = '1'.

The clock output frequency depends on the programming bits as follows:

$$f(\text{CLK\_OUT}) = \begin{cases} f_{\text{SRC}} & ; \quad \text{V\_CLKO\_HI = '1' and V\_CLKO\_PLL = '0'} \\ \frac{1}{8} f_{\text{SRC}} & ; \quad \text{V\_CLKO\_HI = '0' and V\_CLKO\_PLL = '0'} \\ f_{\text{out}} & ; \quad \text{V\_CLKO\_PLL = '1'} \end{cases}$$

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-2S4U / 4SU use several internal clock frequencies. They are generated as shown in the right part of Figure 9.1.

The system clock f_{SYS} is derived either from the OSC_OUT clock or from the internal PLL output clock. When the oscillator frequency is either 24.576 MHz or 49.152 MHz, V_CLK_PLL should be left in its reset state '0'. With any other oscillator frequency the internal PLL must be used to generate the required XHFC-2S4U / 4SU clock f_{SYS} and V_CLK_PLL must be set to '1' in register R_CLK_CFG.

Both the line interface clock f_{SU} and the PCM clock f_{PCM} must be set up to achieve 12.288 MHz and 49.152 MHz respectively. This is done with bit V_SU_CLK in register R_CTRL and bitmap V_PCM_CLK in register R_CLK_CFG.

The internal clocks f_{SYS}, f_{SU} and f_{PCM} can be switched off with V_CLK_OFF = '1' in register R_CIRM. Any write access to the host processor interface or a high level at pin WAKEUP restarts the clock distribution.

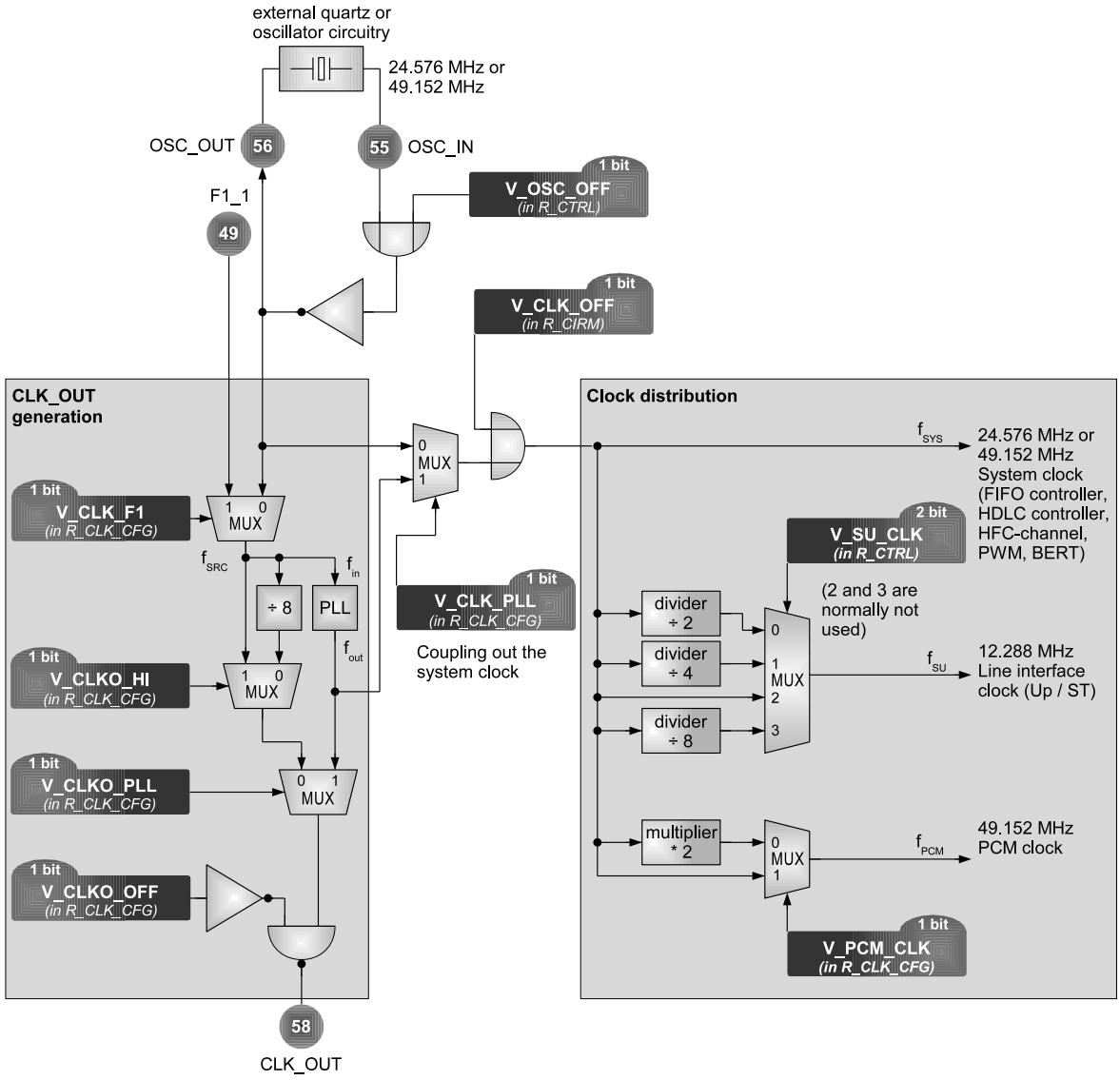


Figure 9.1: *CLK_OUT generation and clock distribution*

> **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 beween XHFC-2S4U / 4SU accesses.

9.1.3 Clock oscillator circuitry

There are different ways to provide the internal clocks of XHFC-2S4U / 4SU. This section describes the Pierce oscillator circuitry, gives a hint to 3rd overtone oscillator and the usage of crystal oscillator circuitries.

9.1.3.1 Frequency accuracy

ISDN applications need an exact clock frequency. By the ISDN specification a precision of ± 100 ppm is minimum requirement for passing the ISDN type approval. In respect to temperature dependence and ageing behavior a crystal with ± 50 ppm is recommended.

9.1.3.2 Pierce oscillator

A typical clock oscillator circuitry using a 24.576 MHz crystal is shown in Figure 9.2. This Pierce oscillator is very popular for clock generation and is widely known from literature.

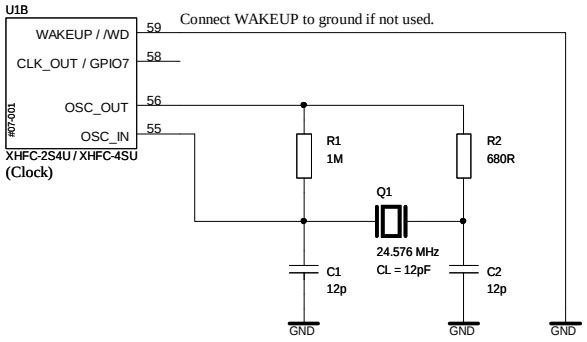


Figure 9.2: *Standard XHFC-2S4U / 4SU quartz circuitry*

The feedback resistor R1 determines the DC operation point and is typically in the range 100 kΩ . . 10 MΩ 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

$$C_L = \frac{C1 \cdot C2}{C1 + C2} + C_S$$

where C_S is the stray capacitance. It is given by the input and output capacitances of the inverter and the shunt capacitance between the crystal terminals. Typically, C1 and C2 are chosen to be equal.

Finally, the resistor R2 is chosen to be roughly equal to the capacitive reactance of C2 at the frequency of oscillation.

$$\mathsf{R2} \sim \frac{1}{2\pi f_{\mathsf{Q1}} \cdot \mathsf{C2}}$$

The minimum value of R2 depends on the recommended power consumption of the crystal. A too small value may damage the crystal or shorten the lifetime. When R2 is too large, the oscillation might not start. As XHFC-2S4U / 4SU has a bufffered inverter between pins OSC_IN and OSC_OUT the value of R2 can be increased. A factor of about $2\,..\,3$, e.g., is well.

The circuitry shown in Figure 9.2 is based on a crystal with $C_{\mathrm{L}} = 12\,\mathrm{pF}$ and a stray capacitance of $C_{\mathrm{S}} = 6\,\mathrm{pF}$. This leads to

$$\mathsf{C1} = \mathsf{C2} = 2 \cdot (C_{\mathrm{L}} - C_{\mathrm{S}}) = 12\,\mathrm{pF}$$

and

$$\mathsf{R2} = (1\,..\,3) \cdot \frac{1}{2\pi f_{\mathsf{Q1}} \cdot \mathsf{C2}} = (1\,..\,3) \cdot 540\,\Omega \sim 680\,\Omega \ .$$

Please note !

The here shown dimensioning of the oscillator circuitry is only an example and depends on the used crystal as well as on the particular board design. In general it is recommanded to check oscillation build-up, power consumption of the crystal and the so-called safety factor within the particular design.

The specified drive level should not be put to the extreme to avoid early crystal ageing. Typically, within a good dimensioned circuitry, a low load capacitance C_L is a condition for a low drive level.

9.1.3.3 3rd overtone oscillator

A different oscillator frequency with double frequency 49.152 MHz can be used alternatively. For this a 3rd overtone crystal or a clock oscillator can be used.

9.1.3.4 Crystal oscillator circuitry

It is possible to feed the OSC_IN input of XHFC-2S4U / 4SU with a standard 3.3 V crystal oscillator. The input switching level is close to $V_{\mathrm{DD}}/2$ (CMOS level) and XHFC-2S4U / 4SU can accept at least a duty cycle of 45 % high/55 % low to 55 % high/45 % low.

9.1.3.5 Several XHFC-2S4U / 4SU with a single oscillator circuitry

When several XHFC-2S4U / 4SU are used within an application, only one oscillator circuitry is required. Pin CLK_OUT can be used to distribute the clock to all XHFC-2S4U / 4SU as shown in Figure 9.3.

Register setup:

R_CLK_CFG	: V_CLK_F1	= 0	oscillator input OSC_IN is used
	: V_CLKO_HI	= 1	high frequency, no divider
	: V_CLKO_PLL	= 0	divider output clock or PLL input clock is used
	: V_CLKO_OFF	= 0	clock output pin CLK_OUT is enabled

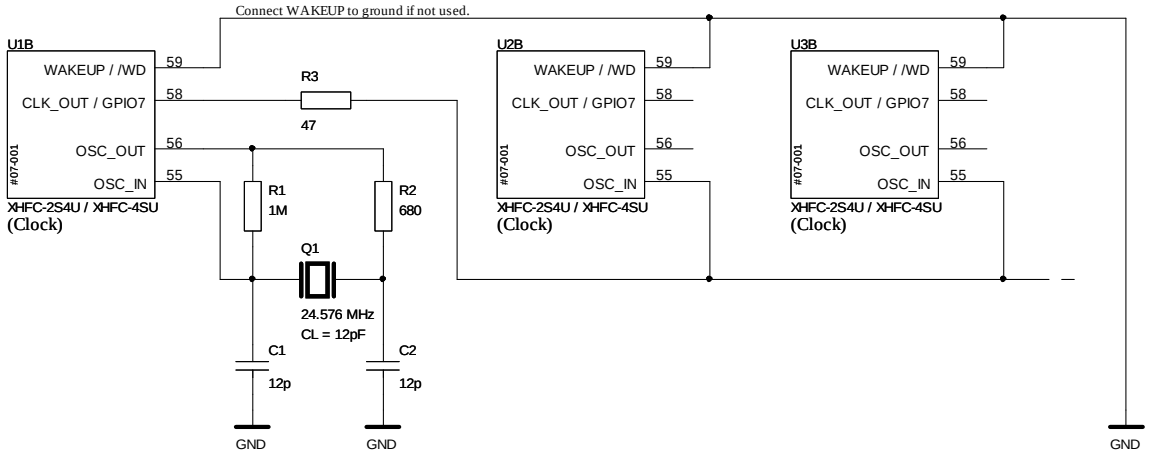


Figure 9.3: *Clock distribution with only one quartz circuitry*

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-2S4U / 4SU 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^{TM} technology introduced by Cologne Chip [1].

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 $1 \ldots 256$ each
- Deterministic clock-to-clock jitter typical 120 ps
- f_{out} duty cycle $40\% \ldots 60\%$
- No external loop filter or capacity needed
- Very short lock time (worst case 2000 periods of f_{ref})

9.2.2 PLL structure

The PLL consists of a predivider, the PLL circuitry and a post-scaler. This structure is shown in Figure 9.4.

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

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

with $N =$ V_PLL_N $= 5 \ldots 255$ ($0 \ldots 4$ 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 $1 \ldots 256$.

[1] Detailed information about the DIGICC^{TM} technology and the here used PLL is documented in [1, 2]. It is *not necessary to read these documents* in order to understand this data sheet.

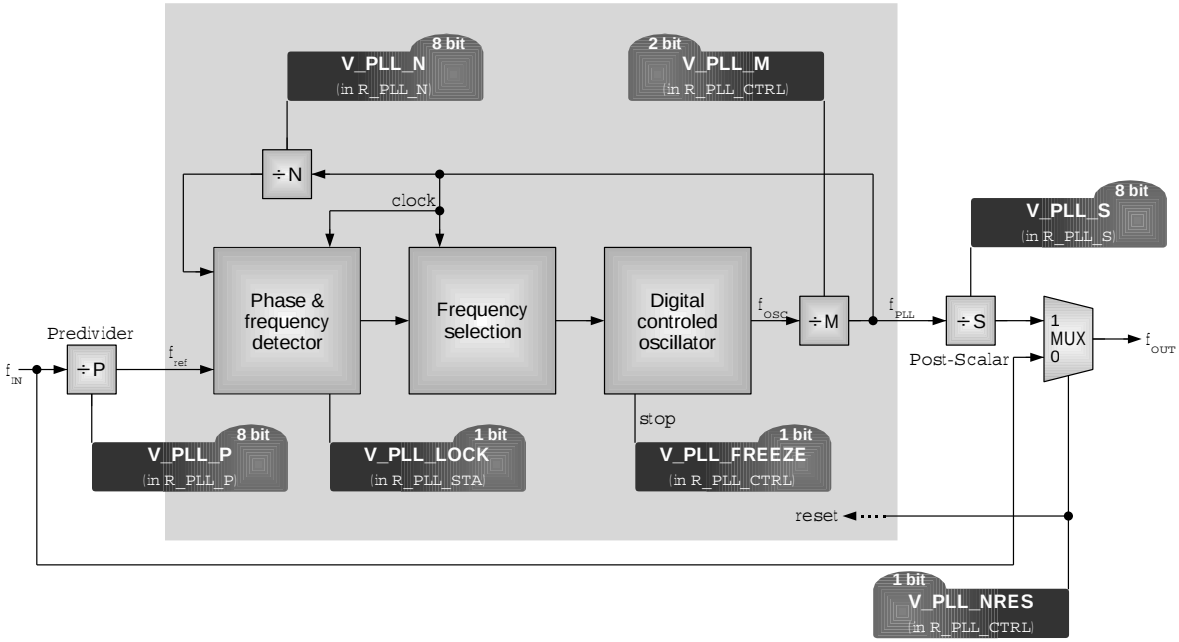


Figure 9.4: *PLL block diagram*

An additional divider with $M = \mathsf{V\_PLL\_M} + 1 = 1..4$ 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

$$f_{\mathrm{PLL}} = \frac{f_{\mathrm{osc}}}{M}$$

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. $f_{\mathrm{out}} = f_{\mathrm{in}}$. 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 $f_{\mathrm{out}} = 0$ 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 $(\mathsf{P},\mathsf{M},\mathsf{N},\mathsf{S})$ is given for often used frequencies. Approximate solutions have additional information about the f_{out} offset in parts per million and absolute value.

If other parameter sets are required, they can be calculated with the formulars given in section 9.2.2. Alternatively, the Cologne Chip support team will help you choosing suitable parameter sets for your application.

Please note !

The ISDN specification provides a clock precision of at least ±100 ppm. PLL output offset must be added to the crystal precision. This must be taken into account when a PLL configuration setup is chosen from the examples in Tables 9.2 and 9.3.

Table 9.2: *PLL setup examples (P, M, N, S) with ISDN related frequencies for $\mathbf{f}_{out}$, (approximations have additional information about f_{out} offset)*

	$\mathbf{f}_{out}$ (MHz)			
$\mathbf{f}_{in}$ (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 $\mathbf{f}_{in}$, (approximations have additional information about f_{out} offset)*

	$\mathbf{f_{in}}$ (MHz)			
$\mathbf{f_{out}}$ (MHz)	**7.68**	**12.288**	**24.576**	**49.152**
1.8432	(1,2,6,25) exact	(1,1,6,40) exact	(2,1,6,40) exact	(4,1,6,40) exact
3.579545	(59,4,110,4) **+33 ppm**, +117 Hz	(10,1,67,23) **+1 ppm**, +3 Hz	(20,1,67,23) **+1 ppm**, +3 Hz	(40,1,67,23) **+1 ppm**, +3 Hz
6	(2,1,25,16) exact	(16,1,125,16) exact	(32,1,125,16) exact	(64,1,125,16) exact
7.68	(1,2,5,5) exact	(1,1,5,8) exact	(2,1,5,8) exact	(4,1,5,8) exact
10.7	(89,4,248,2) **+22 ppm**, +225 Hz	(89,4,155,2) **+22 ppm**, +225 Hz	(89,2,155,4) **+22 ppm**, +225 Hz	(89,1,155,8) **+22 ppm**, +225 Hz
12	(2,1,25,8) exact	(16,1,125,8) exact	(32,1,125,8) exact	(64,1,125,8) exact
12.288	(1,1,8,5) exact	(1,1,5,5) exact	(2,1,5,5) exact	(3,1,6,8) exact
14.31818	(59,4,110,1) **+33 ppm**, +465 Hz	(23,1,134,5) **+1 ppm**, +12 Hz	(23,1,67,5) **+1 ppm**, +12 Hz	(46,1,67,5) **+1 ppm**, +12 Hz
16	(2,1,25,6) exact	(16,1,125,6) exact	(32,1,125,6) exact	(64,1,125,6) exact
24.576	(5,4,16,1) exact	(1,1,6,3) exact	(2,1,6,3) exact	(3,1,6,4) exact
25	(47,4,153,1) **+35 ppm**, +852 Hz	(29,4,59,1) **-12 ppm**, -276 Hz	(29,2,59,2) **-12 ppm**, -276 Hz	(29,1,59,4) **-12 ppm**, -276 Hz
32.768	(5,1,64,3) exact	(1,1,8,3) exact	(2,1,8,3) exact	(3,1,6,3) exact
33	(37,3,159,1) **+99 ppm**, +3244 Hz	(89,3,239,1) **-58 ppm**, -1888 Hz	(89,1,239,2) **-58 ppm**, -1888 Hz	(71,1,143,3) **-38 ppm**, -1240 Hz
48	(2,1,25,2) exact	(16,1,125,2) exact	(32,1,125,2) exact	(64,1,125,2) exact
49.152	(5,2,32,1) exact	(1,1,8,2) exact	(2,1,8,2) exact	(3,1,6,2) exact
66	(27,1,232,1) **-135 ppm**, -8889 Hz	(35,1,188,1) **+63 ppm**, +4115 Hz	(89,1,239,1) **-58 ppm**, -3776 Hz	(178,1,239,1) **-58 ppm**, -3776 Hz

9.3 Reset

XHFC-2S4U / 4SU has a level sensitive reset input at pin 23 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-2S4U / 4SU enters an initialization sequence. Its duration depends on the number of FIFOs and has a maximum length of 40 µs with $f_{SYS} = 24.576$ MHz. When the initialization process is finished, bit V_BUSY in register R_STATUS changes from '1' to '0'.

PCM initialization takes 149 µs 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-2S4U / 4SU has 4 different software resets which means that the registers are assigned to so-called reset groups.

The FIFO registers, PCM registers and ST / U_p registers belong to reset groups 1 . . 3 and can be reset independently with the bits of register R_CIRM which are listed in Table 9.4.

A global software reset puts all registers of reset group 0 back to their default value and implies reset groups 1 . . 3 as well. It is very similar to the hardware reset, except a few registers which have only hardware reset (see register list from page 21).

The reset bits must be set and cleared by software.

A hardware reset implies all reset groups 0 . . 3, of course.

Table 9.4: *XHFC-2S4U / 4SU reset groups*

Reset name	Reset group	Register bit	Description
Hardware reset	H	–	Hardware reset initiated by /RES input pin. The hardware reset implies reset of all registers of reset groups 0 . . 3 as well.
Global Software reset	0	V_SRES	The global software reset, which is similar to the hardware reset, restores the default values to the most registers. The global software reset implies reset of all registers of reset groups 1 . . 3 as well.
HFC reset	1	V_HFC_RES	Reset for all FIFO registers of XHFC-2S4U / 4SU.
PCM reset	2	V_PCM_RES	Reset for all PCM registers of XHFC-2S4U / 4SU.
ST / U_p reset	3	V_SU_RES	Reset for all ST / U_p registers of XHFC-2S4U / 4SU.

Information about the allocation of the registers to the different reset groups can be found in the register list on pages 22 and 24. Some registers are allocated to more than one reset group, some have only hardware reset, and some have no default value at all.

9.4 Interrupt

9.4.1 Common features

XHFC-2S4U / 4SU 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 22 is the interrupt output line. After reset, all interrupts are disabled. The interrupt line must be enabled with V_GLOB_IRQ_EN set to '1' in register R_IRQ_CTRL. The polarity of the interrupt signal can be changed from *active low* to *active high* with bitmap V_IRQ_POL in the same register. Please note, that the interrupt line cannot be shared with active high polarity.

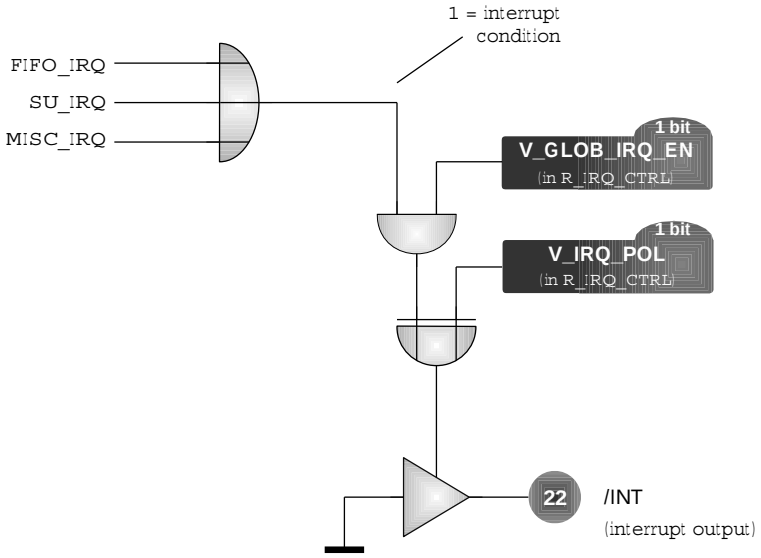


Figure 9.5: *Interrupt output*

9.4.2 ST / U_p interface interrupt

Every line interface can have its own interrupt capability to indicate a state change condition. The interrupt mask has to be programmed in register R_SU_IRQMSK.

When an ST / U_p interface interrupt occured, the corresponding line interface can be determined by reading register R_SU_IRQ. This register contains the state change condition even if the interrupts are disabled.

9.4.3 FIFO interrupt

FIFO interrupts can be enabled to indicate the status for every FIFO individually.

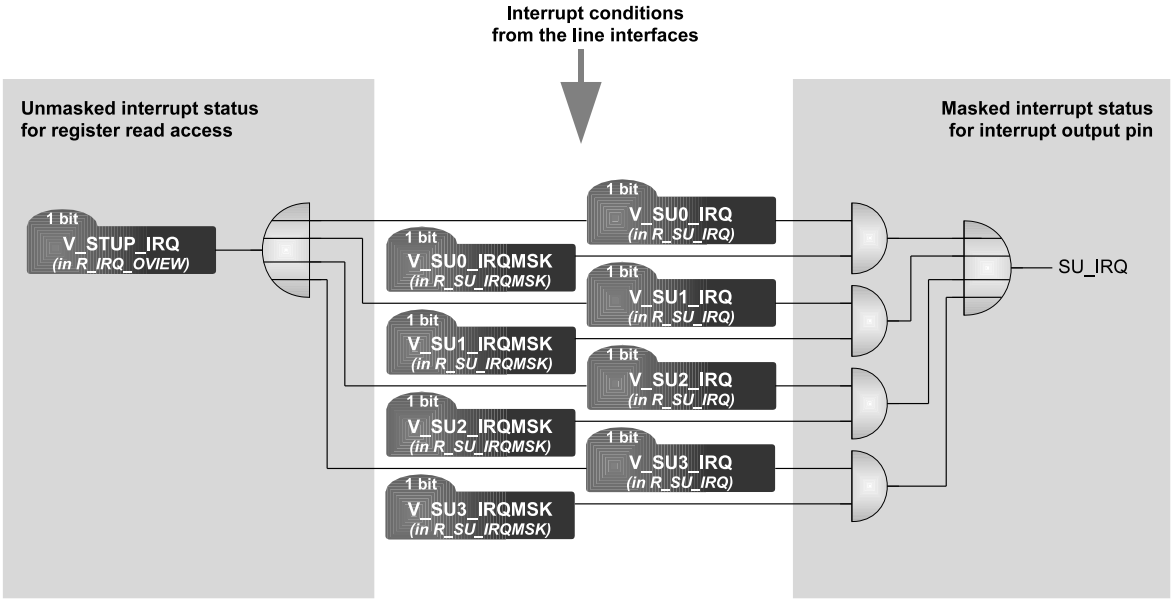


Figure 9.6: *ST / U_p interface interrupt*

The interrupt status of all 32 FIFOs can be read from registers R_FIFO_BL0_IRQ . . R_FIFO_BL3_IRQ. All FIFOs are organized in four blocks with 8 FIFOs each. Every block has an overview bit V_FIFO_BL0_IRQ . . V_FIFO_BL3_IRQ in register R_IRQ_OVIEW.

FIFO interrupt status bits in registers R_FIFO_BL0_IRQ . . R_FIFO_BL3_IRQ are only set, when V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL as shown in Figure 9.7.

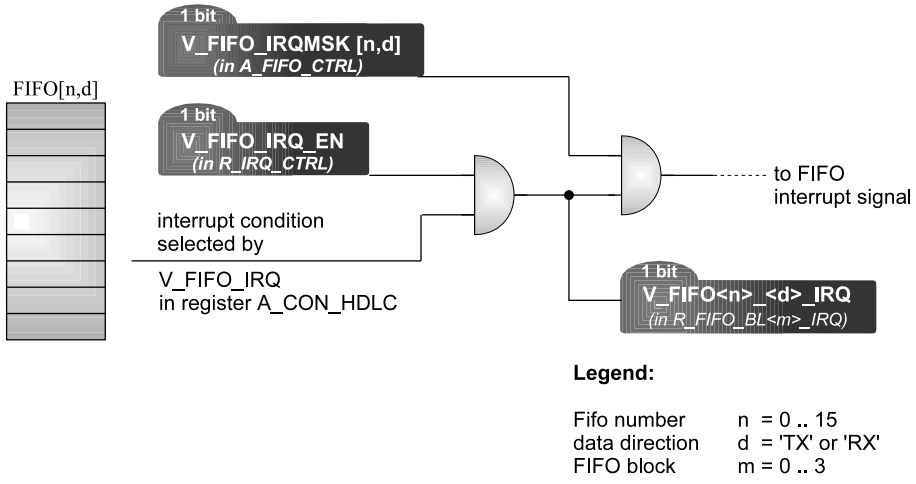


Figure 9.7: *Enable FIFO interrupt condition with V_FIFO_IRQ_EN*

FIFO in transparent mode (V_HDLC_TRP = '1'): An interrupt occurs due to the setting of V_FIFO_IRQ in register A_CON_HDLC.

HDLC mode without mixed interrupt mode (V_HDLC_TRP = '0' and V_MIX_IRQ = '0'): An interrupt occurs at *end of frame* condition (which leads to a frame counter increment) or after a FIFO underrun condition.

HDLC mode and mixed interrupt mode (V_HDLC_TRP = '0' and V_MIX_IRQ = '1'):

An interrupt occurs both at *end of frame* condition (which leads to a frame counter increment) or after a FIFO underrun condition and due to the setting of V_FIFO_IRQ in register A_CON_HDLC.

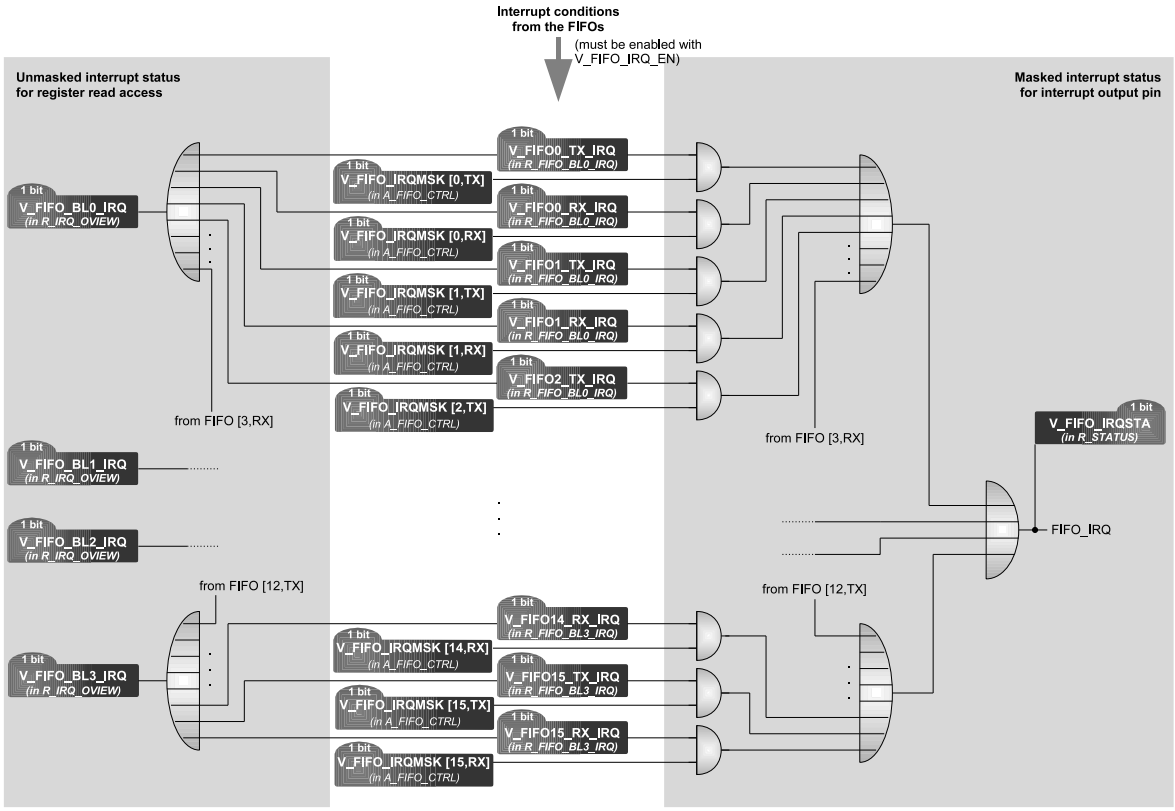


Figure 9.8: *FIFO interrupt*

9.4.4 Miscelleanous interrupts

Seven miscelleanous interrupts are available to report important XHFC-2S4U / 4SU status. Figure 9.9 shows the block diagram of these interrupt capabilities.

An overview bit V_MISC_IRQ can be read from register R_IRQ_OVIEW.

9.4.4.1 Line interface frequency slip interrupt

The frame synchronization signal can either be the F0IO or the AF0 signal as shown in Figure 5.16 on page 190. The actual selection of a line interface can be read from bit V_SU_AF0 in register A_SU_STA or together for all 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.

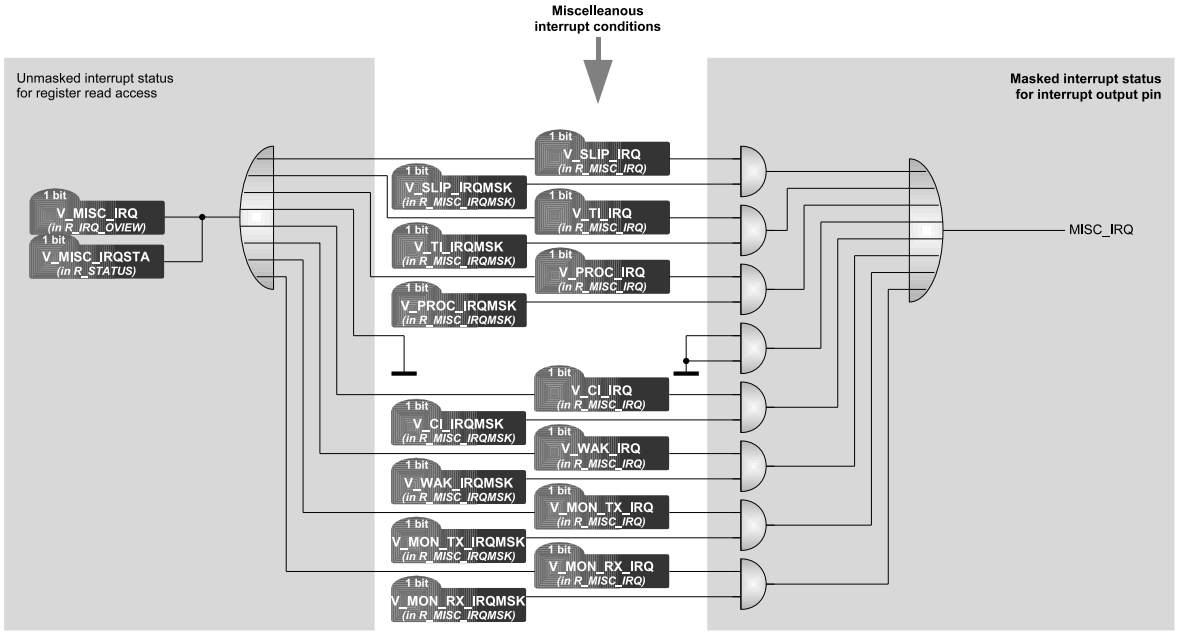


Figure 9.9: *Miscelleanous interrupts*

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 189.

Unused line interfaces might trigger interrupts because of their free running 8 kHz frame signal FSC_RX. These unwanted interrupt events can be avoided when the unused line interfaces are switched into NT mode with V_SU_MD = '1' in register A_SU_CTRL0. Moreover, V_SU_SYNC_NT must be in its default state '0' in register A_SU_CTRL2.

Please note, that the upper line interfaces 2 and 3 of XHFC-2S4U should also be switched into U_p - NT mode even in S/T applications where line interfaces 0 and 1 are used in S/T mode.

9.4.4.2 Timer interrupt

XHFC-2S4U / 4SU includes a timer with interrupt capability. The timer counts F0IO pulses, i.e. it is incremented every 125 µs.

A timer event is indicated with V_TI_IRQ = '1' in register R_MISC_IRQ. This event generates an interrupt if the mask bit V_TI_IRQMSK is set to '1' in register R_MISC_IRQMSK.

A timer event is generated every $2^{V\_EV\_TS} \cdot 250\,\mu s$ where V_EV_TS = 0..15 in register R_TI_WD. 16 timer event frequencies are available in the range 250 µs .. 8.192 s.

9.4.4.3 Processing / non-processing interrupt

XHFC-2S4U / 4SU changes every 125 µs from non-processing into processing state. When it returns to non-procesing 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 µs cycle but the distance between two consecutive interrupts changes due to the load of the internal processing machine.

9.4.4.4 Command / indication interrupt (GCI interface)

This interrupt occurs when the received indication bits of the C/I-channel have changed.

9.4.4.5 Wakeup interrupt

The wakeup pin can be enabled with V_WAK_EN = '1' in register R_PWM_MD as shown in Figure 9.10. A high level is stored in V_WAK_IRQ of register R_MISC_IRQ. This bit generates an interrupt if the mask bit V_WAK_IRQMSK = '1' in register R_MISC_IRQMSK.

The wakeup pin can also be used as common external interrupt input, or as general purpose input configurable with or without interrupt capability.

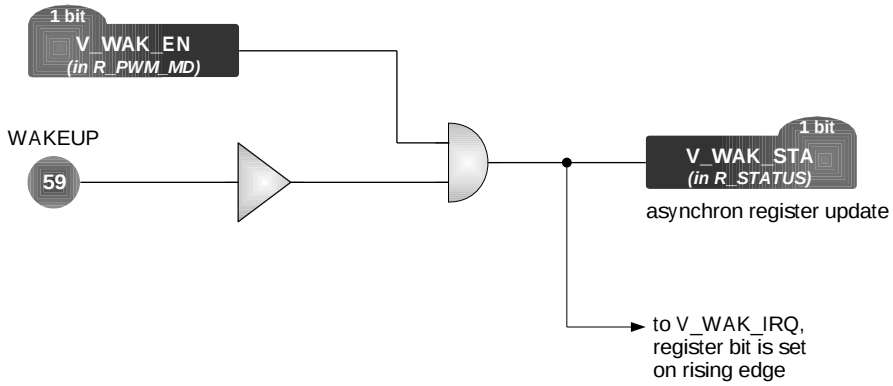


Figure 9.10: *Wakeup interrupt*

9.4.4.6 Interrupt for GCI monitor byte transmission

The interrupt bit V_MON_TX_IRQ in register R_MISC_IRQ is set to '1' when the next monitor byte can be written into register R_MON_TX.

The interrupt status is reported in V_MON_TX_IRQ of register R_MISC_IRQ even when the mask bit V_MON_TX_IRQMSK in register R_MISC_IRQMSK is not set.

9.4.4.7 Interrupt after GCI monitor byte received

After a monitor byte has been received and stored in register R_MON_RX, an interrupt can be generated.

The interrupt status is reported in V_MON_RX_IRQ of register R_MISC_IRQ even when the mask bit V_MON_RX_IRQMSK in register R_MISC_IRQMSK is not set.

9.5 Watchdog reset

The parallel processor interface of XHFC-2S4U / 4SU 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 $2^{V\_WD\_TS} \cdot 2\,ms$ where V_WD_TS = 0..15 in register R_TI_WD. This leads to a watchdog event frequency from 2 ms to 65 536 s.

9.6 Register description

9.6.1 Write only registers

R_CIRM	(w)	(Reset group: H)	0x00

Interrupt and reset register

All reset bits in this register must be cleared by software. It is not allowed to write any register while reset is pending.

Bits	Reset value	Name	Description
0	0	V_CLK_OFF	**Global clock enable / disable** '0' = all internal clocks (f_{SYS}, f_{SU} and f_{PCM}) are enabled '1' = all internal clocks are disabled This bit is reset at every write access to XHFC-2S4U / 4SU or with a wake-up signal on pin WAKEUP .
1	0	V_WAIT_PROC	**Additional /WAIT signal after write access** The wait signal gets low with every access to a register when a preceding write access is internally not yet completed. '0' = normal /WAIT signal '1' = additional /WAIT signal This bit should be set if timing problems occur with fast processors.
2	0	V_WAIT_REG	**Additional /WAIT signal during internal busy phase** The wait signal gets low with every access to a register if a preceding change FIFO, increment FIFO or reset FIFO operation has activated an internal busy phase. '0' = normal /WAIT signal '1' = additional /WAIT signal during busy phase This bit can be set when busy polling (wait until not busy) is not used by the software.
3	0	V_SRES	**Global software reset (reset group 0)** This reset sets alls registers of reset group 0 to their default value. It includes also the reset of all registers of reset groups 1 . . 3. The selected I/O address (CIP) remains unchanged. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
4	0	**V_HFC_RES**	**HFC reset (reset group 1)** Sets all FIFO and HDLC registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
5	0	**V_PCM_RES**	**PCM reset (reset group 2)** Sets all PCM registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
6	0	**V_SU_RES**	**Line interface reset (reset group 3)** Sets all registers of every line interface to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
7	0	(reserved)	Must be '0'.

R_CLK_CFG	(w)	(Reset group: H)	0x02

Clock configuration register

Bits	Reset value	Name	Description
0	0	**V_CLK_PLL**	**Clock source selection** '0' = CLK_OUT clock is derived from oscillator input OSC_IN '1' = CLK_OUT clock is derived from PLL output
1	0	**V_CLKO_HI**	**High / low frequency selection for clock output** When CLK_OUT clock is derived from oscillator input OSC_IN or F1_1 alternatively, the signal can either directly be passed to the CLK_OUT pin (high frequency) or the frequency can be divided by 8 (low frequency). '0' = low frequency, divider by 8 '1' = high frequency, no divider
2	0	**V_CLKO_PLL**	**Source selection for clock output** '0' = either OSC_IN or F1_1 signal is passed to CLK_OUT '1' = PLL output clock is passed to CLK_OUT
4..3	0	(reserved)	Must be '00'.
5	0	**V_PCM_CLK**	**Clock of the PCM module** '0' = $f_{PCM} = 2 \cdot f_{SYS}$ '1' = $f_{PCM} = f_{SYS}$ PCM clock must be set up to 49.152 MHz.
6	0	**V_CLKO_OFF**	**Clock output enable / disable** '0' = clock output pin CLK_OUT is enabled '1' = clock output pin CLK_OUT is disabled (tristate)
7	0	**V_CLK_F1**	**Pin selection for PLL input frequency** The selected pin is used for PLL input frequency as well as for CLK_OUT signal. '0' = oscillator input OSC_IN is used '1' = F1_1 input pin is used

R_MISC_IRQMSK	(w)	(Reset group: H)	0x11

Miscellaneous interrupt mask register

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 22. The interrupt status can be read from register R_MISC_IRQ nevertheless.

Bits	Reset value	Name	Description
0	0	**V_SLIP_IRQMSK**	**Interrupt mask of the line interface frequency slip**
1	0	**V_TI_IRQMSK**	**Timer interrupt mask**
2	0	**V_PROC_IRQMSK**	**Processing / non-processing transition interrupt mask** (every 125 µs)
3	0	(reserved)	Must be '0'.
4	0	**V_CI_IRQMSK**	**Command / indication interrupt mask**
5	0	**V_WAK_IRQMSK**	**Wakeup interrupt mask**
6	0	**V_MON_TX_IRQMSK**	**Transmit monitor byte interrupt mask**
7	0	**V_MON_RX_IRQMSK**	**Receive monitor byte interrupt mask**

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_SU_IRQMSK	(w)	(Reset group: H, 0, 3)	0x12

State change interrupt mask register of the line interfaces

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 22. The interrupt status can be read from register R_SU_IRQ nevertheless.

Bits	Reset value	Name	Description
0	0	**V_SU0_IRQMSK**	**Interrupt mask of line interface 0** '1' = V_SU0_IRQ in register R_SU_IRQ is used for interrupt generation
1	0	**V_SU1_IRQMSK**	**Interrupt mask of line interface 1** '1' = V_SU1_IRQ in register R_SU_IRQ is used for interrupt generation
2	0	**V_SU2_IRQMSK**	**Interrupt mask of line interface 2** '1' = V_SU2_IRQ in register R_SU_IRQ is used for interrupt generation
3	0	**V_SU3_IRQMSK**	**Interrupt mask of line interface 3** '1' = V_SU3_IRQ in register R_SU_IRQ is used for interrupt generation
7..4	0	(reserved)	Must be '0000'.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_IRQ_CTRL	(w)	(Reset group: H, 0)	0x13

Interrupt control register

Bits	Reset value	Name	Description
0	0	**V_FIFO_IRQ_EN**	**FIFO interrupt** '0' = all FIFO interrupts disabled '1' = all FIFO interrupts enabled **Note:** This bit enables or disables the interrupt capability of all FIFOs together. Each FIFO interrupt can be masked individually with V_FIFO_IRQMSK setting in register A_FIFO_CTRL.
2 . . 1	0	(reserved)	Must be '00'.
3	0	**V_GLOB_IRQ_EN**	**Global interrupt signal enable** The interrupt line is pin 22. '0' = disable '1' = enable
4	0	**V_IRQ_POL**	**Polarity of interrupt signal** '0' = active low signal '1' = active high signal
7 . . 5	0	(reserved)	Must be '000'.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_TI_WD	(w)	(Reset group: H, 0)	0x1A

Timer and watchdog control register

Bits	Reset value	Name	Description
3..0	0	**V_EV_TS**	**Timer event after** $2^n \cdot 250\,\mu s$ 0 = 250 µs 1 = 500 µs 2 = 1 ms 3 = 2 ms 4 = 4 ms 5 = 8 ms 6 = 16 ms 7 = 32 ms 8 = 64 ms 9 = 128 ms 0xA = 256 ms 0xB = 512 ms 0xC = 1.024 s 0xD = 2.048 s 0xE = 4.096 s 0xF = 8.192 s
7..4	0	**V_WD_TS**	**Watchdog event after** $2^n \cdot 2\,ms$ 0 = 2 ms 1 = 4 ms 2 = 8 ms 3 = 16 ms 4 = 32 ms 5 = 64 ms 6 = 128 ms 7 = 256 ms 8 = 512 ms 9 = 1.024 s 0xA = 2.048 s 0xB = 4.096 s 0xC = 8.192 s 0xD = 16.384 s 0xE = 32.768 s 0xF = 65.536 s

R_PLL_CTRL	(w)	(Reset group: H)	0x50

PLL control register

Bits	Reset value	Name	Description
0	0	**V_PLL_NRES**	**PLL reset (active low)** '0' = PLL reset, PLL disabled, input clock is feed to PLL output '1' = PLL enabled, a '0' to '1' transition starts the PLL beginning with the lowest oscillator frequency
1	0	**V_PLL_TST**	**PLL test input** '0' = test state disabled, normal operation '1' = test state enabled
4..2	0	(reserved)	Must be '000'.
5	0	**V_PLL_FREEZE**	**PLL standby mode** '0' = normal PLL operation '1' = PLL is in standby mode, no output clock
7..6	0	**V_PLL_M**	**Oscillator divider programming value** The oscillator output signal is divided by V_PLL_M + 1.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

9.6.2 Read only registers

R_IRQ_OVIEW	(r)	(Reset group: H, 0, 1)	0x10

FIFO interrupt overview register

This register gives an overview of all XHFC-2S4U / 4SU interrupt conditions.

Every bit value '1' indicates that an interrupt has occured. Interrupt conditions are only used for generating a signal on the interrupt pin 22 if the belonging mask bit is set to '1'.

Reading this overview register does not clear any interrupt status bit.

Bits	Reset value	Name	Description
0	0	**V_FIFO_BL0_IRQ**	**Interrupt overview of FIFO block 0** FIFO block 0 consists of transmit FIFOs 0 . . 3 and receive FIFOs 0 . . 3. The exact FIFO can be determined by reading register R_FIFO_BL0_IRQ. '0' = No FIFO interrupt occured '1' = At least one FIFO interrupt is pending.
1	0	**V_FIFO_BL1_IRQ**	**Interrupt overview of FIFO block 1** FIFO block 1 consists of transmit FIFOs 4 . . 7 and receive FIFOs 4 . . 7. The exact FIFO can be determined by reading register R_FIFO_BL1_IRQ. '0' = No FIFO interrupt occured '1' = At least one FIFO interrupt is pending.
2	0	**V_FIFO_BL2_IRQ**	**Interrupt overview of FIFO block 2** FIFO block 2 consists of transmit FIFOs 8 . . 11 and receive FIFOs 8 . . 11. The exact FIFO can be determined by reading register R_FIFO_BL2_IRQ. '0' = No FIFO interrupt occured '1' = At least one FIFO interrupt is pending.
3	0	**V_FIFO_BL3_IRQ**	**Interrupt overview of FIFO block 3** FIFO block 3 consists of transmit FIFOs 12 . . 15 and receive FIFOs 12 . . 15. The exact FIFO can be determined by reading register R_FIFO_BL3_IRQ. '0' = No FIFO interrupt occured '1' = At least one FIFO interrupt is pending.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
4	0	**V_MISC_IRQ**	**Miscellaneous interrupt overview** All enabled miscellaneous interrupts of register R_MISC_IRQ are 'ored'. '0' = No miscellaneous interrupt occured '1' = At least one miscellaneous interrupt is pending. This bit has always the same value as V_MISC_IRQSTA in register R_STATUS. Reading register R_MISC_IRQ clears the miscellaneous interrupt bits.
5	0	**V_STUP_IRQ**	**Line interface interrupt overview** '0' = No line interface interrupt occured '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)	(Reset group: H, 0, 1)	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	0	**V_SLIP_IRQ**	**Interrupt status of the line interface frequency slip** This interrupt occurs when the frame synchronization pulse of any line interface switches from F0IO 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 all line interfaces. A change can happen from time to time if two TEs operate on slightly different frame synchronization clocks. Data might be destroyed in these cases.
1	0	**V_TI_IRQ**	**Timer interrupt status** '1' = timer elapsed
2	0	**V_PROC_IRQ**	**Processing / non-processing transition interrupt status** '1' = XHFC-2S4U / 4SU has changed from processing to non-processing phase (every 125 µs). **Note:** The current processing / non-processing status can be read from V_PROC in register R_STATUS.
3		(reserved)	
4	0	**V_CI_IRQ**	**Command / indication interrupt status** '1' = received indication bits changed
5	0	**V_WAK_IRQ**	**Wakeup interrupt status** '1' = a wakeup signal at pin WAKEUP occured
6	0	**V_MON_TX_IRQ**	**Transmit monitor byte interrupt status** '1' = the next monitor byte can be written
7	0	**V_MON_RX_IRQ**	**Receive monitor byte interrupt status** '1' = monitor byte received

R_SU_IRQ	(r)	(Reset group: H, 0)	0x12

State change interrupt status register of the line interfaces

This register reports state changes of the line interfaces. Reading this register clears the bits. These bits are not masked by register R_SU_IRQMSK, i.e. they show state change conditions even if the interrupt is disabled.

Bits	Reset value	Name	Description
0	0	**V_SU0_IRQ**	**Interrupt status of line interface 0** '1' = a state change occured in line interface 0
1	0	**V_SU1_IRQ**	**Interrupt status of line interface 1** '1' = a state change occured in line interface 1
2	0	**V_SU2_IRQ**	**Interrupt status of line interface 2** '1' = a state change occured in line interface 2
3	0	**V_SU3_IRQ**	**Interrupt status of line interface 3** '1' = a state change occured in line interface 3
7 . . 4		(reserved)	

R_STATUS	(r)	(Reset group: H, 0, 3)	0x1C

XHFC-2S4U / 4SU status register

Bits	Reset value	Name	Description
0		V_BUSY	**BUSY / NOBUSY status** '0' = XHFC-2S4U / 4SU is not busy, all accesses are allowed '1' = XHFC-2S4U / 4SU is BUSY after initialising FIFO reset, increment *F*-counter or change FIFO
1		V_PROC	**Processing / non-processing status** '0' = XHFC-2S4U / 4SU has finished the processing phase during the 125 µs cycle '1' = XHFC-2S4U / 4SU is in processing phase (once every 125 µs cycle) **Note:** The processing / non-processing transition can be notified with an interrupt (see V_PROC_IRQ in register R_MISC_IRQ).
2		(reserved)	
3	0	V_LOST_STA	**LOST error (frames have been lost)** This means that XHFC-2S4U / 4SU did not process all data in 125 µs. So data may be corrupted. Bit V_RES_LOST of register A_INC_RES_FIFO must be set to reset this bit.
4		V_PCM_INIT	**PCM module initialization** '0' = initialization sequence is finished '1' = initialization sequence is in progress (after hardware reset, global software reset or PCM reset) **Note:** The PCM clocks F0IO and C4IO are ignored during initialization.
5		V_WAK_STA	**Wakeup status** This bit contains the current value of pin WAKEUP when V_WAK_EN = '1' in register R_PWM_MD.
6	0	V_MISC_IRQSTA	**Miscellaneous interrupt overview** All enabled miscellaneous interrupts of register R_MISC_IRQ are 'ored'. '0' = No miscellaneous interrupt occured '1' = At least one miscellaneous interrupt is pending This bit has always the same value as V_MISC_IRQ in register R_IRQ_OVIEW. Reading register R_MISC_IRQ clears the miscellaneous interrupt bits.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7		V_FIFO_IRQSTA	**Any FIFO interrupt** All enabled FIFO interrupts in registers R_FIFO_BL0_IRQ . . R_FIFO_BL3_IRQ are 'ored'. '0' = No FIFO interrupt is pending '1' = At least one FIFO interrupt is pending.

R_FIFO_BL0_IRQ **(r)** (Reset group: H, 0, 1) **0x20**

Interrupt status register for FIFO block 0

This register reports the interrupt status of FIFO block 0. A bit is set to '1' when an interrupt event occures.

Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.

The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).

Bits	Reset value	Name	Description
0	0	V_FIFO0_TX_IRQ	**FIFO[0,TX] interrupt status**
1	0	V_FIFO0_RX_IRQ	**FIFO[0,RX] interrupt status**
2	0	V_FIFO1_TX_IRQ	**FIFO[1,TX] interrupt status**
3	0	V_FIFO1_RX_IRQ	**FIFO[1,RX] interrupt status**
4	0	V_FIFO2_TX_IRQ	**FIFO[2,TX] interrupt status**
5	0	V_FIFO2_RX_IRQ	**FIFO[2,RX] interrupt status**
6	0	V_FIFO3_TX_IRQ	**FIFO[3,TX] interrupt status**
7	0	V_FIFO3_RX_IRQ	**FIFO[3,RX] interrupt status**

R_FIFO_BL1_IRQ	(r)	(Reset group: H, 0, 1)	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 occures.

Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.

The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).

Bits	Reset value	Name	Description
0	0	**V_FIFO4_TX_IRQ**	**FIFO[4,TX] interrupt status**
1	0	**V_FIFO4_RX_IRQ**	**FIFO[4,RX] interrupt status**
2	0	**V_FIFO5_TX_IRQ**	**FIFO[5,TX] interrupt status**
3	0	**V_FIFO5_RX_IRQ**	**FIFO[5,RX] interrupt status**
4	0	**V_FIFO6_TX_IRQ**	**FIFO[6,TX] interrupt status**
5	0	**V_FIFO6_RX_IRQ**	**FIFO[6,RX] interrupt status**
6	0	**V_FIFO7_TX_IRQ**	**FIFO[7,TX] interrupt status**
7	0	**V_FIFO7_RX_IRQ**	**FIFO[7,RX] interrupt status**

R_FIFO_BL2_IRQ	(r)	(Reset group: H, 0, 1)	0x22

Interrupt status register for FIFO block 2

This register reports the interrupt status of FIFO block 2. A bit is set to '1' when an interrupt event occures.

Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.

The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).

Bits	Reset value	Name	Description
0	0	**V_FIFO8_TX_IRQ**	**FIFO[8,TX] interrupt status**
1	0	**V_FIFO8_RX_IRQ**	**FIFO[8,RX] interrupt status**
2	0	**V_FIFO9_TX_IRQ**	**FIFO[9,TX] interrupt status**
3	0	**V_FIFO9_RX_IRQ**	**FIFO[9,RX] interrupt status**
4	0	**V_FIFO10_TX_IRQ**	**FIFO[10,TX] interrupt status**
5	0	**V_FIFO10_RX_IRQ**	**FIFO[10,RX] interrupt status**
6	0	**V_FIFO11_TX_IRQ**	**FIFO[11,TX] interrupt status**
7	0	**V_FIFO11_RX_IRQ**	**FIFO[11,RX] interrupt status**

R_FIFO_BL3_IRQ	(r)	(Reset group: H, 0, 1)	0x23

Interrupt status register for FIFO block 3

This register reports the interrupt status of FIFO block 3. A bit is set to '1' when an interrupt event occures.

Reading this register clears the bits. These bits are not masked by V_FIFO_IRQMSK in register A_FIFO_CTRL, i.e. they show the FIFO conditions even if the interrupt is disabled. But it is important to enable FIFO interrupts globally with V_FIFO_IRQ_EN = '1' in register R_IRQ_CTRL.

The interrupt condition can be configured for every FIFO individually (see bitmap V_FIFO_IRQ in register V_FIFO_IRQ and bit V_MIX_IRQ in register V_MIX_IRQ).

Bits	Reset value	Name	Description
0	0	**V_FIFO12_TX_IRQ**	**FIFO[12,TX] interrupt status**
1	0	**V_FIFO12_RX_IRQ**	**FIFO[12,RX] interrupt status**
2	0	**V_FIFO13_TX_IRQ**	**FIFO[13,TX] interrupt status**
3	0	**V_FIFO13_RX_IRQ**	**FIFO[13,RX] interrupt status**
4	0	**V_FIFO14_TX_IRQ**	**FIFO[14,TX] interrupt status**
5	0	**V_FIFO14_RX_IRQ**	**FIFO[14,RX] interrupt status**
6	0	**V_FIFO15_TX_IRQ**	**FIFO[15,TX] interrupt status**
7	0	**V_FIFO15_RX_IRQ**	**FIFO[15,RX] interrupt status**

R_PLL_STA	(r)	(Reset group: H)	0x50

PLL status register

Bits	Reset value	Name	Description
6..0		(reserved)	
7	0	**V_PLL_LOCK**	**PLL lock status** '0' = PLL is unlocked or in standby mode '1' = PLL is locked

9.6.3 Read / write registers

R_PLL_P	(r/w)	(Reset group: H)	0x51

PLL predivider programming value

Bits	Reset value	Name	Description
7..0	0	V_PLL_P	**Predivider programming value** The divisor of the predivider is V_PLL_P. The bitmap value 0 has the meaning of 256.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PLL_N	(r/w)	(Reset group: H)	0x52

PLL loop factor

Bits	Reset value	Name	Description
7..0	0	V_PLL_N	**Loop factor programming value** The loop factor is V_PLL_N. **Note:** 0..4 are not allowed.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_PLL_S	(r/w)	(Reset group: H)	0x53

PLL post-scaler programming value

Bits	Reset value	Name	Description
7..0	0	V_PLL_S	**Post-scaler programming value** The divisor of the post-scaler is V_PLL_S. The bitmap value 0 has the meaning of 256.

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

Chapter 10

General purpose I/O pins (GPIO)

Table 10.1: *Overview of the XHFC-2S4U / 4SU general purpose I/O registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x40	R_GPIO_OUT1	333	0x40	R_GPIO_IN1	342
0x41	R_GPIO_OUT3	334	0x41	R_GPIO_IN3	343
0x42	R_GPIO_EN1	334	0x45	R_GPIO_IN2	343
0x43	R_GPIO_EN3	335	0x48	R_GPIO_IN0	344
0x44	R_GPIO_SEL_BL	336			
0x45	R_GPIO_OUT2	337			
0x47	R_GPIO_EN2	337			
0x48	R_GPIO_OUT0	338			
0x4A	R_GPIO_EN0	339			
0x4C	R_GPIO_SEL	340			

10.1 GPIO functionality

XHFC-2S4U / 4SU has up to 32 general purpose I/O (GPIO) pins. 24 pins are shared with the line interfaces and every unused ST / U_p interface makes six additional GPIO pins available. Eight further pins are shared with PCM, PWM and clock functions and are individually selectable as GPIO.

Every pin listed in Table 10.2 has three functions. A detailed GPIO block diagram is shown in Figure 10.1. For GPIO8 , e.g., the following configurations are available:

- **GPIO selection bit V_GPIO_SEL0 = '0' in register R_GPIO_SEL:**

  The line interface function (1st pin function) is selected for interface #0. GPIO8 output is disabled within a group of six pins (2nd pin functions GPIO8 . . GPIO11 , GPIO16 and GPIO17 are not available).

- **GPIO selection bit V_GPIO_SEL0 = '1' in register R_GPIO_SEL and GPIO enable bit V_GPIO_EN8 = '0' in register R_GPIO_EN1:**

  The line interface function (1st pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2nd pin function). GPIO8 ouput is disabled, i.e. the output driver is tristated.

- **GPIO selection bit V_GPIO_SEL0 = '1' in register R_GPIO_SEL and GPIO enable bit V_GPIO_EN8 = '1' in register R_GPIO_EN1:**

  The line interface function (1st pin function) is disabled for interface #0. GPIO functionality is enabled for a group of six pins (2nd pin function). GPIO8 ouput is enabled.

The GPIO input functionality is always enabled, i.e. registers R_GPIO_IN0 . . R_GPIO_IN3 can always be read.

GPIO 0 . . 7 are implemented in the same way with the exception that the selection bits in register R_GPIO_SEL switch only one pin from first to second function.

Unused GPIO pins should be configured as output ports. This sets the level of the input buffer – which is always active at the same pin – to a stable level and avoids floating input effect.

10.2 GPIO output voltage

As the output drivers of the ST / U_p interfaces are supplied from an external source, the GPIO output voltage of these pins is influenced by this external source as well.

Yet it is recommended to connect the VDD_SU0 . . VDD_SU3 pins of unused ST / U_p interfaces to VDD .

For a few applications a GPIO output voltage different from VDD might be useful. In this case the external voltage must not exceed 3.6 V. Table 10.2 shows the allocation of power supply pins to the GPIO output drivers.

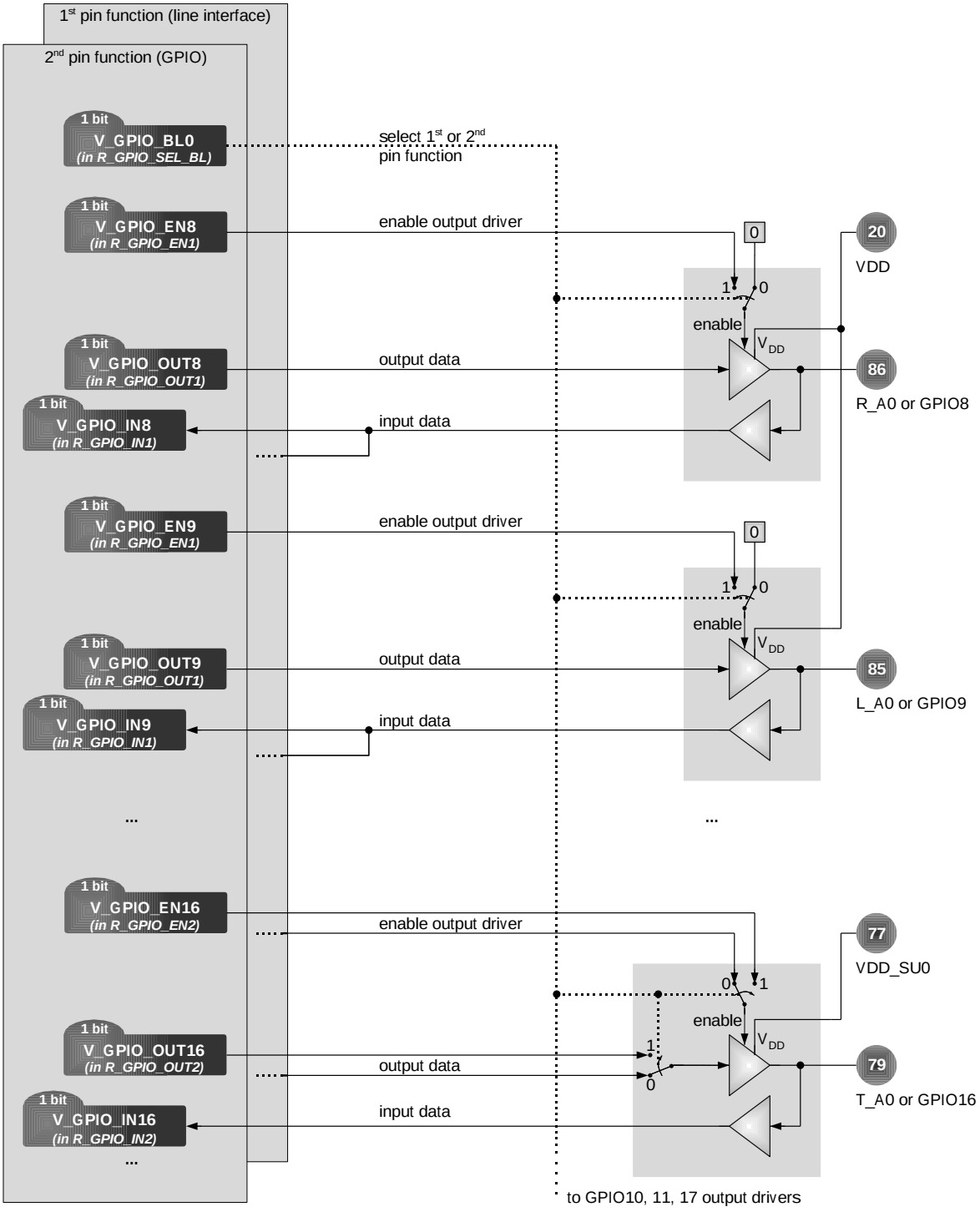


Figure 10.1: *GPIO block diagram (GPIO8 , GPIO9 and GPIO16 exemplarily)*

Table 10.2: *GPIO pins*

Pin number	GPIO pin	GPIO byte	Shared with (1st function)	Output driver power supply
61	GPIO0	0	PWM0	VDD
60	GPIO1	0	PWM1	VDD
50	GPIO2	0	F1_0	VDD
49	GPIO3	0	F1_1	VDD
33	GPIO4	0	STIO1	VDD
34	GPIO5	0	STIO2	VDD
24	GPIO6	0	SYNC_O	VDD
58	GPIO7	0	CLK_OUT	VDD
86	GPIO8	1	R_A0	VDD
85	GPIO9	1	L_A0	VDD
84	GPIO10	1	L_B0	VDD
83	GPIO11	1	R_B0	VDD
70	GPIO12	1	R_B1	VDD
69	GPIO13	1	L_B1	VDD
68	GPIO14	1	L_A1	VDD
67	GPIO15	1	R_A1	VDD
79	GPIO16	2	T_A0	VDD_SU0
78	GPIO17	2	T_B0	VDD_SU0
75	GPIO18	2	T_B1	VDD_SU1
74	GPIO19	2	T_A1	VDD_SU1
39	GPIO20	2	T_A2	VDD_SU2
38	GPIO21	2	T_B2	VDD_SU2
96	GPIO22	2	T_B3	VDD_SU3
95	GPIO23	2	T_A3	VDD_SU3
46	GPIO24	3	R_A2	VDD
45	GPIO25	3	L_A2	VDD
44	GPIO26	3	L_B2	VDD
43	GPIO27	3	R_B2	VDD
91	GPIO28	3	R_B3	VDD
90	GPIO29	3	L_B3	VDD
89	GPIO30	3	L_A3	VDD
88	GPIO31	3	R_A3	VDD

10.3 Activation state F7 / G3 signalling

Some GPIO outputs can alternatively be used to report the activation states F7 or G3. This function is typically used to drive LEDs directly from XHFC-2S4U / 4SU pins. Figure 10.2 shows exemplarily for GPIO0 , how the activation state is linked up to the GPIO signal path.

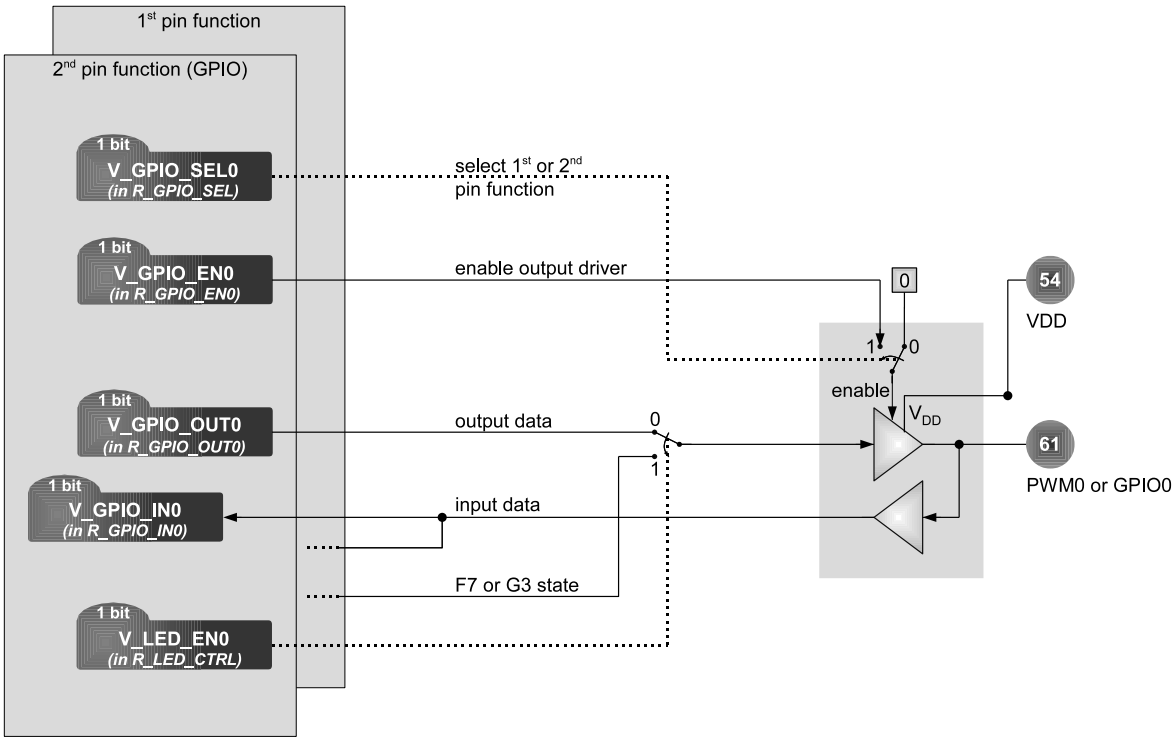


Figure 10.2: *Activation state signalling exemplarily shown with GPIO0 (similar for GPIO1 . . GPIO5)*

Table 10.3: *State F7 / G3 reporting*

	V_LED_ROT in register R_SU_LED_CTRL			
Line interface	**'00'**	**'01'**	**'10'**	**'11'**
0	GPIO0 and GPIO4	GPIO1 and GPIO5	GPIO2	GPIO3
1	GPIO1 and GPIO5	GPIO2	GPIO3	GPIO0 and GPIO4
2	GPIO2	GPIO3	GPIO0 and GPIO4	GPIO1 and GPIO5
3	GPIO3	GPIO0 and GPIO4	GPIO1 and GPIO5	GPIO2

The line activation state of each line interface numer 0 . . 3 can be assigned to pins GPIO0 . . GPIO5 by a signal rotator / shifter.

$$\text{GPIO port number} = (\text{line interface number} + \text{V\_LED\_ROT}) \; mod \; 4$$

Additionally, GPIO4 can be used to report the activation state in parallel to GPIO0 and GPIO5 can

be used to report the activation state in parallel to GPIO1 .

Table 10.3 shows the assigment of GPIO port numbers 0 . . 5 and line interface numbers 0 . . 3.

10.4 Register description

10.4.1 Write only registers

R_GPIO_OUT1	(w)	(Reset group: H, 0)	0x40

GPIO output data bits 15 . . 8

Bits	Reset value	Name	Description
0	0	V_GPIO_OUT8	Output data bit for pin GPIO8
1	0	V_GPIO_OUT9	Output data bit for pin GPIO9
2	0	V_GPIO_OUT10	Output data bit for pin GPIO10
3	0	V_GPIO_OUT11	Output data bit for pin GPIO11
4	0	V_GPIO_OUT12	Output data bit for pin GPIO12
5	0	V_GPIO_OUT13	Output data bit for pin GPIO13
6	0	V_GPIO_OUT14	Output data bit for pin GPIO14
7	0	V_GPIO_OUT15	Output data bit for pin GPIO15

R_GPIO_OUT3	(w)	(Reset group: H, 0)	0x41

GPIO output data bits 31 . . 24

Bits	Reset value	Name	Description
0	0	V_GPIO_OUT24	Output data bit for pin GPIO24
1	0	V_GPIO_OUT25	Output data bit for pin GPIO25
2	0	V_GPIO_OUT26	Output data bit for pin GPIO26
3	0	V_GPIO_OUT27	Output data bit for pin GPIO27
4	0	V_GPIO_OUT28	Output data bit for pin GPIO28
5	0	V_GPIO_OUT29	Output data bit for pin GPIO29
6	0	V_GPIO_OUT30	Output data bit for pin GPIO30
7	0	V_GPIO_OUT31	Output data bit for pin GPIO31

R_GPIO_EN1	(w)	(Reset group: H, 0)	0x42

GPIO output enable bits 15 . . 8

Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.

Bits	Reset value	Name	Description
0	0	V_GPIO_EN8	Output enable for pin GPIO8
1	0	V_GPIO_EN9	Output enable for pin GPIO9
2	0	V_GPIO_EN10	Output enable for pin GPIO10
3	0	V_GPIO_EN11	Output enable for pin GPIO11
4	0	V_GPIO_EN12	Output enable for pin GPIO12
5	0	V_GPIO_EN13	Output enable for pin GPIO13
6	0	V_GPIO_EN14	Output enable for pin GPIO14
7	0	V_GPIO_EN15	Output enable for pin GPIO15

R_GPIO_EN3	(w)	(Reset group: H, 0)	0x43

GPIO output enable bits 31 . . 24

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_EN24	Output enable for pin GPIO24
1	0	V_GPIO_EN25	Output enable for pin GPIO25
2	0	V_GPIO_EN26	Output enable for pin GPIO26
3	0	V_GPIO_EN27	Output enable for pin GPIO27
4	0	V_GPIO_EN28	Output enable for pin GPIO28
5	0	V_GPIO_EN29	Output enable for pin GPIO29
6	0	V_GPIO_EN30	Output enable for pin GPIO30
7	0	V_GPIO_EN31	Output enable for pin GPIO31

R_GPIO_SEL_BL	(w)	(Reset group: H, 0)	0x44

Selection register for GPIO block

Every line interface has six GPIO pins as second pin function which can be enabled or disabled in groups.

This register controls only the output driver, whereas the input functionality needs no programming.

Bits	Reset value	Name	Description
0	0	**V_GPIO_BL0**	**GPIO function on line interface no. 0** '0' = line interface no. 0 used (first pin function) '1' = GPIO8 . . GPIO11 , GPIO16 and GPIO17 used (second pin function)
1	0	**V_GPIO_BL1**	**GPIO function on line interface no. 1** '0' = line interface no. 1 used (first pin function) '1' = GPIO12 . . GPIO15 , GPIO18 and GPIO19 used (second pin function)
2	0	**V_GPIO_BL2**	**GPIO function on line interface no. 2** '0' = line interface no. 2 used (first pin function) '1' = GPIO24 . . GPIO27 , GPIO20 and GPIO21 used (second pin function)
3	0	**V_GPIO_BL3**	**GPIO function on line interface no. 3** '0' = line interface no. 3 used (first pin function) '1' = GPIO28 . . GPIO31 , GPIO22 and GPIO23 used (second pin function)
7 . . 4	0	(reserved)	Must be '0000'.

R_GPIO_OUT2 (w) (Reset group: H, 0) **0x45**

GPIO output data bits 23 . . 16

Bits	Reset value	Name	Description
0	0	V_GPIO_OUT16	Output data bit for pin GPIO16
1	0	V_GPIO_OUT17	Output data bit for pin GPIO17
2	0	V_GPIO_OUT18	Output data bit for pin GPIO18
3	0	V_GPIO_OUT19	Output data bit for pin GPIO19
4	0	V_GPIO_OUT20	Output data bit for pin GPIO20
5	0	V_GPIO_OUT21	Output data bit for pin GPIO21
6	0	V_GPIO_OUT22	Output data bit for pin GPIO22
7	0	V_GPIO_OUT23	Output data bit for pin GPIO23

R_GPIO_EN2 (w) (Reset group: H, 0) **0x47**

GPIO output enable bits 23 . . 16

Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.

Bits	Reset value	Name	Description
0	0	V_GPIO_EN16	Output enable for pin GPIO16
1	0	V_GPIO_EN17	Output enable for pin GPIO17
2	0	V_GPIO_EN18	Output enable for pin GPIO18
3	0	V_GPIO_EN19	Output enable for pin GPIO19
4	0	V_GPIO_EN20	Output enable for pin GPIO20
5	0	V_GPIO_EN21	Output enable for pin GPIO21
6	0	V_GPIO_EN22	Output enable for pin GPIO22
7	0	V_GPIO_EN23	Output enable for pin GPIO23

R_GPIO_OUT0	(w)	(Reset group: H, 0)	0x48

GPIO output data bits 7 . . 0

Bits	Reset value	Name	Description
0	0	V_GPIO_OUT0	Output data bit for pin GPIO0
1	0	V_GPIO_OUT1	Output data bit for pin GPIO1
2	0	V_GPIO_OUT2	Output data bit for pin GPIO2
3	0	V_GPIO_OUT3	Output data bit for pin GPIO3
4	0	V_GPIO_OUT4	Output data bit for pin GPIO4
5	0	V_GPIO_OUT5	Output data bit for pin GPIO5
6	0	V_GPIO_OUT6	Output data bit for pin GPIO6
7	0	V_GPIO_OUT7	Output data bit for pin GPIO7

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_GPIO_EN0	(w)	(Reset group: H, 0)	0x4A

GPIO output enable bits 7 . . 0

Every bit value '1' enables the allocated output driver. If an output driver is disabled (bit value '0'), the pin is used for data input.

Bits	Reset value	Name	Description
0	0	V_GPIO_EN0	Output enable for pin GPIO0
1	0	V_GPIO_EN1	Output enable for pin GPIO1
2	0	V_GPIO_EN2	Output enable for pin GPIO2
3	0	V_GPIO_EN3	Output enable for pin GPIO3
4	0	V_GPIO_EN4	Output enable for pin GPIO4
5	0	V_GPIO_EN5	Output enable for pin GPIO5
6	0	V_GPIO_EN6	Output enable for pin GPIO6
7	0	V_GPIO_EN7	Output enable for pin GPIO7

R_GPIO_SEL	(w)	(Reset group: H, 0)	0x4C

GPIO selection register

This register controls only the output driver, whereas the input functionality needs no programming.

Bits	Reset value	Name	Description
0	0	**V_GPIO_SEL0**	**Selection of first or second pin function** '0' = PWM0 (first pin function) '1' = GPIO0 (second pin function)
1	0	**V_GPIO_SEL1**	**Selection of first or second pin function** '0' = PWM1 (first pin function) '1' = GPIO1 (second pin function)
2	0	**V_GPIO_SEL2**	**Selection of first or second pin function** '0' = F1_0 (first pin function) '1' = GPIO2 (second pin function)
3	0	**V_GPIO_SEL3**	**Selection of first or second pin function** '0' = F1_1 (first pin function) '1' = GPIO3 (second pin function)
4	0	**V_GPIO_SEL4**	**Selection of first or second pin function** '0' = STIO1 (first pin function) '1' = GPIO4 (second pin function)
5	0	**V_GPIO_SEL5**	**Selection of first or second pin function** '0' = STIO2 (first pin function) '1' = GPIO5 (second pin function)
6	0	**V_GPIO_SEL6**	**Selection of first or second pin function** '0' = SYNC_O (first pin function) '1' = GPIO6 (second pin function)
7	0	**V_GPIO_SEL7**	**Selection of first or second pin function** '0' = CLK_OUT (first pin function) '1' = GPIO7 (second pin function)

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

R_SU_LED_CTRL	(w)	(Reset group: H, 0)	0x4D

LED control register

This register can be used to indicate activated states F7 or G3 on four pins of GPIO0 .. GPIO5 .
Activated state is indicated by high level. LEDs with series resistor can directly be connected to XHFC-2S4U / 4SU.

Bits	Reset value	Name	Description
0	0	**V_LED_EN0**	**Enable LED output signal on pin GPIO0** '0' = normal GPIO function on pin GPIO0 '1' = activated state is indicated on pin GPIO0
1	0	**V_LED_EN1**	**Enable LED output signal on pin GPIO1** '0' = normal GPIO function on pin GPIO1 '1' = activated state is indicated on pin GPIO1
2	0	**V_LED_EN2**	**Enable LED output signal on pin GPIO2** '0' = normal GPIO function on pin GPIO2 '1' = activated state is indicated on pin GPIO2
3	0	**V_LED_EN3**	**Enable LED output signal on pin GPIO3** '0' = normal GPIO function on pin GPIO3 '1' = activated state is indicated on pin GPIO3
4	0	**V_LED_EN4**	**Enable LED output signal on pin GPIO4** '0' = normal GPIO function on pin GPIO4 '1' = activated state is indicated on pin GPIO4
5	0	**V_LED_EN5**	**Enable LED output signal on pin GPIO5** '0' = normal GPIO function on pin GPIO5 '1' = activated state is indicated on pin GPIO5
7 .. 6	0	**V_LED_ROT**	**LED output rotator / shifter** Activated state of the line interfaces can be assigned to pins GPIO0 .. GPIO5 via a signal rotator / shifter. When I_n is the line interface with number n: '00' = I_0 is assigned to GPIO0 and GPIO4 , I_1 is assigned to GPIO1 and GPIO5 , I_2 is assigned to GPIO2 , I_3 is assigned to GPIO3 '01' = I_0 is assigned to GPIO1 and GPIO5 , I_1 is assigned to GPIO2 , I_2 is assigned to GPIO3 , I_3 is assigned to GPIO0 and GPIO4 '10' = I_0 is assigned to GPIO2 , I_1 is assigned to GPIO3 , I_2 is assigned to GPIO0 and GPIO4 , I_3 is assigned to GPIO1 and GPIO5 '11' = I_0 is assigned to GPIO3 , I_1 is assigned to GPIO0 and GPIO4 , I_2 is assigned to GPIO1 and GPIO5 , I_3 is assigned to GPIO2

(See Section 13 on page 351 for a fault description and workaround of an address decoding problem which concerns this register among others.)

10.4.2 Read only registers

R_GPIO_IN1	(r)	(Reset group: –)	0x40

GPIO input data bits 15 . . 8

Note: Unused GPIO pins should be configured as output pins.

Bits	Reset value	Name	Description
0		V_GPIO_IN8	Input data bit from pin GPIO8
1		V_GPIO_IN9	Input data bit from pin GPIO9
2		V_GPIO_IN10	Input data bit from pin GPIO10
3		V_GPIO_IN11	Input data bit from pin GPIO11
4		V_GPIO_IN12	Input data bit from pin GPIO12
5		V_GPIO_IN13	Input data bit from pin GPIO13
6		V_GPIO_IN14	Input data bit from pin GPIO14
7		V_GPIO_IN15	Input data bit from pin GPIO15

R_GPIO_IN3	(r)	(Reset group: –)	0x41

GPIO input data bits 31 . . 24

Note: Unused GPIO pins should be configured as output pins.

Bits	Reset value	Name	Description
0		V_GPIO_IN24	Input data bit from pin GPIO24
1		V_GPIO_IN25	Input data bit from pin GPIO25
2		V_GPIO_IN26	Input data bit from pin GPIO26
3		V_GPIO_IN27	Input data bit from pin GPIO27
4		V_GPIO_IN28	Input data bit from pin GPIO28
5		V_GPIO_IN29	Input data bit from pin GPIO29
6		V_GPIO_IN30	Input data bit from pin GPIO30
7		V_GPIO_IN31	Input data bit from pin GPIO31

R_GPIO_IN2	(r)	(Reset group: –)	0x45

GPI input data bits 23 . . 16

Note: Unused GPIO pins should be configured as output pins.

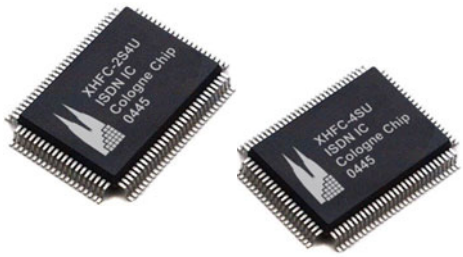
Bits	Reset value	Name	Description
0		V_GPIO_IN16	Input data bit from pin GPIO16
1		V_GPIO_IN17	Input data bit from pin GPIO17
2		V_GPIO_IN18	Input data bit from pin GPIO18
3		V_GPIO_IN19	Input data bit from pin GPIO19
4		V_GPIO_IN20	Input data bit from pin GPIO20
5		V_GPIO_IN21	Input data bit from pin GPIO21
6		V_GPIO_IN22	Input data bit from pin GPIO22
7		V_GPIO_IN23	Input data bit from pin GPIO23

R_GPIO_IN0	(r)	(Reset group: –)	0x48

GPIO input data bits 7 . . 0

Note: Unused GPIO pins should be configured as output pins.

Bits	Reset value	Name	Description
0		V_GPIO_IN0	Input data bit from pin GPIO0
1		V_GPIO_IN1	Input data bit from pin GPIO1
2		V_GPIO_IN2	Input data bit from pin GPIO2
3		V_GPIO_IN3	Input data bit from pin GPIO3
4		V_GPIO_IN4	Input data bit from pin GPIO4
5		V_GPIO_IN5	Input data bit from pin GPIO5
6		V_GPIO_IN6	Input data bit from pin GPIO6
7		V_GPIO_IN7	Input data bit from pin GPIO7



Chapter 11

Electrical characteristics

Absolute maximum ratings [*1]

Parameter	Symbol	Min.	Max.	Conditions
Power supply	V_{DD}	-0.3 V	+4.6 V	
Input voltage	V_I	-0.3 V	$V_{DD} + 0.3$ V (≤ 4.6 V)	3.3 V pins
	V_I	-0.3 V	6.0 V	5 V tolerant pins
Output voltage	V_O	-0.3 V	$V_{DD} + 0.3$ V	
Storage temperature	T_{stg}	-55 °C	+125 °C	

Recommended operating conditions

Parameter	Symbol	Min.	Typ.	Max	Conditions
Power supply	V_{DD}	3.0 V	3.3 V	3.6 V	
Operating temperature	T_{opr}	-30 °C		+85 °C	

Electrical characteristics for 3.3 V power supply

Parameter	Symbol	Min.	Typ.	Max	Conditions
Low input voltage [*2]	V_{IL}	-0.3 V		0.8 V	
High input voltage [*3]	V_{IH}	2.0 V		$V_{DD} + 0.3$ V	
High input voltage [*4]	V_{IH}	2.0 V		5.5 V	
Low output voltage [*2]	V_{OL}			0.4 V	
High output voltage [*2]	V_{OH}	2.4 V			

[*1]: Stresses beyond those listed under 'Absolute maximum ratings' may cause permanent damage to the device. These are stress ratings only, and operation of the device at these or at any other conditions above those given in this data sheet is not implied. Exposure to limiting values for extended periods may affect device reliability.

[*2]: All pins except oscillator and S/T / U_p type.

[*3]: All pins except oscillator, S/T / U_p type and 5 V tolerant pins.

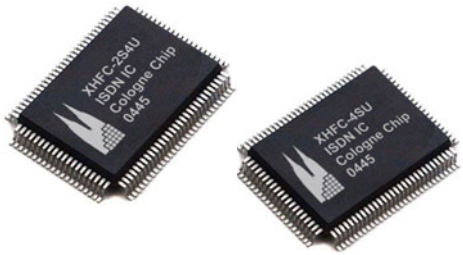
[*4]: Only 5 V tolerant pins.

Power consumption for 3.3 V power supply

Parameter	Symbol	Min.	Typ.	Max	Conditions
Oscillator running with 24.576 MHz, internal clock disabled	I_{opr}		4 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running	I_{opr}		19 mA		21 °C
Oscillator running with 24.576 MHz, internal clock enabled, FIFOs and PCM 30 clocks running, all S/T interfaces sending 96 kHz test signal on 50 Ω load	I_{opr}		59 mA		21 °C
Oscillator stopped	I_{opr}		1 mA		21 °C
Power consumption of PLL only with $f_{OSC} = 74$ MHz and $M = 1$	I_{opr}		7 mA		21 °C
Power consumption of PLL only with $f_{OSC} = 74$ MHz and $M = 3$	I_{opr}		5 mA		21 °C

Thermal package characteristics

Parameter	Symbol	Min.	Typ.	Max	Conditions
Junction to air	Θ_{JA}		+54 °C/W		Multi-Layer PCB
Junction to case	Θ_{JC}		+22 °C/W		Multi-Layer PCB



Chapter 12

XHFC-2S4U / 4SU package dimensions

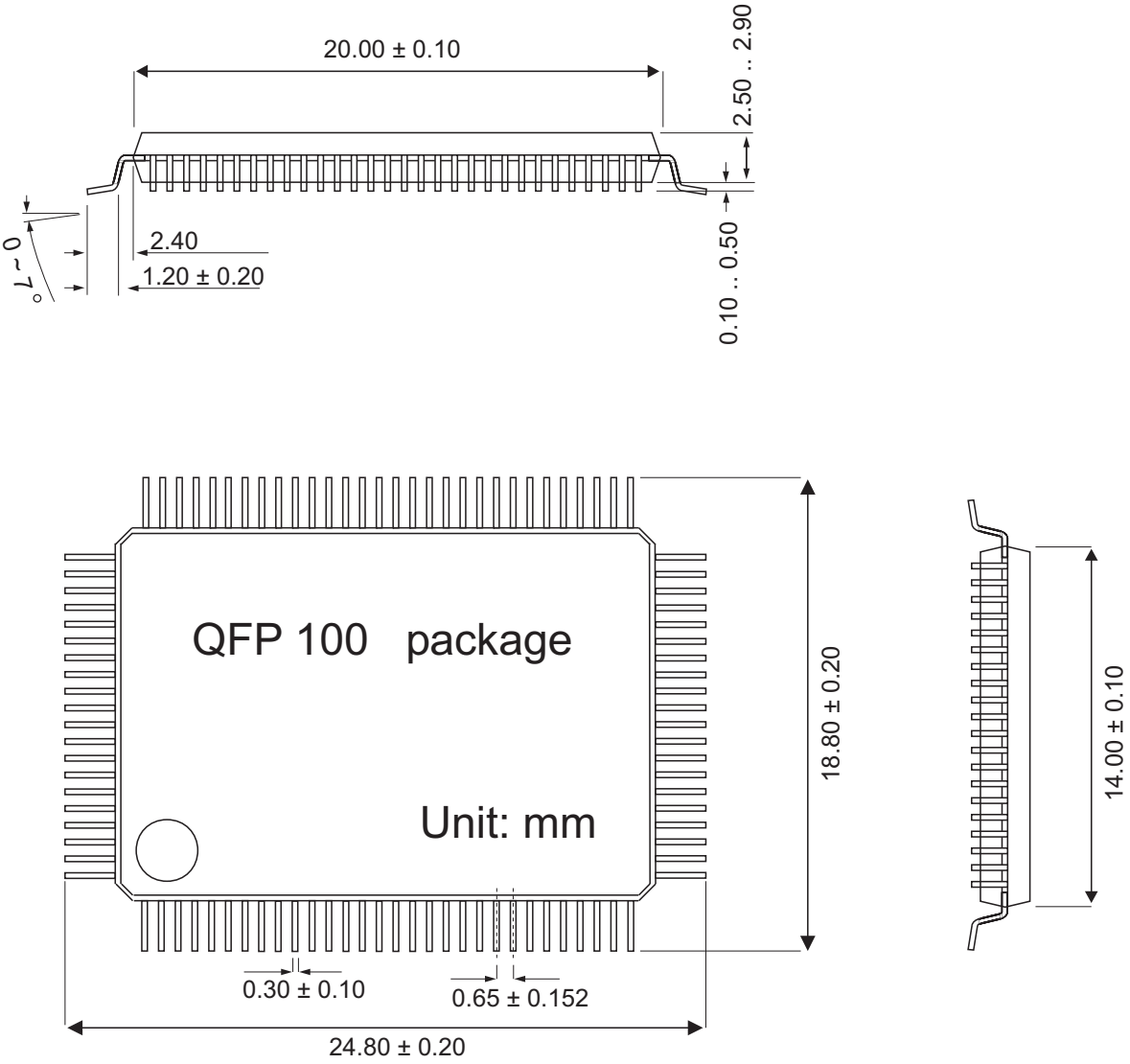
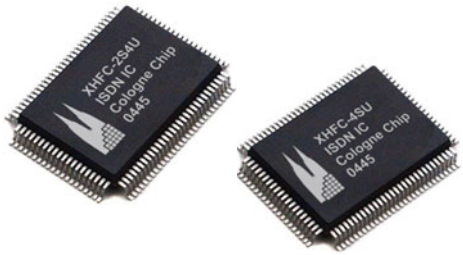


Figure 12.1: *XHFC-2S4U and XHFC-4SU package dimensions*



Chapter 13

XHFC-2S4U / 4SU address decoding erratum

13.1 Fault description

XHFC-2S4U / 4SU has an address decoding problem with write access to some registers. This erratum chapter describes the problem as well as work-arounds to avoid unwanted chip behaviour.

The address decoding fault refers to register addresses 0x48 . . 0x5F. Any write access to these registers writes also to their counterregisters at in the address range 0x08 . . 0x1F.

Due to the fact, that some addresses are not used by the chips, there are only few registers involved as shown in Table 13.1.

Table 13.1: *Involved registers of the address decoding fault*

Write access to		Writes also to	
0x48	R_GPIO_OUT0	0x08	R_RAM_ADDR
0x4C	R_GPIO_SEL	0x0C	R_FIFO_THRES
0x4D	R_SU_LED_CTRL	0x0D	R_FIFO_MD
0x50	R_PLL_CTRL	0x10	R_SLOT
0x51	R_PLL_P	0x11	R_MISC_IRQMSK
0x52	R_PLL_N	0x12	R_SU_IRQMSK
0x53	R_PLL_S	0x13	R_IRQ_CTRL

Read-only registers are not concerned in this fault.

13.2 Work-Around

The general work-around is to write registers in the address range 0x48 . . 0x5F only once during chip initialization. The counterregisters 0x08 . . 0x1F should be written afterwards. Exceptionally, register R_GPIO_OUT0 (0x48) can be written at any time, because it's counterregister R_RAM_ADDR (0x08) is normally not used.

Table 13.2 shows a complete list of the involved register adresses. Only gray marked lines are concerned in the decoding fault. In these cases, a detailed work-around is described to ensure a faultless chip behaviour.

Table 13.2: *Detailed register list and work-around description*

Write access to		Writes also to		Remark
0x48	R_GPIO_OUT0	0x08	R_RAM_ADDR	R_RAM_ADDR must be written again after R_GPIO_OUT0 write access. Typically, there is no reason for using register R_RAM_ADDR, so that any write access to register R_GPIO_OUT0 causes no problem.
0x49	–	0x09	R_RAM_CTRL	Address 0x49 is not used for write access.
0x4A	R_GPIO_EN0	0x0A	–	Counterregister 0x0A is not available.
0x4B	–	0x0B	R_FIRST_FIFO	Address 0x4B is not used for write access.
0x4C	R_GPIO_SEL	0x0C	R_FIFO_THRES	Register R_GPIO_SEL should be written during chip initialization. Typically, there is no need to write R_GPIO_SEL later once more. If necessary, register R_FIFO_THRES must be rewritten afterwards and unwanted FIFO fill levels might be reported in registers R_FILL_BL0 . . R_FILL_BL3. Software should always check the actual FIFO fill level before writing or reading FIFO bytes and should not rely on the expected FIFO fill level threshold.

(continued on next page)

Table 13.2: *Detailed register list and work-around description*

(continued from previous page)

Write access to		Writes also to		Remark
0x4D	R_SU_LED_CTRL	0x0D	R_FIFO_MD	Register R_SU_LED_CTRL should be written during chip initialization. Typically, there is no need to write R_SU_LED_CTRL later once more. If necessary in particular applications, write access to R_SU_LED_CTRL is only allowed when no FIFO is in use because this destroys the overall FIFO setup. Write access to this pair of registers must fulfill strong timing constraints as follows: 1. Configure FIFO mode by writing R_FIFO_MD. 2. Initiate a global software reset to accept this setting. 3. Configure R_SU_LED_CTRL within a not-busy phase (V_BUSY = '0' in register R_STATUS). This write access is not allowed while V_BUSY = '1'! 4. Re-write R_FIFO_MD within the same not-busy phase. As the FIFO setup is evaluated only during busy phases, it is important to restore the value immediately after R_SU_LED_CTRL write access.
0x4E	–	0x0E	A_INC_RES_FIFO	Addresses 0x4E and 0x4F are not used for write access.
0x4F	–	0x0F	–	
0x50	R_PLL_CTRL	0x10	R_SLOT	Typically, register R_PLL_CTRL is written during initialization. R_SLOT should be written afterwards. Applications that use the PLL's enable / disable function should ensure that PCM time slot selection with register R_SLOT and following access to PCM slot array registers is not interrupted by R_PLL_CTRL write access.
0x51	R_PLL_P	0x11	R_MISC_IRQMSK	It is strongly recommended to write registers R_PLL_P, R_PLL_N and R_PLL_S during initialization only. The counterpart registers R_MISC_IRQMSK, R_SU_IRQMSK and R_IRQ_CTRL should be initialized afterwards to avoid unwanted interrupt generation, interrupt suppression or unwanted interrupt controller behaviour.
0x52	R_PLL_N	0x12	R_SU_IRQMSK	
0x53	R_PLL_S	0x13	R_IRQ_CTRL	

(continued on next page)

Table 13.2: *Detailed register list and work-around description*

(continued from previous page)

Write access to		Writes also to		Remark
0x54	–	0x14	R_PCM_MD0	Addresses 0x54..0x5F are not used for write access.
0x55	–	0x15	(11 multi-registers)	
0x56	–	0x16	R_SU_SEL	
0x57	–	0x17	R_SU_SYNC	
0x58	–	0x18	–	
0x59	–	0x19	–	
0x5A	–	0x1A	R_TI_WD	
0x5B	–	0x1B	R_BERT_WD_MD	
0x5C	–	0x1C	–	
0x5D	–	0x1D	–	
0x5E	–	0x1E	R_PWM_CFG	
0x5F	–	0x1F	–	

References

[1] Cologne Chip AG. *DIGICC^{TM} PLL Technology*, November 2004.

[2] Cologne Chip AG. *C3-PLL-2– Phase-locked Loop (PLL) Frequency Synthesizer IP Core*, June 2005.

[3] Electronic Manufacturing Industry Association of Germany, Information & Communication Technology Group (ZVEI Zentralverband Elektrotechnik, Fachverband Informations- und Kommunikationstechnik (I+K Forum)). *ZVEI Documentation DKZ-N; Interfaces and signaling protocols for ISDN telecommunication installations. (only available in German under title: ZVEI-Dokumentation DKZ-N; Schnittstellen und Signalisierungsprotokolle für Telekommunikationsanlagen im ISDN)*, Mai 1989. Volume IV: DKZ-N part 1.2, DKZ-N2 part 2.2.

[4] European Telecommunications Standards Institute. *Technical Basis for Regulation (TBR 3): Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment to connect to an ISDN using ISDN basic access*.

[5] European Telecommunications Standards Institute. *Technical Basis for Regulation (TBR 4): Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment to connect to an ISDN using ISDN primary rate access*.

[6] Infineon Technologies. *PEB 2096 Data Sheet; ICs for Communications: Octal Transceiver for Upn Interfaces (OCTAT-P)*, 2.1 edition, April 1999.

[7] Siemens AG. *ICs for Communication. IOM®-2 Interface Reference Guide*, 3 1991.

[8] Telecommunication Standardization Sector of International Telecommunication Union (ITU). *ITU-T V.27: Data communication over the telephone network; 4800 bits per second MODEM with manual equalizer standardized for use on leased telephone-type circuits*, 1988, 1993.

[9] Telecommunication Standardization Sector of International Telecommunication Union (ITU). *ITU-T I.430: Integrated services digital network (ISDN); ISDN user-network interfaces. Basic user-network interface – Layer 1 specification*, November 1995.

[10] Telecommunication Standardization Union (ITU). O.151: Error performance measuring equipment operating at the primary rate and above. In *Specification of measurement equipment*. ITU-T (Telecommunication Standardization Sector of ITU), 1992.

[11] Telecommunication Standardization Union (ITU). O.150: Equipment for the measurement of digital and analogue/digital parameters. In *General requirements for instrumentation for performance measurements on digital transmission equipment*. ITU-T (Telecommunication Standardization Sector of ITU), 1996.

List of register and bitmap abbreviations

This list shows all abbreviations which are used to define the register and bitmap names. Appended digits are not shown here except they have a particular meaning.

16KHZ	16 kHz
2KHZ	2 kHz
AB	A/B-bit
ABO	abort
ACT	active, activation
ADDR	address
ADJ	adjust
AF0	alternative frame synchronization signal
AUTO	automatic
B1	B1-channel
B12	B1- and B2-channels
B2	B2-channel
BAC	BAC-bit
BERT	bit error rate test
BIT	bit
BL	block
BUSY	busy
C	command bits of the C/I-channel
C2I	C2 clock input (PCM bit clock)
C2O	C2 clock output (PCM bit clock)
C4	C4IO clock (PCM double bit clock)
CFG	configuration
CH	HFC-channel
CHANNEL	HFC-channel
CHIP	microchip
CI	Command / Indication (C/I channel of the GCI interface)
CI6	6 bit C/I-channel length
CIRM	configuration, interrupt and reset
CLK	clock
CLKO	clock output
CNT	counter
CNTH	counter, high byte
CNTL	counter, low byte
CON	connection settings
CONT	contention
CRC	cyclic redundancy check
CTRL	control
D	D-channel
DATA	data
DC	DC-balancing bit
DF	data flow
DIR	direction
DLL	double last look criterion
DLY	delay
DLYH	delay, high byte
DLYL	delay, low byte
DONE	done
DR	data rate
DST	destination
E	E-channel
ECH	error counter, high byte
ECL	error counter, low byte
EN	enable
END	end
ERR	error
EV	event
EXCHG	exchange
EXP	expired
F	*F*-counter
F0	frame synchronization signal
F1	F1_1 pin
F1	*F*1-counter
F2	*F*2-counter
FDIR	direction (FIFO-related)
FIFO	FIFO
FILL	fill level
FIRST	first
FLOW	flow
FNUM	number (FIFO-related)
FR	frame
FREEZE	freeze
FRQ	frequency
FSM	FIFO sequence mode
G2	G2 state

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

STUP	S/T / U_p interface)
SU	Universal ISDN Port (combined S/T and U_p interface)
SUBCH	subchannel
SWAP	swap
SYNC	synchronize, synchronization
SYNCI	synchronization input signal
T	T-bits
T2	S/T timer T2
THRES	threshold
TI	timer
TRI	tristate
TRP	transparent
TS	time step
TST	test
TX	transmit
TXHS	transmitter handshake
TXR	transmitter ready
UNIDIR	unidirectional
UP	U_p interface
USAGE	usage
USE	use, usage
VAL	value
VIO	code violation
WAIT	wait
WAK	wakeup
WD	watchdog timer
WR	write
Z1	*Z*1-counter
Z2	*Z*2-counter

Index of register and bitmap names

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

Index of pin names

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

Cologne Chip AG

Data Sheet of XHFC-2S4U / 4SU